UNIT 9 WEB BASED CONTENT DEVELOPMENT

Structure

9.0 OBJECTIVES

The objective of this unit is to give an overview of technologies used in Web content development. After studying this unit you will be able to :

- know the different technologies used in content development;
- understand markup languages like HTML, DHTML, XML;
- understand the linking of databases through Web interface;
- understand how websites are hosted on the Web; and
- know the tools used in designing Web pages.

9.1 INTRODUCTION

Hypertext Markup Language (HTML) is a structured markup language that is used to create Web pages. A markup language such as HTML is simply a collection of codes called elements that are used to indicate the structure and format of a document. Elements in HTML consist of alphanumeric tokens within angular brackets, such as , <html>, <body>, etc.

Most elements consist of paired tags: a start tag and an end tag. For example, is a start tag and is the end tag. The end tag is similar to start tag, except that the symbol is preceded by forward slash. An element's instruction applies to whatever content is contained between its start and end tags:

Eg. **** This text is bold but this text is not.

Element names are not case-sensitive. An element such as <hTml> is equivalent to <html>. However using either upper or lower case consistently makes HTML documents easier to understand and maintain. Element names cannot contain spaces. Unit 8 of this block provides further insight into HTML.

9.2 BASIC HTML TAGS

Lets see some of the basic HTML tags that are used in developing HTML documents.

Markup Tags

HTML

This element tells the browser that the file contains HTML-coded information. The file extension .html and .htm also indicates this is an HTML document.

Head

The head element identifies the first part of your HTML-coded document that contains the title. The title is shown in the title bar of browser's window.

Title

The title element contains your document title and identifies its content in a global context. The title is typically displayed in the title bar at the top of the browser window, but not inside the window itself. The title is also what is displayed on someone's hotlist or bookmark list, so choose something descriptive, unique, and relatively short. A title is also used to identify your page for search engines (such as HotBot or Infoseek).

Generally you should keep your titles to 64 characters or fewer.

Body

The second—and largest—part of your HTML document is the body, which contains thecontent of your document (displayed within the text area of your browser window). The tags explained below are used within the body of your HTML document.

Headings

HTML has six levels of headings, numbered 1 through 6, with 1 being the largest. Headings are typically displayed in larger and/or bolder fonts than normal body text. The first heading in each document should be tagged <H1>.

The syntax of the heading element is:

<Hy>*Text of heading* </Hy>

Where *y* is a number between 1 and 6 specifying the level of the heading.

Do not skip levels of headings in your document. For example, don't start with a level-one heading (<H1>) and then next use a level-three (<H3>) heading.

For example:

<html>

<head>

<title>IGNOU Homepage</title>

</head>

<body> Welcome to the Home Page of IGNOU. IGNOU is one of the open universities of India providing distance education courses in different fields. </ body>

<html>

Paragraphs

Unlike documents in most word processors, carriage returns in HTML files aren't significant. In fact, any amount of *whitespace* — including spaces, linefeeds, and carriage returns — are automatically compressed into a single space when your HTML document is displayed in a browser. So you don't have to worry about how long your lines of text are. Word wrapping can occur at any point in your source file without affecting how the page will be displayed.

<P>Welcome to the world of HTML.

This is the first paragraph.

While short it is

still a paragraph!</P>

In the source file there is a line break between the sentences. A Web browser ignores this line break and starts a new paragraph only when it encounters another <P> tag.

Important: You must indicate paragraphs with <P> elements. A browser ignores

any indentations or blank lines in the source text. Without <P> elements, the document becomes one large paragraph. (One exception is text tagged as "preformatted," which is explained below.) For example, the following would produce identical output as the first example:

```
<H1>Level-one heading</H1>
<P>Welcome to the world of HTML. This is the
first paragraph. While short it is still a paragraph!
</P> <P>And this is second paragraph.</P>
```

NOTE: The </P> closing tag may be omitted. This is because browsers understand that when they encounter a <P> tag, it means that the previous paragraph has ended. However, since HTML now allows certain attributes to be assigned to the <P> tag, it's generally a good idea to include it.

Using the <P> and </P> as a paragraph container means that you can center a paragraph by including the ALIGN=*alignment* attribute in your source file.

```
<P ALIGN=CENTER>
This is a centered paragraph.
</P>
```

This is a centered paragraph.

It is also possible to align a paragraph to the right instead, by including the ALIGN=RIGHT attribute. ALIGN=LEFT is the default alignment; if no ALIGN attribute is included, the paragraph will be left-aligned.

Lists

HTML supports unnumbered, numbered, and definition lists. You can nest lists too, but use this feature sparingly because too many nested items can get difficult to follow.

Unnumbered Lists

To make an unnumbered, bulleted list,

1) Start with an opening list <UL> (for unnumbered list) tag
2) Enter the <LI> (list item) tag followed by the individual item; no closing </LI> tag is needed
3) End the entire list with a closing list </UL> tag

Below is a sample three-item list:

```
<UL>
<LI> apples
<LI> bananas
<LI> grapefruit
</UL>
```

The output is:

- apples
- bananas
- grapefruit

The <LI> items can contain multiple paragraphs. Indicate the paragraphs with the <P> paragraph tags.

Numbered Lists

A numbered list (also called an *ordered list*, from which the tag name derives) is identical to an unnumbered list, except it uses <OL> instead of <UL>. The items are tagged using the same <LI> tag. The following HTML code:

<OL>
<LI> oranges
<LI> peaches
<LI> grapes
</OL>

produces this formatted output:

1) oranges
2) peaches
3) grapes

Definition Lists

A definition list (coded as <DL>) usually consists of alternating a *definition term* (coded as <DT>) and a definition description (coded as <DD>). Web browsers generally format the definition on a new line and indent it.

The following is an example of a definition list:

<DL>
<DT> IGNOU
<DD> IGNOU, Indira Gandhi National Open University is located in New Delhi.
<DT> IISc
<DD> IISc, the Indian Institute of Science is located in Bangalore
</DL>

The output looks like:

IGNOU

IGNOU, Indira Gandhi National Open University is located in New Delhi.

IISc

IISc, the Indian Institute of Science is located in Bangalore.

The <DT> and <DD> entries can contain multiple paragraphs (indicated by <P> paragraph tags), lists, or other definition information.

Nested Lists

Lists can be nested. You can also have a number of paragraphs, each containing a nested list, in a single list item.

Here is a sample nested list:

```
<UL>
<LI> A few Indian states:
<UL>
<LI> Haryana
<LI>Tamilnadu
<LI> Andhra Pradesh
</UL>
<LI> Two Eastern states:
<UL>
<LI> Assam
<LI> West Bengal
</UL>
</UL>
```

The nested list is displayed as

- A few Indian states:
  - Haryana
  - Tamilnadu
  - Andhra Pradesh
- Two Midwestern states:
  - Assam
  - West Bengal

Self Check Exercise

1) What is HTML? Mention some basic HTML tags.
2) Small exercise of Lists?

.....

.....

.....

.....

9.2.1 Linking

The chief power of HTML comes from its ability to link text and/or an image to another document or section of a document. A browser highlights the identified text or image with colour and/or underlines to indicate that it is a *hypertext link* (often shortened to *hyperlink* or just *link*).

HTML's single hypertext-related tag is <A>, which stands for *anchor*. To include an anchor in your document:

1) Start the anchor with <A (include a space after the A)

2) *Specify the document you're linking to by entering the parameter HREF="*filename*" followed by a closing right angle bracket (>)*

3) Enter the text that will serve as the hypertext link in the current document

4) Enter the ending anchor tag: </A> (no space is needed before the end anchor tag)

Here is a sample hypertext reference in a file called US.html:

```
<A HREF="hello.html">Hello</A>
```

This entry makes the word *Hello* the hyperlink to the document hello.html, which is in the same directory as the first document.

9.2.1.1 Relative Pathnames Versus Absolute Pathnames

You can link to documents in other directories by specifying the *relative path* from the current document to the linked document. For example, a link to a file assam.html located in the subdirectory *temp* would be:

```
<A HREF="temp/assam.html">Content </A>
```

These are called *relative links* because you are specifying the path to the linked file relative to the location of the current file. You can also use the absolute pathname (the complete URL) of the file, but relative links are more efficient in accessing a server.

They also have the advantage of making your documents more "portable" — for instance, you can create several web pages in a single folder on your local computer, using relative links to hyperlink one page to another, and then upload the entire folder of web pages to your web server. The pages on the server will then link to other pages on the server, and the copies on your hard drive will still point to the other pages stored there.

It is important to point out that UNIX is a case-sensitive operating system where filenames are concerned, while DOS and the MacOS are not. For instance, on a Macintosh, "DOCUMENT.HTML", "Document.HTML", and "document.html" are all the same name. If you make a relative hyperlink to "DOCUMENT.HTML", and the file is actually named "document.html", the link will still be valid. But if you upload all your pages to a UNIX web server, the link will no longer work. Be sure to check your filenames before uploading.

Pathnames use the standard UNIX syntax. The UNIX syntax for the parent directory (the directory that contains the current directory) is "..".

If you were in the assam.html file and were referring to the original document INDIA.html, your link would look like this:

<A HREF="../INDIA.html">India</A>

In general, you should use relative links whenever possible because:

1) it's easier to move a group of documents to another location (because the relative path names will still be valid)

2) it's more efficient connecting to the server, and

3) there is less to type.

However, use absolute pathnames when linking to documents that are not directly related. For example, consider a group of documents that comprise a user manual. Links within this group should be relative links. Links to other documents (perhaps a reference to related software) should use absolute pathnames instead. This way if you move the user manual to a different directory, none of the links would have to be updated.

9.2.1.2 URLs

The World Wide Web uses Uniform Resource Locators (URLs) to specify the location of files on other servers. A URL includes the type of resource being accessed (e.g., Web, gopher, FTP), the address of the server, and the location of the file. The syntax is:

scheme://host.domain [:port]/path/ filename

where *scheme* is one of the following:

file

a file on your local system

ftp

a file on an anonymous FTP server

http

a file on a World Wide Web server

gopher

a file on a Gopher server

WAIS

a file on a WAIS server

news

a Usenet newsgroup

telnet

a connection to a Telnet-based service

9.2.1.3 Links to Specific Sections

Anchors can also be used to move a reader to a *particular section* in a document (either the same or a different document) rather than to the top, which is the default. This type of an anchor is commonly called a *named anchor* because to create the links, you insert HTML names within the document.

You can also link to a specific section in another document. That information is presented first because understanding that helps you understand linking within the same document.

9.2.1.3.1 Links Between Sections of Different Documents

Suppose you want to set a link from document A (documentA.html) to a specific section in

another document (delhi.html).

Enter the HTML coding for a link to a named anchor:

documentA.html:

In addition to the many institute, Delhi is also home to

Jawaharlal Nehru University.

Think of the characters after the hash (#) mark as a tab within the delhi.html file. This tab tells your browser what should be displayed at the top of the window when the link is activated. In other words, the first line in your browser window should be the Jawaharlal Nehru University heading.

Next, create the *named anchor* (in this example "JNU") in delhi.html:

<H2><A NAME="JNU"> Jawaharlal Nehru University </H2>

With both of these elements in place, you can bring a reader directly to the JNU reference in delhi.html.

NOTE: You cannot make links to specific sections within a different document unless either you have written permission to the coded source of that document or that document already contains in-document named anchors.

9.2.1.3.2 Links to Specific Sections within the Current Document

The technique is the same except that the filename is *omitted.*

For example, to link to the JNU anchor from within delhi.html, enter:

...More information about

<A HREF="#JNU"> Jawaharlal Nehru University

is available elsewhere in this document.

Be sure to include the <A NAME=> tag at the place in your document where you want the link to jump to (<A NAME="JNU"> Jawaharlal Nehru University).

Named anchors are particularly useful when you think readers will print a document in its entirety or when you have a lot of short information you want to place online in one file.

9.2.1.3.3 Mailto

You can make it easy for a reader to send electronic mail to a specific person or mail alias by including the mailto attribute in a hyperlink. The format is:

<A HREF="mailto:*emailinfo@host*">*Name*

For example, enter:

<A HREF="mailto:*pubs@jnu.edu*">*JNU Publications*

to create a mail window that is already configured to open a mail window for the JNU Publications Group alias.

Self Check Exercise

2) Mention the different types of links that can be created in a HTML document.

.....

.....

3) Write the formula for sending a mail to .....

.....

.....

9.3 CASCADING STYLE SHEETS (CSS)

A style sheet is made up of style rules that tell a browser how to present a document. There are various ways of linkihg these style rules to your HTML documents.

Style information may be included in an HTML document in any one of three basic ways:

9.3.1 Linking to an External Style Sheet

An external style sheet may be linked to an HTML document through HTML's **LINK** element:

```
<LINK REL=StyleSheet HREF="style.css" TYPE="text/css"
MEDIA=screen>
<LINK REL=StyleSheet HREF="color-8b.css" TYPE="text/css"
TITLE="8-bit Color Style" MEDIA="screen, print">
<LINK REL="Alternate StyleSheet" HREF="color-24b.css"
TYPE="text/css" TITLE="24-bit Color Style" MEDIA="screen,
print">
<LINK REL=StyleSheet HREF="aural.css" TYPE="text/css"
MEDIA=aural>
```

The **<LINK>** tag is placed in the document **HEAD <http://**

www.htmlhelp.com/reference/html40/head/head.html>. The optional **TYPE** attribute is used to specify a media type—**text/css** for a Cascading Style Sheet—allowing browsers to ignore style sheet types that they do not support. Configuring the server to send **text/css** as the **Content-type** for CSS files is also a good idea.

External style sheets should *not* contain any HTML tags like **<HEAD>** or **<STYLE>**. The style sheet should consist merely of style rules or statements. A file consisting solely of

```
P { margin: 2em }
```

could be used as an external style sheet.

The **<LINK>** tag also takes an optional **MEDIA** attribute, which specifies the medium or media to which the style sheet should be applied. Possible values are

- **screen** (the default value), for presentation on non-paged computer screens;
- **print**, for output to a printer;
- **projection**, for projected presentations;
- **aural**, for speech synthesizers;
- **braille**, for presentation on braille tactile feedback devices;
- **tty**, for character cell displays (using a fixed-pitch font);
- **tv**, for televisions;
- **all**, for all output devices.

9.3.2 Embedding a Style Sheet

A style sheet may be embedded in a document with the **STYLE** element:

```
<STYLE TYPE="text/css" MEDIA=screen>
<!—
 BODY  { background: url(foo.gif) red; color: black }
 P EM  { background: yellow; color: black }
.note { margin-left: 5em; margin-right: 5em }
 —>
</STYLE>
```

The **STYLE** element is placed in the document **HEAD**. The required **TYPE** attribute is used to specify a media type, as it is its function with the **LINK <http://www.htmlhelp.com/reference/css/>** element. Similarly, the **TITLE** and **MEDIA** attributes may also be specified with **STYLE**.

Older browsers, unaware of the **STYLE** element, would normally show its contents as if they were part of the **BODY**, thus making the style sheet visible to the user. To prevent this, the contents of the **STYLE** element should be contained within an SGML comment (**<!— comment —>**), as in the preceding example.

An embedded style sheet should be used when a single document has a unique style. If the same style sheet is used in multiple documents, then an external

style sheet <http://www.htmlhelp.com/reference/css/> would be more appropriate.

9.3.3 Importing a Style Sheet

A style sheet may be *imported* with CSS's **@import** statement. This statement may be used in a CSS file or inside the **STYLE** element:

```
<STYLE TYPE="text/css" MEDIA="screen, projection">
<!—
  @import url(http://www.htmlhelp.com/style.css);
  @import url(/stylesheets/punk.css);
  DT { background: yellow; color: black }
—>
</STYLE>
```

Note that other CSS rules may still be included in the **STYLE** element, but that all **@import** statements must occur at the start of the style sheet. Any rules specified in the style sheet itself override conflicting rules in the imported style sheets. For example, even if one of the imported style sheets contained **DT {background: aqua}**, definition terms would still have a yellow background.

The order in which the style sheets are imported is important in determining how they cascade. In the above example, if the **style.css** imported style sheet specified that **STRONG** elements be shown in red and the **punk.css** style sheet specified that **STRONG** elements be shown in yellow, then the latter rule would win out, and **STRONG** elements would be in yellow.

Imported style sheets are useful for purposes of modularity. For example, a site may separate different style sheets by the selectors used. There may be a simple.css style sheet that gives rules for common elements such as **BODY**, **P**, **H1**, and **H2**. In addition, there may be an extra.css style sheet that gives rules for less common elements such as **CODE**, **BLOCKQUOTE**, and **DFN**. A tables.css style sheet may be used to define rules for table elements. These three style sheets could be included in HTML documents, as needed, with the **@import** statement. The three style sheets could also be combined <http://www.htmlhelp.com/reference/css/> via the **LINK <http://www.htmlhelp.com/reference/css/>** element.

9.4 INTRODUCTION TO DHTML

DHTML is a new and emerging technology that has evolved to meet the increasing demand for eye-catching websites.It's a concept that has been enabled (to different extents in different browsers, of course) by a number of technologies, HTML, Scripting languages including JavaScript, VBScript, the Document Object Model (DOM), layers, and Cascading Style Sheets (CSS).

Dynamic HTML allows a web page to change after it's loaded into the browser —there doesn't have to be any communication with the web server for an update. You can think of it as 'animated' HTML. For example, a piece of text can change from one size or color to another, or a graphic can move from one location to another, in response to some kind of user action, such as clicking a button.

9.4.1 Scripting Languages

Scripts that are executed when a document is loaded may be able to modify the document's contents dynamically. The ability to do so depends on the scripting language itself (e.g., the "document.write" statement in the HTML object model supported by some vendors).

JavaScript is often used to create dynamic HTML documents. One of the more practical things that JavaScript does at this time is increase the aesthetics and friendliness of websites by adding author-specificed user events to static pages. For example, JavaScript allows the page author to embed into a page, statements, which allow user response to certain common events such as when you move the mouse cursor over a link

9.4.2 DOM (Document Object Model)

The Document Object Model sets out to map the Web page for developers. Broken downinto its own components, the DOM details the characteristic properties of each element of a Web page, thereby detailing how we might manipulate these components and, in turn, manipulate the page. The Document Object Model is prescriptive, as it defines what components we may or may not modify, and how, as decided by a panel of developers.

The inventory of components which make up a web page are rather arbitrarily defined. Simply put, one panel of developers may decide to break the Web page down into a certain Set X of components while another panel may prefer to reduce a page into Set Y of components. Perhaps these sets overlap to some degree, but that, too, is arbitrary.

A particular DOM does not necessarily apply to a particular Web page. Web pages are not necessarily compatible, because web browsers are not. Microsoft and Netscape do not share the same DOM between their two browsers and, as a result, the anatomy of web pages differs between them as well.

9.4.3 Layers

In general, "layer" refers to elements that can be positioned at exact coordinates on the page. These elements can be defined with the DIV, SPAN, LAYER, or ILAYER tags. Layers created with DIV and SPAN are referred to as CSS layers because their properties are defined by the Cascading Style Sheets specification by the World Wide web Consortium. (*'WebDeveloper's VirtualLibrary', http://www.wdvl.com/Authoring/DHTML/*) This specification defines style properties (e.g. font, color, padding, margin, word-spacing) in addition to the positioning properties associated with layers (top, left, z-index, visibility). Microsoft Internet Explorer 4.0, allows you to change style properties on the fly with a scripting language such as JavaScript or VB Script. Netscape Navigator 4.0 cannot change style properties after the page has loaded, but positioning properties can be changed with JavaScript.

Both Microsoft and Netscape support CSS layers in their 4.0 browsers. Only Netscape Navigator 4.0 and later, supports layers created with the LAYER and lLAYER tags.

Self Check Exercise

3) Why is DHTML named so? Name the different technologies that comprise DHTML.

.....

.....

.....

.....

9.4.4 Difference between HTML and DHTML

DHTML is a combination of various technologies like scripting languages, CSS, layers, DOM, including HTML. It can be said to be the superset of HTML.Dynamic HTML allows a web page to change after it's loaded into the browser —there doesn't have to be any communication with the web server for an update,whereas HTML pages are static and have to be communicated to the web server if changes are made.

9.5 WEB INTERFACE TO DATABASE LINKING

Servlet technology is one of the hot technologies of the day and Server pages are an extension to it. Introduction of this new technology has hyped the web environment. A lot of its applications are seen with e-commerce. This technology can be very well exploited for the user's convenience in libraries. Servlet was initially developed by Sun Microsystems afterwards Microsoft also brought Active Server Pages (ASP) a more user friendly environment of developing servlets. Later Sun also introduced Java Server Pages (JSP). These act as database connectivity languages, which help in creating Web Interfaces to databases, which convert the user queries into the database query language and retrieve the results and display on the browser.

9.5.1 Introduction to Server Pages

There are two server page technologies presently accepted and widely in use i.e. Java Server Page (JSP) and Active Server Page (ASP), developed by Sun and Microsoft respectively. Server pages are nothing but codes embedded in HTML tags to perform specific routine on certain request. An excellent example is, running a SQL query on request and fetching the results in a particular format or in a customized format.

Let us see two of the most popular server pages today i.e. ASP and JSP.

9.5.2 Overview of ASP

Developed by Microsoft, ASP is one of the most powerful and market catching technology. It runs on the *Internet Information Server (IIS)*, which comes with *Windows 2000* and *Windows NT 4.0*. For *Windows NT 4.0* one requires to load *Windows NT Option Pack drivers* downloadable free from Microsoft site. ASP applications can be developed on Windows 9x and on *Windows Me* using *Microsoft's Personnel Web Server (PWS)* (1). Efforts have been made to use ASP on *Linux* also. *Sun Chilli!Soft ASP* provides full ASP support for *Apache*,

Lotus, and also *Microsoft*, running on *HP-UX*, *Linux*, *Sun Solaris* and *IBM AIX*. (2)

9.5.2.1 ASP at Work

An ASP file normally contains HTML tags, just as a standard HTML file. In addition, an ASP file can contain server scripts, surrounded by the delimiters <% and %>. Server scripts are executed on the server, and can contain any expressions, statements, procedures, or operators that are valid for the scripting language. One can use any scripting language VBScript, JScript, PERL or PYTHON. VBScript is default script language for ASP while *JScript* needs to be defined in program. To use *PYTHON* or *PERL* one needs to load the scripting engines. (3)

There are Objects defined in ASP to perform different tasks. For example, to write there is an object called as '*Response*' which further uses a function '*write*' within the class '*Response*'.

For example, to write name of a person, code should be as follows and the file name as <filename>.*asp* extension,

```
<html>
<body>
<%
response.write("S.R.Ranganathan")
%>
</body>
</html>
```

This code uses VBScript and finally writes on the browser S. R. Ranganathan. If one is using JScript he has to define at the beginning of the document the Scripting language he is using. For example:

```
<%@ language="Jscript" %>
```

9.5.3 Overview of JSP

JSP is developed by Sun Microsystem. It is based on Java technology. This technology was developed more in anti-Microsoft flux which can be seen in development of JSP, like integrating JSP with *Java classes* so that one can write java codes in JSP page and run. JSP promises to run on any platform. There are a number of servers which supports JSP. Sun offers the Java Web Server to run JSP pages. But still there are servers available free either as standalone or add ons. *Tomcat* which is developed under *Jakarta project*, is the server which can be used as standalone as well as add on for the Apache web server, only a slight modification is required in Tomcat to connect to Apache.

9.5.3.1 JSP at Work

Definitely JSP is developed under a competition between Microsoft and Sun Microsystem, but there are many features that put JSP on top of ASP. The biggest advantage with JSP is use of Java as scripting language. It also has the feature of Tag library where one can define own tags and use it like XML (Extensible Markup Language) (5). Another important feature with JSP is use of Java beans and an even more advanced version is EJB (Enterprise JavaBeans).

Definitely JSP is a bit cryptic compared to ASP and that is why often it is claimed that ASP is nonprogrammer's language and JSP is programmers language. JSP by default uses JavaScript. But one can also use PHP.

```
<%@ page language="java" %>
<html>
<body>
<%
  out.print("S.R. Ranganathan");
%>
</body>
</html>
```

JSP is object-oriented which means once an object is defined it can be called many times. To call a class JSP provides a method called page property named '*Import*'. One can call any number of classes in one page. For example,

```
<%@ page import="java.sql.*" %>
```

But use of many classes reduce the loading speed of webpagse in browser.Most of the work can be done without defining Tags or using Beans.

Self Check Exercise

4) What are server pages? Name different server pages available.

.....

.....

.....

.....

9.6 INTRODUCTION TO XML

"If you use the original World Wide Web program, you never see a URL or have to deal with HTML. You're presented with the raw information. You then input more information. So you are linking information to information—like using a word processor. That was a surprise to me—that people were prepared to painstakingly write HTML."

—Tim Berners-Lee

Tim Berners-Lee had vision about the easy access to the data all round the world, when he first proposed the WWW. Since than Internet has gone a long.It started from text based browser to now completely GUI (Graphical User Interface) browser. The representation of information has also changed. We have seen the development of HTML in 1980s and then its various versions in 1990s, then came the era of XML (eXtensible Markup Language). This change also supports the vision of Lee though it was difficult for him also, to believe about the development of such a system for the web content development. But finally it was he who also started visioning the use of semantic Internet as finding information had become as difficult as finding a gold coin in garbage. That is why the domain specific information interchange systems were thought of and hence XML also came into the picture.

9.7 NEED FOR XML

The idea of markup was to format a particular kind of document. The markup languages that carry the instructions for text processing are known as *Procedural markup*. But later on, it was felt that for system to system information interchange, markup languages can be used. This was first realized by Charles Goldfarb, Ed Mosher and Ray Lorie when they were working with legal documents. They designed first markup language known as GML (Generalized Markup Language) based on the following observations:

- The document processing programs needed to support a common document format.
- The common formats needed to be specific to their domain-for example legal documents.
- To achieve a high degree of reliability, the document format would have to follow specific rules.

For instance, take an example of memorandum,

From: Akkamahadevi
To: Suchitra Pattanayak
CC: Prasenjit Kar
Date: 27.01.2002
Subject: Appointment order

We are extremely happy to inform you that you are selected as the coordinator of Knowledge management team.

If we look into this document we find that there are six fields in this document.

- Who sent the document (the From: field),
- Who the document is intended for (the To: field),
- Who has been sent a copy of document (the CC: field),
- The date of document written (the Date: field,
- The subject of document (the Subject: field),and
- The document body.

So, if we make a fixed structure of this document then whoever writes the document has to follow the same structure. Thus porting information from one system to another it will not be a problem as the structure of document is well defined. The definition of the structure of document is known as DTD (Document Type Definition).

Glodfarb further fine-tuned GML and proposed the SGML (Standardized General Markup Language) which was further approved by ISO (International Organization for Standardization) in 1986. This language was not a language itself but it was a meta language to develop other markup languages. HTML

(HyperText Markup Language) is a derivative of SGML. HTML acts more like a formatting language so it is always difficult to pull out the kind of data is stored inside a HTML document. Once this difficulty was understood, for information interchange the need for domain specific tags was felt. Development of such tags was not possible with HTML. Hence, XML was developed. It is always said that XML is more near than HTML to SGML.

9.7.1 Semantic Tags

XML was designed to attach semantic to data i.e. adding context to the data. It does so by allowing to define your own tags. For example,

```
<?xml version="1.0" encoding="UTF-8" ?>
- <book>
  <title>Prolegomena to library classification</title>
- <author>
  <f_name>Ranganathan</f_name>
  <l_name>S.R.</l_name>
  </author>
  <edition>3rd reprint</edition>
  <place>Bangalore</place>
  <publisher>Sarada Ranganathan Endowment</publisher>
  <physical_desc>640 p.</physical_desc>
  </book>
```

The example shows the structure of a document, which describes a book, titled *Prolegomena to library classification.* The book has a title, author, edition, place, publisher, physical description elements. Author is further divided into first name (f_name) and last name (l_name). Inside these tags the actual data is stored. These tags provide context to the whole structure of the document, hence these are known as semantic tags.

Self Check Exercise

5) What are semantic tags?

.....

.....

.....

.....

9.8 XML DOCUMENT DESIGN

9.8.1 DTD (Document Type Definition)

One can define a structure of XML document and give it to others to write the XML document against his own schema to avoid the mistakes. A schema is nothing but the logical structure of document. This schema is called as DTD (Document Type Definition). When the XML document is prepared against DTD it is called a *Valid document* and when there is no DTD for the document and the syntax of document is correct it is known as *Well-formed* document.

A DTD can be defined for a Valid-document. The declaration of DTD used for the validation is given in the processing tag of XML file. Further extension of *book.xml* as a catalog for the description of book is as follows:

```
1—<?xml version="1.0" encoding="UTF-8"?>
2—<!DOCTYPE catalog SYSTEM "C:\hydra\book.dtd">
3—<catalog>
4—-     <book>
5—-         <title>Mastering XML</title>
6—-         <author authorship="primary">
7—-             <f_name>Ann</f_name>
8—-             <l_name>Navaroo</l_name>
9—-         </author>
10—         <author authorship="secondary">
11—             <f_name>Chuck</f_name>
12—             <l_name>White</l_name>
13—         </author>
14—         <author authorship="secondary">
15—             <f_name>Linda</f_name>
16—             <l_name>Burman</l_name>
17—         </author>
18—         <edition>First edition</edition>
19—         <place>New Delhi</place>
20—         <publisher>BPB</publisher>
21—         <physical_desc>xxxiv, 882p.</physical_desc>
22—     </book>
23—</catalog>
```

In this example, the second line,

```
<!DOCTYPE catalog SYSTEM "C:\hydra\book.dtd">
```

indicates that this XML file should use *book.dtd* for the validation of document. The absolute path of the DTD file is mentioned, even the relative path can be mentioned as the case may be. The tag starting from <! > contains processing instruction for parser. Comments are always given inside <!— —>.

For example: <!—Your comments —>

The listing of *book.dtd* is as follows:

```
1—<?xml version="1.0" encoding="UTF-8"?>
2—<!ELEMENT catalog (book+)>
3—<!ELEMENT book (title, author+, edition, place, publisher,
physical_desc)>
```

4—<!ELEMENT title (#PCDATA)>

5—<!ELEMENT author (f_name, l_name?)>

6—<!ELEMENT edition (#PCDATA)>

7—<!ELEMENT place (#PCDATA)>

8—<!ELEMENT publisher (#PCDATA)>

9—<!ELEMENT physical_desc (#PCDATA)>

10—<!ELEMENT f_name (#PCDATA)>

11—<!ELEMENT l_name (#PCDATA)>

The first line shows the version of XML specification used for the generation of file. The second line means the element *catalog* will have atleast one or more occurrence of element *book*. That means the element book is mandatory and repeatable field in the terms of CDS/ISIS language. The sign plus (+) represents the field in question is mandatory and repeatable. There are other notations for different kind of repeatability.

No sign	Element is mandatory and non-repeatable
Plus (+)	Element is mandatory and repeatable
Star (*)	Element is optional and repeatable
Question mark (?)	Element is optional and non-repeatable

The third line means the element *book* has the child elements title, author (which is repeatable and mandatory), place, publisher and physical_desc (physical description). The element which contains data is known as *Leaf element*. A *Leaf element* is represented by either #PCDATA, i.e. parsed character data (the data will be parsed by XML parser) or #CDATA i.e. character data (XML parser will not parse the data). Parsed character data or Character data are nothing but the actual data content of the element. For example, is we look at the corresponding xml file to this DTD,

<title>Mastering XML</title>

the data "*Mastering XML*" is parsed character data according to our DTD. The use #CDATA is to represent the formulae or data which contains restricted characters for example, angular brackets (<) and so on. So in our example, all the leaf elements are represented as #PCDATA. One can use DTD in the same XML file but it is always good to use a separate DTD file for your XML file.

Once you define DTD any non-conformity with the structure during the generation of XML document will be complained at the time of validation.

Self Check Exercise

6) What is DTD?

.....

.....

.....

.....

9.8.2 XSL (eXtensible Stylesheet Language)

We know that the objective of XML document is to store data, so when we see a XML file in a browser it appears as Fig. 1.

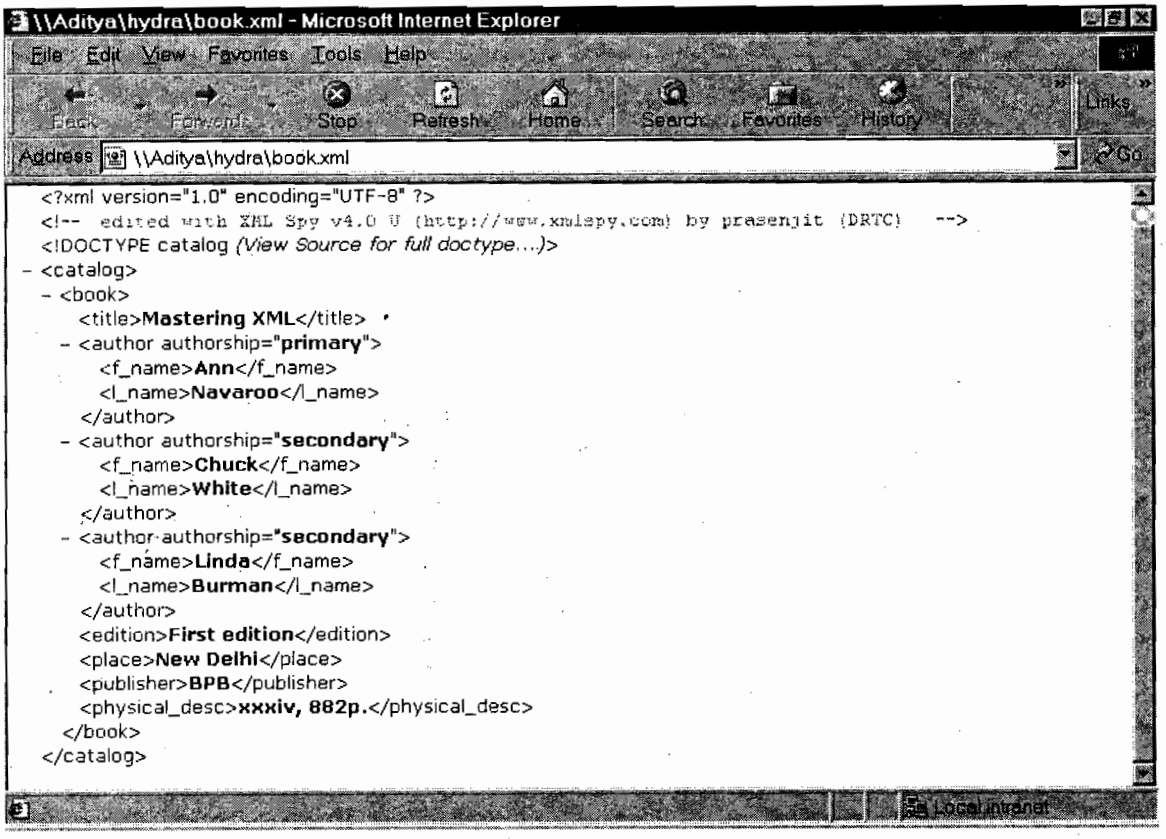


Fig. 1: Displaying plain XML file in browser

To see a formatted file in the browser we need to process the XML file and then display. There are two ways of formatting a file for display, one is use of CSS (Cascading Style Sheet) and second is XSL (eXtensible Stylesheet Language). XSL is far more sophisticated than CSS. One way to use XSL is to transform XML into HTML before it is displayed by the browser. XSL is in itself a huge language and its specification is brought by W3C.

Let us take new example, for formatted display in browser. Let us look at the content of book2.xml,

```
1—<?xml version="1.0" encoding="UTF-8"?>
2—<!— edited with XML Spy v4.0 U (http://www.xmlspy.com) by prasenjit
(DRTC) —>
3—<!DOCTYPE catalog SYSTEM "book2.dtd">
4—<?xml-stylesheet type="text/xsl" href="book2.xsl"?>
5—<catalog>
6—- <book>
7—-      <image>proleg.gif</image>
8—-      <alt>Prolegomena to library classification</alt>
9—-      <link>http://www.isibang.ac.in/drtc/srr/index.htm</link>
10—        <title>Prolegomena to library classification</title>
```

```
11—        <author authorship="primary">
12—            <f_name>Ranganathan</f_name>
13—            <l_name>S.R.</l_name>
14—        </author>
15—        <edition>3rd Reprint</edition>
16—        <place>Bangalore</place>
17—        <publisher>Sarada Ranganathan Endowment</publisher>
18—        <physical_desc>640p.</physical_desc>
19—    </book>
20—    <book>
21—        <image>xml.gif</image>
22—        <alt>Mastering XML</alt>
23—        <link>http://www.ibiblio.org/xml/books/bible2/chapters/
ch17.html</link>
24—        <title>Mastering XML</title>
25—        <author authorship="primary">
26—            <f_name>Ann</f_name>
27—            <l_name>Navaroo</l_name>
28—        </author>
29—        <edition>First edition</edition>
30—        <place>New Delhi</place>
31—        <publisher>BPB</publisher>
32—        <physical_desc>xxxiv, 882p.</physical_desc>
33—    </book>
34—</catalog>
```

Following is the XSL file *(book2.xsl)* for the display of *book2.xml*. This XSL file transforms *book.xml* to a tabular form and represents it as shown in Fig 2.

```
1—<?xml version="1.0" encoding="UTF-8"?>
2—<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/
Transform" xmlns:fo="http://www.w3.org/1999/XSL/Format">
3—<xsl:template match="/">
4—<html>
5—<xsl:apply-templates/>
6—</html>
7—</xsl:template>
8—<xsl:template match="catalog">
9—<body>
10—<xsl:apply-templates select="book"/>
11—</body>
12—</xsl:template>
13—<xsl:template match="book">
```

```
14—<img src="{./image}" alt="{./alt}" height="125" width="100"></img>
15—<table >
16—<tr>
17—<td><font color="red">Author</font></td>
18—<td><a href="{./link}"><xsl:value-of select ="author/f_name"/
>&#32;<xsl:value-of select="author/l_name"/>
19—</a></td>
20—</tr>
21—<tr>
22—<td><font color="red">Title</font></td>
23—<td>
24—<xsl:value-of select="title"/>
25—</td>
26—</tr>
27—<tr>
28—<td><font color="red">Edition</font></td>
29—<td><xsl:value-of select="edition"/></td>
30—</tr>
31—<tr>
32—<td><font color="red">Publisher</font></td>
33—<td>
34—<xsl:value-of select="publisher"/>
35—</td></tr>
36—<tr>
37—<td><font color="red">Place</font></td>
38—<td>
39—<xsl:value-of select="place"/>
40—</td>
41—</tr>
42—<tr>
43—<td><font color="red">Physical Description</font></td>
44—<td>
45—<xsl:value-of select="physical_desc"/>
46—</td>
47—</tr>
48—</table>
49—<br></br>
50—<br></br>
51—</xsl:template>
52—</xsl:stylesheet>
```

The first line is processing instruction to identify an XML document and its version of it. The second line is the instruction for the parser to use the schema for XML transformation. The third line instructs the parser to match with the root (/) node and when root node is found it writes <html> to the virtual output file. The line eight shows that the parser now matches with node called "*catalog*" then writes <body> inside the <html> tag in the virtual output file. It further selects the node called "*book*". It has to be kept in mind that each

```
<xsl:template match="">
```

has to be closed by

```
</xsl:template>
```

9.8.2.1 Putting Image

The line 13 matches with <book> node. Then it puts

```
<img src="{./image}" alt="{./alt}" height="125" width="100"></img>
```

where the source of image (src) is defined by defining the path of <image> in the XML document. The relative path should be given for finding the data. Then further attributes like height and width are defined. Alternative caption is stored in <alt> of book2.xml file. XSL extracts the value of <alt> from the XML file.

9.8.2.2 Creation of Link

Line 18 represents how the links to other document can be given in XML document. To link the pages anchor <a> tag is used. Similar to image tag here also link file is mentioned in XML document <link> tag. *./link* represents the relative path of the node <link>.

```
<a href="{./link}"><xsl:value-of select ="author/f_name"/>&#32;<xsl:value-of select="author/l_name"/>
```

Line 19 onwards each node is selected from *book2.xml* and formatting is applied to it.

Basically the output is generated in a tabular form.

Self Check Exercise

7) What is XSL? How it is implemented?

.....

.....

.....

.....

9.9 WEB SERVERS

A Web server is a program that, using the client/server model and the World Wide Web's Hypertext Transfer Protocol (HTTP), serves the files that form

Web pages to Web users (whose computers contain HTTP clients that forward their requests). Every computer on the Internet that contains a Web site must have a Web server program. Two leading Web servers are Apache, the most widely-installed Web server, and Microsoft's Internet Information Server (IIS). Other Web servers include Novell's Web Server for users of its NetWare operating system and IBM's family of Lotus Domino servers, primarily for IBM's OS/390 and AS/400 customers.

Web servers often come as part of a larger package of Internet- and intranet-related programs for serving e-mail, downloading requests for File Transfer Protocol (FTP) files, and building and publishing Web pages. Considerations in choosing a Web server include how well it works with the operating system and other servers, its ability to handle server-side programming, security characteristics, and publishing, search engine, and site building tools that may come with it.

Self Check Exercise

8) What is a Web Server?

.....

.....

.....

.....

9.9.1 Website Hosting

Every webpage, email, file, or online service is stored ("hosted") on a computer (called a "server") that is connected to the Internet. Putting the resource on a server to provide access to it on Web is called "Website Hosting".

Years ago, when Web servers were first prototyped, they served simple HTML documents and images. Today, they are frequently used for many other applications. Most Internet users believe a Web site's success or failure is due to its content and functionality rather than the server used to power it. However, the choice of the correct server, and understanding its capabilities and limitations is an important step on the road to success.

Lets see what a Web server does? It serves static content to a Web browser at a basic level. This means that the Web server receives a request for a Web page such as

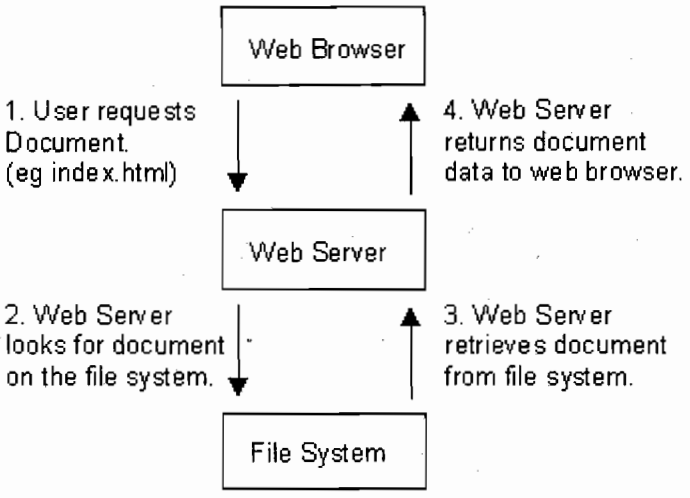
http://www.Webcompare.com/index.html

and maps that Uniform Resource Locator (URL) to a local file on the host server.

In this case, the file

index.html

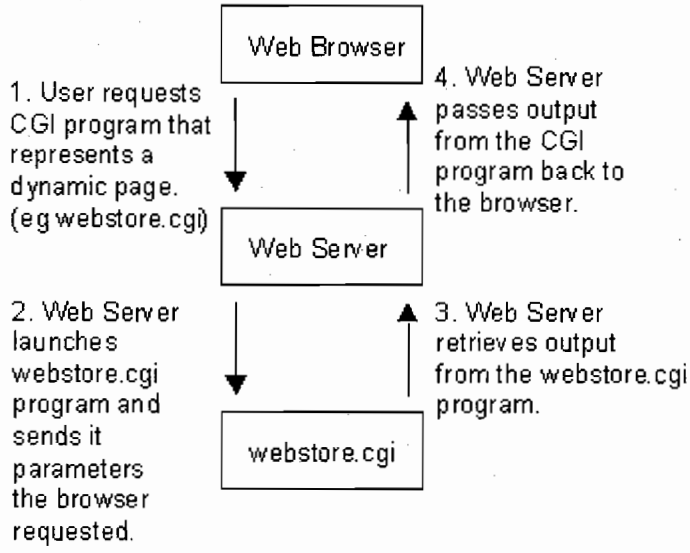
is somewhere on the host file system. The server then loads this file from disk and serves it out across the network to the user's Web browser. This entire exchange is mediated by the browser and server talking to each other using Hypertext Transfer Protocol (HTTP). This workflow is shown in the figure below.



This simple arrangement, which allows the serving of static content such as HyperText Markup Language (HTML) and image files to a Web browser was the initial concept behind what we now call the World Wide Web. The beauty of its simplicity is that it has led to much more complex information exchanges being possible between browsers and Web servers.

The most important expansion on this was the concept of dynamic content (i.e., Web pages created in response to a user's input, whether directly or indirectly). The oldest and most used standard for doing this is Common Gateway Interface (CGI). It basically defines how a Web server should run programs locally and transmit their output through the Web server to the user's Web browser that is requesting the dynamic content.

For all purposes the user's Web browser never really has to know that the content is dynamic because CGI is basically a Web server extension protocol. The figure below shows what happens when a browser requests a page dynamically generated from a CGI program.



The second important advance, and the one that makes e-commerce possible, was the introduction of HyperText Transmission Protocol, Secure (HTTPS). This protocol allows secure communication to go on between the browser and Web server. In short, this means that it is safe for user and server to transmit sensitive data to each another across what might be considered an insecure network.

9.10 TOOLS FOR WEB PAGE DESIGNING

There are many software used for web page designing, like Front page, Dreamweaver, Coffee cup HTML editor, etc. Each software has its own advantages and disadvantages. The most popular ones are Dreamweaver and Front page.

9.10.1 Dreamweaver

This section shows you how to use Dreamweaver to create and edit Web documents.

The different steps involved in designing a Web page are:

- Create a document
- Tables
- Attach a behavior to an image
- Import a Microsoft Word in HTML document
- Format text using HTML styles
- Create links
- Apply a template
- Create a jump menu

9.10.1.1 Create a Document

Create a home page for a site. As you build this document, you'll add a page title, layers, images, text, and links; desirably all the pages will contain the same design components. The Object palette (choose Window > Objects), is used to add objects to the document. The Property inspector (choose Window > Properties), is used to set properties or attributes for objects in the document

9.10.1.2 Save Your Document

Save the blank document created when you launched Dreamweaver. It is better to save all the files in one folder and the index.html or index.htm file in a separate folder to avoid cluttering of files.

9.10.1.3 Define the Title of the Page

Defining a page title for an HTML document helps users identify and keep track of a page that they're browsing. The page title appears in the title bar of the browser when a page is viewed in it. When a page is bookmarked, the page title appears in the bookmark list. If a document is created without a page title, the document appears in the browser with the title 'Untitled Document'.

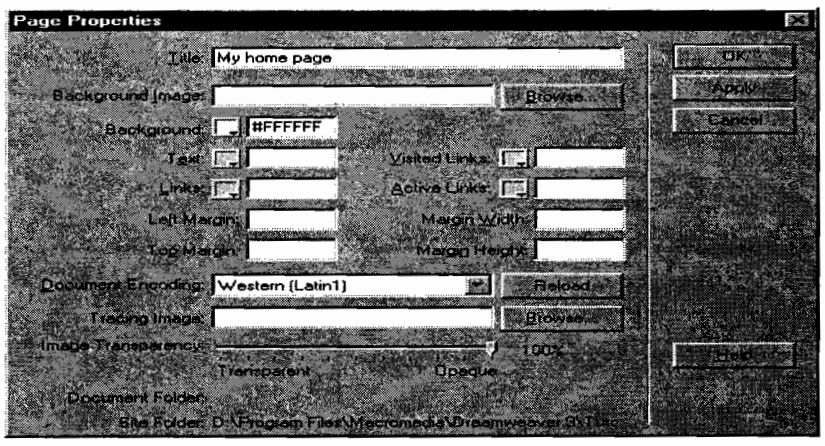


Fig. 2: Page Properties box

In the Page properties the text color, background color, hyperlinked color, active link and visited links color also can be set as user wants.

9.10.1.4 Add an Image

Adding an image in Dreamweaver is a very easy task. The cursor is placed on the screen where the image is wanted and the image can be selected by using the Insert Image option. Mouse-over images can be inserted with the Rollover option.

9.10.1.5 Formatting Text

You can format text in the Document window by setting properties in the Property inspector. First, select the text you want to format, and then apply the changes. You can change the font, color and size of the text.

Before you format the text, create a couple of paragraphs. Add a paragraph return after the first sentence and after the second sentence.

- If the Property inspector isn't open, choose Window > Properties.
- With the insertion point anywhere in the layer, press Control+A (Windows) to select all the text in the layer.
- In the Property inspector's second Format pop-up menu, which currently reads Default Font, select Arial, Helvetica, sans-serif.
- In the Size pop-up menu, select 3.
- The text in your document automatically updates to reflect the changes.

9.10.1.6 Importing a Word Document to a HTML Document

Import this document into Dreamweaver, and then clean up any nonstandard HTML code in the document using a new Dreamweaver feature called Clean Up Word HTML. The Clean Up Word HTML feature provides options for cleaning up or fixing HTML tags, defining CSS (Cascading Style Sheets) elements, setting a page background color, removing Word-specific markup tags, and converting Word font sizes and headings to HTML size attributes.

9.10.1.7 Making Font Style

There are two methods of changing the font style:

Method 1: Select the text, right click and select from Properties

Method 2: Select the text, and choose from the Text menu.

To change the font style,

we can also prepare the ordered list, unordered list, paragraph or general text, text size etc through the Text menu.

9.10.1.8 Creating Hyperlinks

Hyperlinking provides a link from one hypertext to another or text to image or

image to text or image to image. It can be within the same document or links to other documents. Hyperlinks lead to related information or other pages on within that document itself. Hyperlinks can be created by using the Link option of the Properties menu.

By default, the hyperlink opens the linked page in the same window. If you want the hyperlink to open the page in a separate window, then Select the Target option.

9.11 SUMMARY

In today's Internet-savvy world, the presentation and display of websites is of importance. Libraries can develop and design their own Web pages and host them on the Internet to gain visibility as well as give access to their local OPAC. Technologies like HTML, DHTML, and XML help in creating and designing Web pages. Especially XML, is gathering lot of interest in the library community, as it allows for semantic tags. It is in the interest of the librarians to keep abreast of emerging technologies, so that they can provide the user better services. Advanced tools for web page designing are now available and they are made very user friendly so that one does not need to worry about memorizing the syntax for all tags. It is much simpler with these tools also to incorporate various multimedia elements into the resources. But care should be taken not to over load the network capacity by adding unnecessary elements, which make the file size large.

9.12 KEYWORDS

Browser : User's software program for viewing & browsing information on the Internet.

DTD : Document Type Definition—this is the formal specification of a markup language, written using SGML

Element : An *element* is a fundamental component of the structure of a text document. Some examples of elements are heads, tables, paragraphs, and lists. Elements can contain plain text, other elements, or both.

HTML : HyperText Markup Language—HTML is an SGML DTD. In practical terms, HTML is a collection of platform-independent styles (indicated by markup tags) that define the various components of a World Wide Web document. Tim Berners-Lee while at CERN, the European Laboratory for Particle Physics in Geneva, invented HTML.

Plug-In : This is a program that your browser uses to manipulate a downloaded file. It differs from a Helper Application in that the plug-in works inside the browser window.

SGML : Standard Generalized Markup Language—a standard for describing markup languages. A mark-up language alters the presentation of the contents of a document. A generalized mark-up language can deal with many different document types, and a standard generalized mark-up language deals with all these documents in a standardized way.

Server : This is a mainframe computer that serves the other computers attached to it.

9.13 ANSWERS TO SELF CHECK EXERCISES

1) Hypertext Markup Language (HTML) is a structured markup language that is used to create Web pages. A markup language such as HTML is simply a collection of codes called elements that are used to indicate the structure and format of a document. Elements in HTML consist of alphanumeric tokens within angle brackets, such as <b>, <html>, <body>, etc.

2) The different types of links that can be made in a HTML document are: links to URLs; Links between sections of different documents; Links to specific sections within the current document; links to emails.

3) Dynamic HTML (DHTML) is named so because even after it is loaded on the browser it can be changed, without having to send a request to the browser. The different technologies that make up DHTML are DOM (Document Object Model), Layers, CSS (Cascading Style Sheets), and HTML.

4) Server pages act as database connectivity languages, which help in creating Web Interfaces to databases, which convert the user queries into the database query language and retrieve the results and display on the browser. Some of the available server pages, today, are ASP (Active Server Pages) from Microsoft, JSP (Java Server Pages) from Sun Microsystems Inc. and PHP (Hypertext Preprocessor).

5) XML was designed to attach semantic to data i.e. adding context to the data. It does so by allowing to define your own tags. For example,

```
<?xml version="1.0" encoding="UTF-8" ?>
- <book>
<title>Prolegomena to library classification</title>
- <author>
<f_name>Ranganathan</f_name>
<l_name>S.R.</l_name>
</author>
```

```
<edition>3rd reprint</edition>
<place>Bangalore</place>
<publisher>Sarada Ranganathan Endowment</publisher>
<physical_desc>640 p.</physical_desc>
</book>
```

The example shows the structure of a document, which describes a book, titled *Prolegomena to library classification.* The book has a title, author, edition, place, publisher, physical description elements. Author is further divided into first name (f_name) and last name (l_name). Inside these tags the actual data is stored. These tags provide context to the whole structure of the document, hence these are known as semantic tags.

6) One can define his own structure of XML document and give others to write the XML document against his own schema to avoid the mistakes. A schema is nothing but the logical structure of document. This schema is called as DTD (Document Type Definition).

7) There are two ways of formatting a file for display, one is use of CSS (Cascading Style Sheet) and second is XSL (eXtensible Stylesheet Language). XSL is far more sophisticated than CSS. One way to use XSL is to transform XML into HTML before it is displayed by the browser. XSL is in itself a huge language and its specification is brought by W3C.

8) A Web server is a program that, using the client/server model and the World Wide Web's Hypertext Transfer Protocol (HTTP), serves the files that form Web pages to Web users (whose computers contain HTTP clients that forward their requests).

9.14 REFERENCES AND FURTHER READING

ASP A-Z.

http://msdn.microsoft.com/library/default.asp?URL=/library/en-us/dnasp/html/aspatoz.asp

Serving Up Web Server Basics By Chris Hughes and Gunther Birznieks *http://webcompare.internet.com/webbasics/index.html*

Sun ChilliSoft ASP homepage: products. *http://www.chilisoft.com/products/default.asp*

Introduction to JSP. *http://www.w3schools.com/asp/asp_intro.asp*

JavaServer Pages. *http://java.sun.com/products/jsp/keyfeatures.html*

A Beginner's Guide to HTML.

http://archive.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimerAll.html

Beginner's guide to DHTML.

http://www.wdvl.com/Authoring/Tutorials/dhtml.html

Comparing JavaServer PagesTM and Microsoft ActiveServer PagesTM. *http://java.sun.com/products/jsp/jsp-asp.html*

Linking Style Sheets to HTML. *http://www.htmlhelp.com/reference/css/style-html.html*

MARCHAL (Benoit). XML by example, Prentice-Hall : New Delhi, 2000.

NAVARRO (Ann), WHITE (Chuck) and BURMAN (Linda). Mastering XML. BPB : New Delhi, 2000.

NewML. *http://www.newsml.org*

POWELL (Thomas A.). The complete reference HTML, 2nd ed. Tata McGraw Hill : New Delhi, 2000.

The Jakarta Project-Jakarta Tomcat. *http://jakarta.apache.org/tomcat/*

The XML Bible, Second Edition: XSL Transformations. (Chapter 17) *http://www.ibiblio.org/xml/books/bible2/chapters/ch17.html*

What is Dynamic HTML? by Nadav Savio *2 Oct 1997*
http://hotwired.lycos.com/webmonkey/geektalk/97/39/index3a.html?tw=authoring