Course Introduction

A web site can be defined as set of web pages that has been designed for specific organizational purpose. A web site consists of many web pages and can be hosted on one or more web servers, with a single web domain name. In general, the default web page of the web site is called an index page and can be accessed using the Uniform Resource Locator (URL). For example, you access IGNOU web site can be accessed using the URL- http://www.ignou.ac.in / which displays the index page of the IGNOU web site in your browser. From this page then you can access many other links provided by IGNOU's web site.Thus, a web site involves two aspects – client and the server. The client of a web site is a browser program which accesses the web site hosted on a web server through the Internet. The access of a web page by a client from a server, in addition to TCP/IP protocol which is the base protocol for Internet, would require additional protocol. The protocol for web page access involving transfer of a web page from web server to client browser is called HyperText Transfer Protocol (HTTP). In case, you are accessing a secure web page of a web site such as your bank account, then a secure HTTP called HTTPS is used. This protocol uses encryption of data to ensure security of data transfer. For more details on these concepts, you may go through the course BCS-052.

Web programming is the art of creating web pages of a web site. Web programming has two aspects – client side programming and server side programming. A client side program may be written in Markup languages like HTML. The display of these markup languages is controlled by a style sheet called Cascading Style Sheets (CSS). In addition, a browser can also run simple programs that may be written in a scripting language like JavaScript. Basic introduction to HTML and JavaScript has already been given to you in MCS016: Internet Concepts and Web Design Course.

The first Block of this course discusses the Client Side languages. This Block introduces the concept of Web 2.0 and HTML5. Web 2.0 has brought dynamisms to web site such that you are able to interact with the web sites or other users of the web sites. CSS are a flexible way of changing the look and feel of a web site. A good web site must use CSS. This block explains how you can create and use CSS. XML has become a standard for information interchange. It also has been discussed in this Block. You can make your website dynamic, if you are able to access its component and process them. Such processing would require interaction of the user in the form of events. This block explains the Document Object Model (DOM) that defines the component hierarchy and Events. You can use DOM, Events and JavaScript commands to create dynamic client pages. This block also covers aspects of Wireless markup language.

The second block of this course discusses the server side programming. When you access a web page, the browser displays the HTML code of that page under the control of CSS. The web page may also contain other scripts like JavaScript. These pagesare transferred from the server to the browser on your request. Are these web pages stored at the web server as same HTML, CSS, XML, JavaScript etc.pages or they are generated with the help of a programming language? Web sites on the server side use scripting languages that are used to create pages for clients as per their requirements. The languages that are used to create client pages at the server, based on clients requirements are called Server Side Scripting Languages. Some of the common server side scripting languages are PHP, JSP, SERVLET, ASP.NET, PYTHON and many more. In the second Block, we have explained the use of JSP as the server side scripting language.

Block Introduction

A web site is displayed on a web browser. Every browser provides an option by which you can view the page source of the web page that is being displayed. Look into it closely, you will find that the web page you have opened may consist of HTML, JavaScript and other code. This Block is an introduction about the code that is displayed on the client browser. In case you want to make good website then you should learn about these concepts in details. Please note that by just reading through the block you will not be able to learn. You must perform the practical as suggested in BCSL057:Web Programming Lab, in order to take best advantage of these Blocks.

The Unit 1 explains the Web 2.0 technology. It provides information about some of the newer products that have developed over the years. Some of these include applications like Rich Internet Applications, widgets, mashups, social networking, web services, blogs etc. The Unit also describes features of HTML 5. It also gives an example of use of different tags of HTML 5.

Unit 2 is devoted to Cascading Style Sheet (CSS). Style sheets are very important for creating flexible, consistent web pages along a web site. The Unit provides details on differnt ways of attaching style sheet to a web page. It also gives list of different kind of properties of styles. In addition to these, this Unit introduces the concept of box model. It also discusses two important tags and which are very useful in logical division of the content of a web page on a single display window. This Unit also demonstrates the use of style sheet with the help of an example.

Unit 3 provides a basic introduction to XML. XML has become a standard data format for exchange of data over the net. it allows you to create your own tags. Since you can create your own custom tags in XML, therefore, they need to be validated. You can define the XML validations using Document Type Declarations (DTD) or XML Schema. This Unit introduces both these mechanism. The Unit also discusses the ways of displaying XML documents.

Unit 4 introduces you to the concept of Document Object Model (DOM) which defines the structure of a HTML document. The elements of the document then can be consistently accessed and changed using event driven programming of JavaScript. This mechanism thus helps in updating specific portion of a document of a web page that is displayed in the browser.

Unit 5 of this Block introduces the concept of Wireless Markup Language (WML). WML are used for creating web pages for display on wireless devices. Although utility of such applications has reduces with the advent of high end smart devices, still many older web pages uses these concepts.

UNIT1 WEB 2.0 AND XHTML/HTML5

Structure

1.0 Introduction

Since the advent of World Wide Web (WWW) in 1990, the Internet and its uses are growing rapidly. The Web Programming technologies have changed in the past substantially. You have already been introduced to the process of designing web pages in the course MCSL-016. The collection of web pages is website. The way of writing a web page is web programming. Web development refers to building, creating, and maintaining websites. It includes aspects such as web design, web publishing, web programming, and database management. A web developer may use web programming languages viz., HTML, JavaScript, XML, ASP, JSP, PHP and maintains a database used by dynamic websites. The initial web was focussed on information. There was no or little need for user interaction. With the advent of Web 2.0, communication has changed. There are more than two billion Internet users worldwide exchanging ideas through social networking sites or using chat or any other two way or group communication software. Blogs, Wikis and Web Services are also components of Web 2.0. In this unit, you will learn about the concepts and characteristics of Web 2.0. This unit also explains the use of HTML5 tags for creating static web pages.

1.1 Objectives

After going through this unit, you should be able to:

- state the concepts applicable to Web 2.0

- define the technologies used in Web 2.0
- explain about the XHTML/HTML5
- differentiate between XHTML and HTML
- use XHTML/HTML5 tags to create simple static web pages
- add <audio>, <video> tags of HTML5

1.2 Web 2.0: An Introduction

Web 2.0 is a second generation of the World Wide Web. Web 2.0 refers to the transition from static web pages to dynamic web. Web 2.0 is more organized and is based on creating web applications to the specific needs of the users. Web 2.0 has improved communication among the users, with an emphasis on Web-based communities of users and more open sharing of information. Web 2.0 concepts highlight services that allow user to find and manipulate contents.

How is Web 2.0 different?

In early days of Web (Web 1.0), there was no or little need for different web applications to communicate and share data with each other. It was concerned with only creating and viewing online contents. However, web 2.0 has changed this scenario. Web 2.0 offers services that allow user to find and manipulate contents, coupled with all types of media services. Examples of web 2.0 are social networking sites, blogs, video sharing, wikis, web application, mashups etc. The basic idea of each of these websites is user interaction. On blogs, you can post comments; on social networking site, you can make a friend; on social news, you can vote for article and on wikis, you can create, edit and share information. The concept of web 2.0 is based on participation of user. This design encourages user interaction and community contribution.

Web 2.0 provides user with more interaction, better communication and storage facilities through their browsers. The client-side technologies used in Web 2.0 development include Ajax (Asynchronous JavaScript and XML), JavaScript, XML (eXtensible Markup Language-basis for XHTML), SOAP (Simple Object Access Protocol), JSON (JavaScript Object Notation). Ajax is used to create web application using XHTML, CSS, the DOM (Document Object Model) and XMLHttpRequest. You can use Ajax to create more dynamic and interactive Web pages without needing to refresh or reload the page. SOAP is an XML based protocol to allow you to activate an application across the internet. During the user's communication to the web, data is fetched by an Ajax application which is formatted in XML or JSON format; the two widely used structured data formats. JavaScript programs use DOM to dynamically update the web page data. For example, Google Docs uses this technology to create web based word processor.

In addition to Ajax and JavaScript, Adobe Flex is another technology often used in web 2.0 applications. It is used to populate large data grid, charts and other user interactions. It is used as a plugin component. Application build in flex is displayed as Flash in the browser. Now, HTML5 is capable for doing such things. HTML5 is used for gaming, mobile and business application.

On server side, you can use technologies like PHP, Ruby, ASP.Net and Java Server Pages for developing web applications. Over time Web 2.0 has been used more as a marketing term than a computer-science-based term.

One of the key points of web 2.0 development is the availability of open content access which means that you can modify the content with few or no restrictions. Some of the key web 2.0 websites are Google, YouTube, Twitter, Wikipedia etc. MySpace, Flickr and Wikipedia are the websites that provide platforms for users to create and modify the content.

Figure-1 shows the characteristics and techniques of web 2.0. The following subsection explains some of the important web 2.0 technologies.

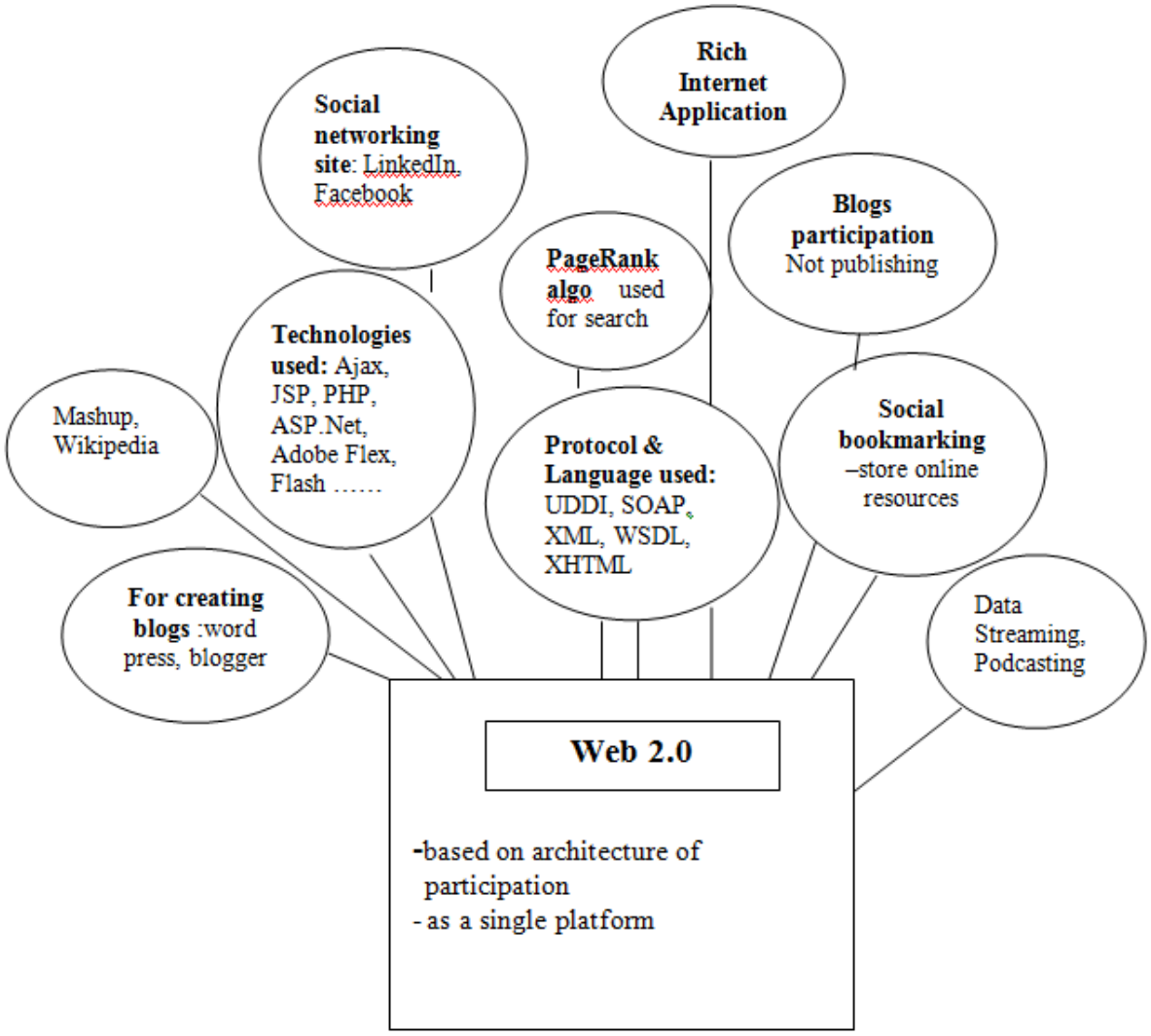


Fig.1: Web 2.0 Tree

1.2.1 Search

The term "Search" syntactically means *to look over carefully in order to find something*. In the context of Web, it is defined as - *to obtain information on the Internet using search engines*. Some of the popular search engines are Google, Yahoo, MSN, ASK, Bing, Alta Vista, Hot Bot etc. Search engines are the primary tools to find information on the web like text, images, news, videos, blogs and more. When you enter a keyword or phrase, the search engine finds the relevant web pages and shows in the order of ranking of pages. But, how does these search engines work?

In early days, search engines used text based ranking system to decide which pages are more relevant to given query. Modern search engines used different ranking methods to give best result. Google search is based on a priority rank called a PageRank. The PageRank algorithm

considers the web pages that match a given search string and display the sites with the highest PageRank at the top of the search results. Google's PageRank algorithm is based on the premise that a good page is the one that is pointed to by good pages. A detailed discussion on this algorithm is beyond the scope of this Unit.

1.2.2 Content Networks

A vast amount of information resides on the web. Searching a particular term is a complex problem because search engine are listed with number of links related to the search string. The solution of this problem is content networks. Content Networks are website or collection of websites that provide information in the form of articles, wikis, blogs, etc. These sites provide another way of filtering the vast amounts of information on the Internet by allowing users to go a trusted site that has already sorted through many sources to find the best content or has provided its own content. Some names of the content networks sites are eHow, About and HowStuffWorks.

1.2.3 Blogging

A blog or weblog is an informational website maintained by its blogger (or author). It contains discrete entries or post. It is displayed in reverse chronological order (the most recent post appears first). A typical blog contains text, images and links to other blogs or web pages. Most of the blogs are textual. Other blogs are on art (art blogs), music (MP3 blogs), videos (video blogs or 'vlog'), photographs (photoblogs) and audio (podcasts). In education, blogs can be used as instructional resources. These blogs are called as edublogs. WordPress and Blogger are the names of websites that enable anybody to start their own blog. WordPress is one such advanced blogging tool and it provides a rich set of features, through its administrative panels, you can set options for the behaviour and presentations of your blog and easily compose a blog post, push a button and be published on the internet instantly.

Micro-blogging is another type of blogging but it contains very short post. Micro-blogging is used in Twitter. The blogs are interactive as they allow readers to leave comments. Bloggers or blog author do not only produce content to post on their blog but also build social relations.

1.2.4 Social Networking

A Social Networking is an online platform that focuses on building the social network or social relation among the users who share their profiles, interests and activities. Friendster, Myspace, Xing, LinkedIn, Twitter, Orkut and Facebook are some popular social networking sites. Facebook is the largest social networking site. LinkedIn is a business oriented social networking site. It allows users to stay in touch with professional contacts. Social networks are also being used by teachers and students as communication tool. Some social networks have additional features such as to create groups or forums and upload photographs or videos. Using social networking site, you can interact by adding friends, commenting on profiles, joining groups and having discussions. Social networking sites allow two way interactions. You can make accounts, publish content and involve in two way communication.

1.25 Social Media

Social Media is a kind of interactive website that does not give any information but interact with you while giving you that information. In social media, user interaction is possible by commenting on photo, editing articles in wiki and voting on articles.

Here are some examples of social media websites:

- **Social Bookmarking:** allow internet users to organize and store to online resources. For example, Delicious, Blinklist. Some of the Bookmarking websites are free and some provide paid services. It allows users to save links to web pages with these bookmarking sites to remember and share.
- **Social News:** You may interact by voting for articles and commenting on them. For example, Digg, Propeller.
- **Social Photo and Video Sharing:** You may interact by sharing photos and videos. For example, YouTube, Flickr.
- **Wikis:** multi-lingual, web-based encyclopaedia Wikipedia. You may interact by adding and editing articles.

1.2.6 Rich Internet Applications (RIA's)

RIAs are a combination of user interface functionality of desktop applications; web applications and interactive multimedia communication techniques (refer to figure2). Rich Internet Applications are fast, powerful and user friendly. A Rich Internet Applications have same features and functions as the traditional desktop applications. A RIA runs inside a web server and does not require software installation on the client side. RIA split the application processing on different stages as:

- operations related to data manipulation are operated on the server side
- user interaction activity is performed on the client side within a special area of the client desktop called sandbox.

The basic idea of this approach is to decrease the frequency of client – server traffic for handling local activity, calculations and so forth.

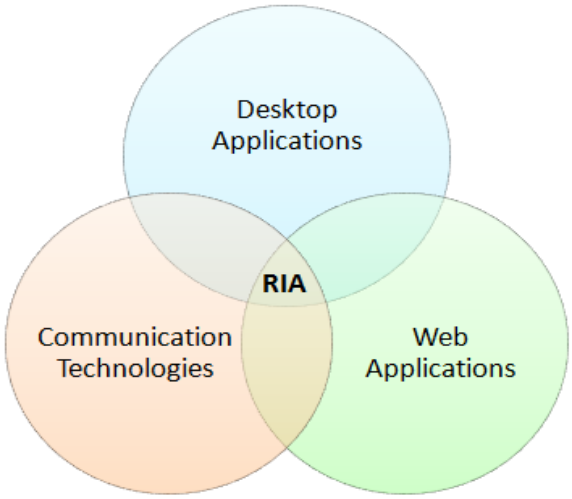


Fig.2: Rich Internet Application

For implementing the Rich Internet Applications, you can use different types of software such as Ajax, JavaScript, Adobe Flex, Flash, Ajax with PHP and many more. Using these

technologies, Rich Internet Applications (RIA) solutions provide good interactivity to the users.

1.2.7 Web Services

In Web 2.0, some of the websites are facing more user interaction such as YouTube and Facebook. Basically, these websites are facilitates the user for free interpersonal content sharing, it means that it involves person-to-person interactions through these websites that enables the users for content creation, sharing and manipulation. This is the key feature of the web 2.0. When these interactions are between two or more people and other resources on the Web, then the role of web services come. The Web services provide the improvements in terms of interactions or interconnections that may exist between two or more different web resources and hence between those organizations that deliver them. For example, if any organisation wants to include a feature of credit card payments online, it can either set its own setup or can integrate the web service of a Payment Service Provider e.g. Paypal into the website. The user will make their purchase from the Company's website but for payment, the control is automatically transferred to the Service Provider's website. All of this will happen automatically between two organisations.

Web services combines' different protocol to allow all components to interoperate entirely under the control of computer, without human intervention. It means that when a system requires a service, computer can implicitly find that service on the web. For example: In client/server model, a client sends a request to web service (in the server) for weather information. The web service return forecast through service response to the client as shown in the following figure3.

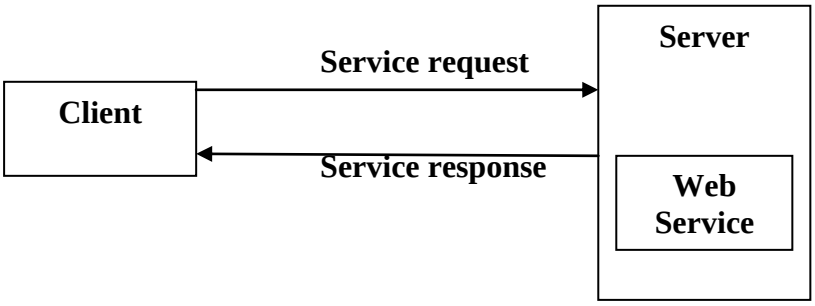


Fig.3: Client/Server Model for weather information

Web service has three roles for providing interaction between different components over web. These are:

- Servic e requestor,
- Servic e registry and
- Servic e provider.

These three roles are combined with find, publish and bind operation as seen in the following figure4. Different protocol and languages are used in web services such as XML, SOAP, WSDL and UDDI open standards over an Internet protocol backbone. The service provider

for providing services, write the services in WSDL (Web Services Definition Languages) and service registry uses UDDI (Universal Description Discovery and Integration Service) to listing what services are available. The service requestor for requesting service uses XML and SOAP standards over Internet Protocol backbone. XML is used to tag the data; SOAP is used to transfer the data. On the web, there are number of web services that use different standards.

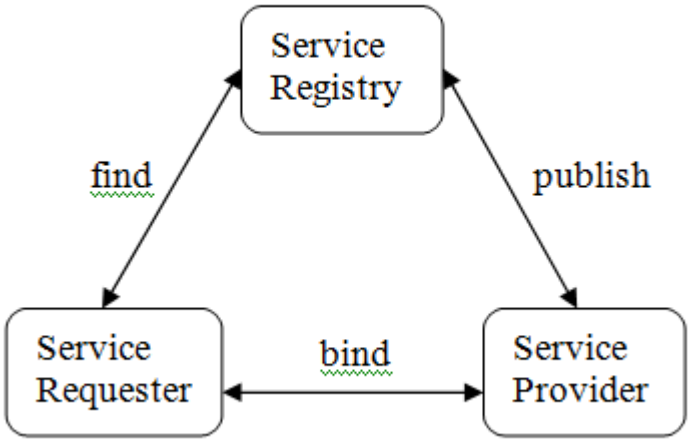


Fig.4: Web Services

1.28 Mashups

Mashups is a web application that retrieved data from two or more external sources to create entirely new and innovative services as shown in figure5. This is a new breed of web based data integration application. There are many types of mashups such as consumer mashups, data mashups and business mashups. Consumer mashups combine visual elements such as maps and data from multiple sources. For example, Wikipediavision is a site that combines Google Map and a Wikipedia API. Business mashups define applications that combine their own data and application with external web services to create single presentation. Data mashups combine data from multiple sources into single source on a similar type of media and information. Mash-ups are often created by using a development approach called Ajax.

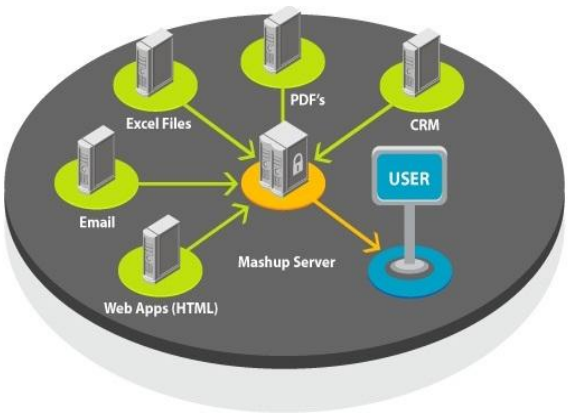


Fig.5: Mashups

1.29 Widgets and Gadgets

Widgets are also referred to as Gadgets. A web widget is a small piece of code that can be placed on a webpage, social media or blog. Widgets are useful application in the form of on-screen device viz., clocks, auction-trickers, event countdowns, flight arrival information and daily weather. A common example of a widget on a website is ad blocks such as Google Ads. YouTube also provides a widget, allowing you to make play list of your favourite videos. These dynamic widgets can be used in your website to enhance its appeal to the users.

Another kind of widget is desktop widget. A **desktop widget** is a small application that resides on your computer's desktop and does not require a web browser to be opened while a web widget is a component of web page, so it does require a web browser. Yahoo widgets are popular source of desktop widgets. Yahoo and Microsoft Vista also provide a widget toolbox to manage desktop widgets. Windows 8 operating system has number of desktop Gadgets as seen in figure 6.

Figure 6: Windows 8 Desktop Gadgets

1.2.10 Podcasting, Message Board and Data Streaming

Podcasting is a way to allow users to share their work over the web. A podcast is an audio file that is recorded and delivered over the web. User can listen to it any time. There are thousands of podcasts available on the web ranging from general topics to those which focus on specific topics such as computers, education and music. Teachers and library media experts are using podcasts to deliver their content to students.

A **message board or forum** is a Web 2.0 application that puts students in a web-based place where they can post a comment about any subject, and other students on the same message board or form can respond to the comments or post comments of their own. The message board is a good place for teachers and students for discussing a particular topic.

Data Streaming is a technique for transferring data at high speed rate so that it can be processed as a steady and continuous stream. The client side computers do not have fast access to download heavy media files quickly. With data streaming, the client browser can start displaying the data before the entire file has been transmitted.

Check Your Progress 1

1. Explain the term Web 2.0.

---

2. Define the Search Engine.

---

3. Explain Social media with examples.

---

4. What is a Mashup?

---

5. Differentiate between web widget and desktop widget.

1.3 Introduction to XHTML/HTML5

As discussed earlier, a web page is a document that contains information that is displayed on a browser. These web pages are created using the markup languages XHTML/HTML5. You have already gone through the basic concepts of HTML in MCSL-16 course. This section describes some of the basic entities used in the HTML in more details.

XHTML (Extensible Hypertext Markup Language) is a markup language. A markup language is a way of describing the meaning of and relationship between different parts of document. XHTML and its predecessor HTML (Hypertext Markup Language) are the most widely used languages on the web. XHTML defines a set of rules for encoding documents in a format that is both human-readable and machine-readable that reproduces all the features of HTML. XHTML documents can be processed by both Extensible Markup Language (XML) and HTML software. XHTML markup separates presentation details from the information. For this purpose they use style sheets called CSS, which you will learn in Unit 2.

Several features and structural rules of all XHTML document are similar to HTML document (pl. refer to MCSL-016 Block1) and for more details you may refer to the website http://www.w3schools.com.

HTML5 is the new standard for HTML. HTML5 is markup language for structuring and presenting content for web. HTML5 is a platform for mobile, gaming and enterprise applications. Major web browsers such as Internet Explorer, Mozilla Firefox, Apple Safari, Google Chrome, Opera support many of the new HTML5 elements to correctly display web pages. You must use the most recent version of browser to use HTML5.

HTML5 supports most interesting new features like <canvas> element for 2D drawing, <video> and <audio> elements for media playback, scalable vector graphics (SVG) content (that replaces the use of <object> tags) and MathML for mathematical formulas. In addition to this, some new content-specific elements like <header>, <footer>, <section> and <nav> are used to enrich the content of documents. Other new form controls such as calendar, date, time, email and search are to be added to the functionality of HTML5. A detailed discussion on all the features of HTML5 is beyond the scope of this Unit. You may refer to further readings for more details. Before explaining the new features of HTML5, let us first define some basic tags.

<!DOCTYPE html> Tag: In HTML5, there is only one <!doctype> declaration. This declaration must be first statement in your HTML document that is even before <html> tag. It

is not a HTML tag rather it is an instruction to the browser about the version of HTML in which the page is written.

<HTML> Tag: This is root element. It defines the content of web page. The opening HTML tag that is <*html*> immediately follows the DOCTYPE declaration and closing tag that is </html> is placed at the end of the document. The html element must contain head and body element.

<HEAD> Tag: In head portion of the document, you can write information about the document. You can define title, meta, style, script and link element. You will learn about the style, script and link element in the next units of this block. The head element is written immediately after the opening html tag and end with closing tag appears before the body tag.

<TITLE> Tag: It displays text at the top of browser window.

<META> Tag: The <meta> tag is used for declaring metadata (i.e. data about data) for html document. This element is placed between the head element and the basic use of meta element is to provide information about the page to search engine (i.e. Goggle, Yahoo), browsers and other web services. If you want top search engine rankings, just add meta tag in your web pages. You can use multiple meta element with different attributes on same page.

Some examples of usage of meta tags are given in the following three examples:

Example 1: Define keywords for Search engine

<meta name="keywords" content="HTML, CSS, XML, JavaScript" />

Example 2: Define description of your page

<meta name=”description” content=”Web Page Designing Issues” />

Example 3 - Refresh your document every 60 seconds:

<meta http-equiv="refresh" content="60">

<BODY> Tag: This tag contains the body of the web page that is displayed in the browser display area. In body tag, you can write those tags which are specific to the contents to be displayed in browser.

You can now create a simple web page with minimum but required tags. You can simply use a Notepad as a text editor to create a file of a webpage and save file with an .html file extension. To see the result of your work in one of the two ways:

1) Find the respective location of html file (prg1.html) and double click on it.

2) In your browser, click on File/Open File and browse to the name of file.

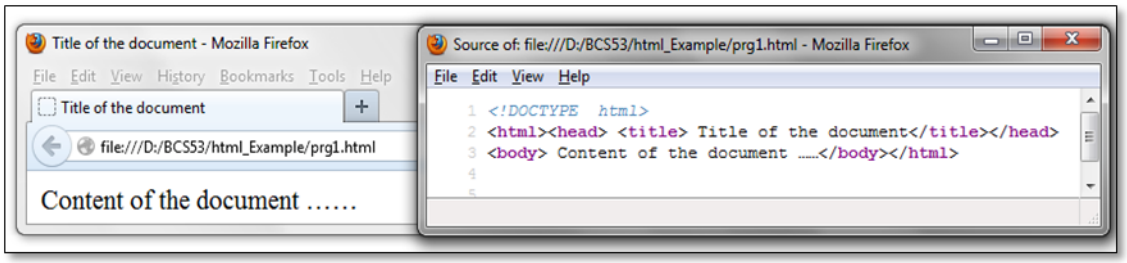


Figure 7: Display of a simple web page and the source code of this web page prg1.html

HEADING Elements: <h1>, <h2>, <h3>, <h4>, <h5> and <h6> heading elements are used for different level of heading. <h1> is the highest level of heading and <h6> is the lowest level of heading. <h1> is used once in web page and headings should be used in order. The output and source code of the program are as follows:

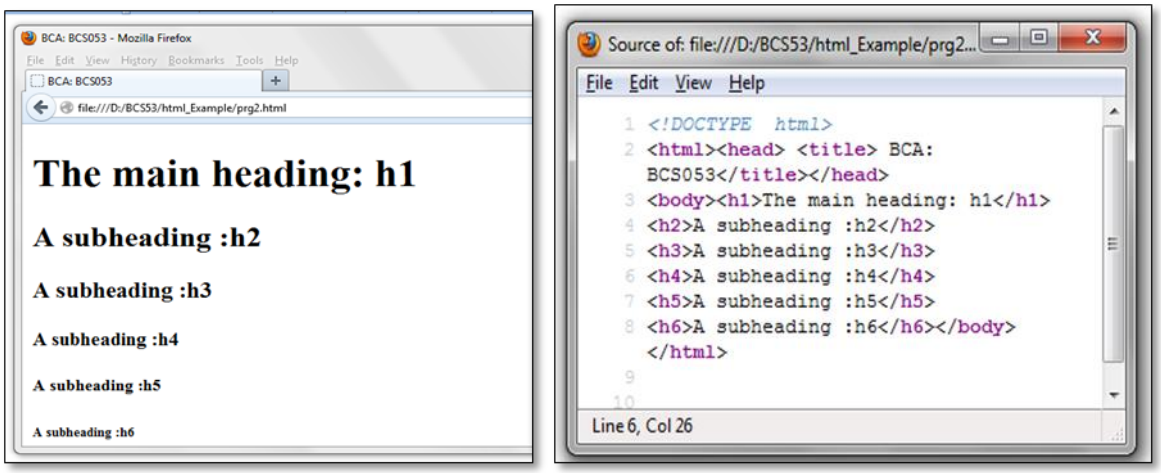


Figure 8: A page with headings and related source code.

<P> Tag: This tag is used for writing paragraph text.

<TABLE> Tag: As you know, table contains row and column. This tag is used to create a table within a web page. In conjunction with table tag, you can used <tr>(table row),<th> (table head cell), <td> (table data cell), <thead>(table header), <tfoot>(table footer) and <tbody>(table body) tags.

<TR> Tag: This tag is used for creating a row. It must appear inside the table element. Two types of cell used in table row. The first cell is <th> and other is <td>.

<Th> Tag: It defines a row or column header. It acts as a table header or data. <td> should be used instead <th>.

<TD> Tag: It creates a column that holds data of table such as text, images, links, lists and other table also. It appears within <tr> element. Table data cell has attributes like rowspan and colspan. The rowspan attribute defines the number of row a cell should span and colspan attributes specifies number of column should span. The syntax for colspan and rowspan are as follows:

<td colspan=”number”> and <td rowspan=”number”>

Attributes appear inside the opening tag and their values sit inside quotation marks.

<THEAD> Tag: This tag is used for table header along with < tfoot> and <tbody> tags.

<TFOOT> Tag: It can be used once within table and must appear before the <tbody> tag.

Consider the following example for table element:

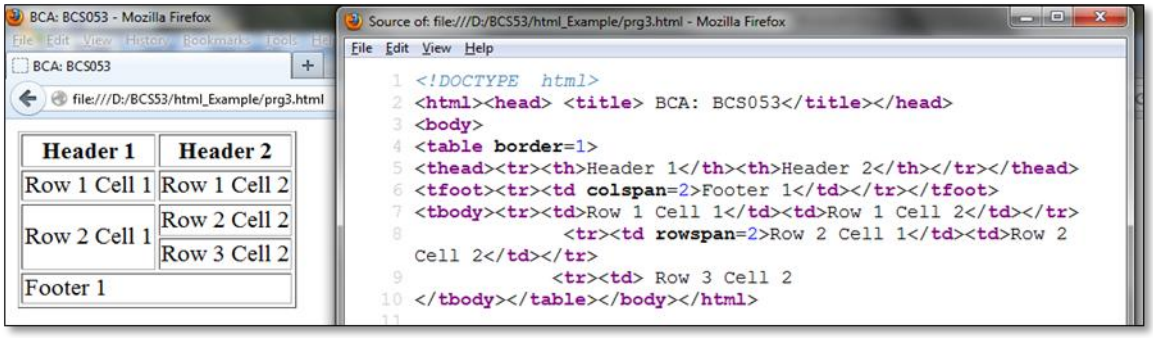


Figure 9: Web page and source code for a table

In the above example, two attribute colspan and rowspan of <TD> element are used. The colspan attribute is used to create a cell that spans more than one cell. The last row or footer of the table shows the result of colspan attribute - two cells are merged into one. The rowspan attribute is similar to colspan, except, it span across rows rather than columns.

ANCHOR Elements: If your website has several pages, you need to provide links between HTML pages, the tag is used to create link is called <a> which stands for anchor. The anchor <a> element is used to create a hyperlink which provides a link from one page to another or within the same page. The syntax of this tag is as follows:

In the above syntax, href attribute defines the url of linked resources or web page and target specifies where to open a linked file and name specify the name of intra link which is used within same page.

Some examples of usage of <a> element are given in the following two examples:

Example 1:

<!—linking to other pages-->

< a href=”head.html”> Header page

Example 2:

<!—links within a page-->

 The above example is used to provide links from one page to another page. Now, this example shows you how to use the same element to links within a page. This is used at top of the page and at end of the page, you can use following statement which transfer control back to top.

Back

Comments:

You can use the following two tags that enable you to add comments within the HTML code. These comments are not displayed by the browser.

Example:

<!--This is a HTML comment -->

<comment> This is also a comment </comment>

FORM Elements:

Form elements, as the name suggests, contain predefined field names with blank or default values which you can fill and submit to their respective website. You can create a form for your website using form elements such as input, password, text area, radio button, submit button, select with option, label, fieldset and legend. Forms can be used for filling information online. Some examples of forms include online examination form, feedback forms and so on. Forms are useful tool for client interaction on a website.

<INPUT> Element: The INPUT element is most important element. You can build an entire form using no other element due to its type attributes which have wide variety of values. The syntax is as follows:

```
< INPUT [type=text| password| checkbox| radio| submit| reset| image| button| hidden]
[name=controlName] [value=controlValue] [checked] [size=controlWidth]
[maxlength=word Length] >
```

In the above syntax, name attribute is used to define a name of control element and value attribute specifies value of input control element. The size attribute is used to define width of control element and maxlength attribute indicates a length of input element in characters. The type attribute is defined in the following sections:

The text value: The text value of type attribute enables you to input single line text in a text box. For example, define text box using input element

```
<input type=”text” name =”username” size=30 >
```

The password value: It is similar to text attribute except that the visible portion of text on screen is masked with other character so that no one can read the password. For example,

```
<input type=”password” name =”pwdname” size=10 maxlength=”8”>
```

In the above example, name attribute specify the name of input element and maxlength attribute specify the maximum numbers of characters allowed in input password control. The size attribute indicates width of input element.

In similar way, you can change only value of attribute type of input element for creation of other elements.

<TEXT AREA> Element: This element is similar to INPUT elements text type. Using this tag, user can type a larger section of text in multiple lines rather than a single line text in text boxes. The rows attribute of textarea is used to specify the viewable row and cols attribute is used to specify the viewable columns of the text. For example:

```
<textarea name =”username” rows=5 cols=20> </textarea>
```

<LABEL> Element: It is used to add label to form control. It is used with radio button or checkbox option element.

<SELECT> Element: The select form control element is used in conjunction with optgroup and option element to create a list of choices that the user can choose from. It may be drop down menu or list box. The optgroup (option group) is used to define group of elements in a <select> form element.

For this, you can see the following program. Three <select> elements are used in this example. The first <select> statement is used in conjunction with <optgroup> to define group of elements. The label attribute of <optgroup> element specifies the two names such as Maruti Cars and Germans Cars as group header. Each option group element have <option> element to describe the further options in the list such as Wagon R and Swift Dzire in Maruti Cars header. The second <select> element is displayed a simple drop down menu with two option i.e. male and female. The third <select> element is used as a list box.

```
<!DOCTYPE  html><html><head>
<title>BCA:BCS053</title>
<h2>Example for select element</h2>
<select><optgroup label="Maruti Cars">
        <option value="wagon R">Wagon R</option>
        <option value="Swift Dzire">Swift Dzire</option>
        </optgroup>  <optgroup label="German Cars">
        <option value="mercedes">Mercedes</option>
       <option value="audi">Audi</option>
       </optgroup>
</select>
<select name="Gender">
<option value="">Male</option>
<option value="">Female </option></select>
<select name="country" size="2">
<option value="">India</option>
<option value="">Nepal</option>
<option value="">Canada</option>
</select></body></html>
```

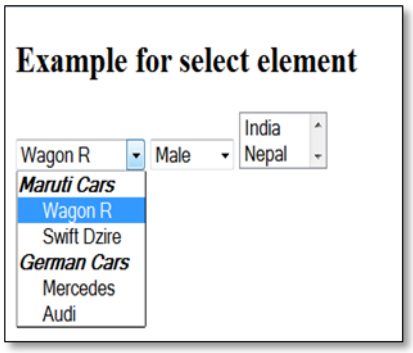


Figure 10: Source code and output screen for select element

<FIELDSET> Element: This element is used for grouping related form elements by drawing a box around the related elements.

<LEGEND> Element: It is used to display a title or caption for group of elements which is grouped by <fieldset> element. It is also used in conjunction with <figure> and <details> element. By using the both elements <fieldset> and <legend>, you can make your web page easier to understand for your user.

```
<!DOCTYPE html><html><head></head><body>
 <fieldset>
   <legend><b>Personal Information</b></legend>
   Name: <input type="text" size="20">
  Address<input type="text" size="20">
</fieldset>
```

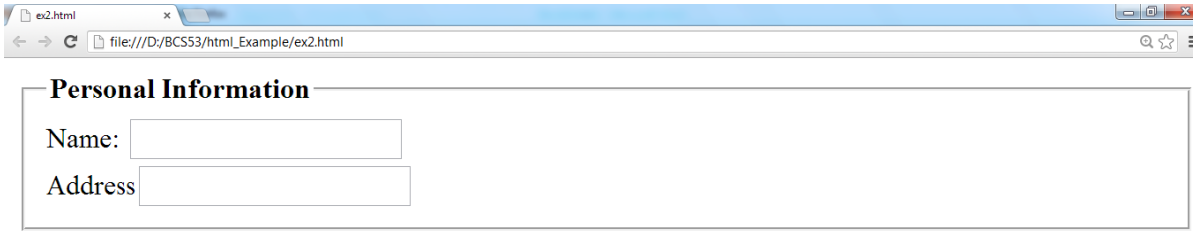


Figure 11: A web page showing Personal information and source code

The above example shows that the legend element creates a caption as 'Personal Information' for fieldset element. The fieldset element is used as grouping the form elements i.e. name and address.

A detailed example of these elements is given in section 1.6. In the remaining section, you can learn about the new features of the HTML5 markup language:

<ARTICLE> Tag: This tag is used to represent an article. It is used in blog for post entry, a newspaper article. The following example shows the result for anchor, article and paragraph elements.

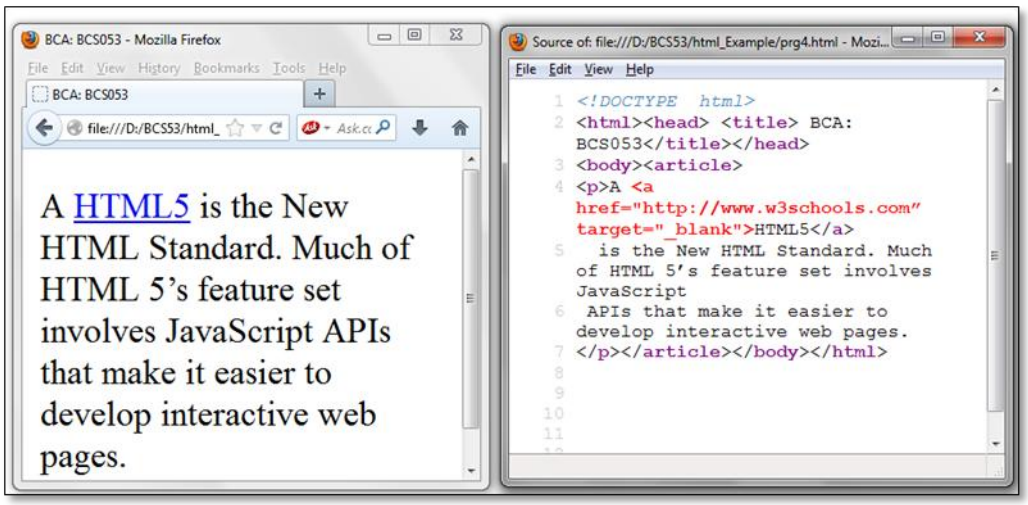


Figure 12: A web page and source code for article and anchor element

<HEADER> Tag: The <header> element is used to define section's heading. The <header> tag cannot be placed within a <footer>, <address> or another <header> element. It is used in a search form or any relevant logos.

<FOOTER> Tag: The HTML <footer> tag is used for defining the footer of an HTML document or section.

<HGROUP> Tag: It is used to group a set of <h1> to <h6> heading elements when a heading has multiple levels.

Example of <article>, <header>, <footer> and <hgroup> tags:

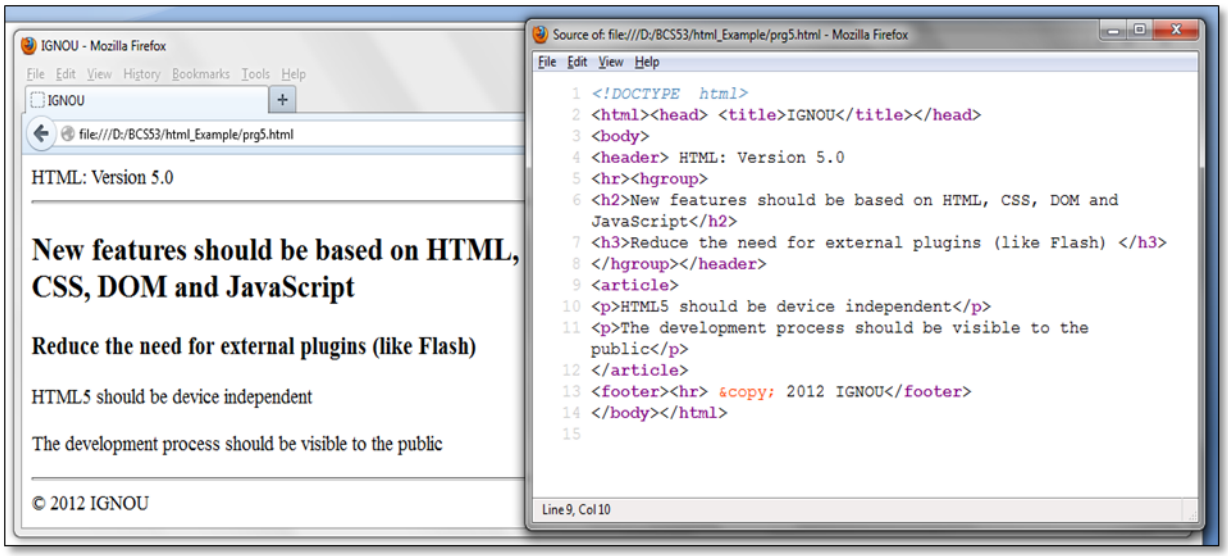


Figure 13: A web page and source code for different HTML5 tags

<AUDIO> Tag: The <audio> tag is used to specify audio on an HTML document.

<VIDEO> Tag: The HTML5 <video> tag is used to specify video on an HTML document.

Both the tag audio and video accepts attributes i.e. control, preload, autoplay, loop, src that specify how the video/audio should be played. The description is as follows:

- controls – it display the respective control on web page
- autoplay- makes the audio/video play automatically.
- loop – play again once it has finished.
- src – name of audio/video source file

This is an example of how you can make use of the HTML5 <video> and <audio> tag in a web page to include a video clip without the need of Flash or Silverlight to be installed in the browser.

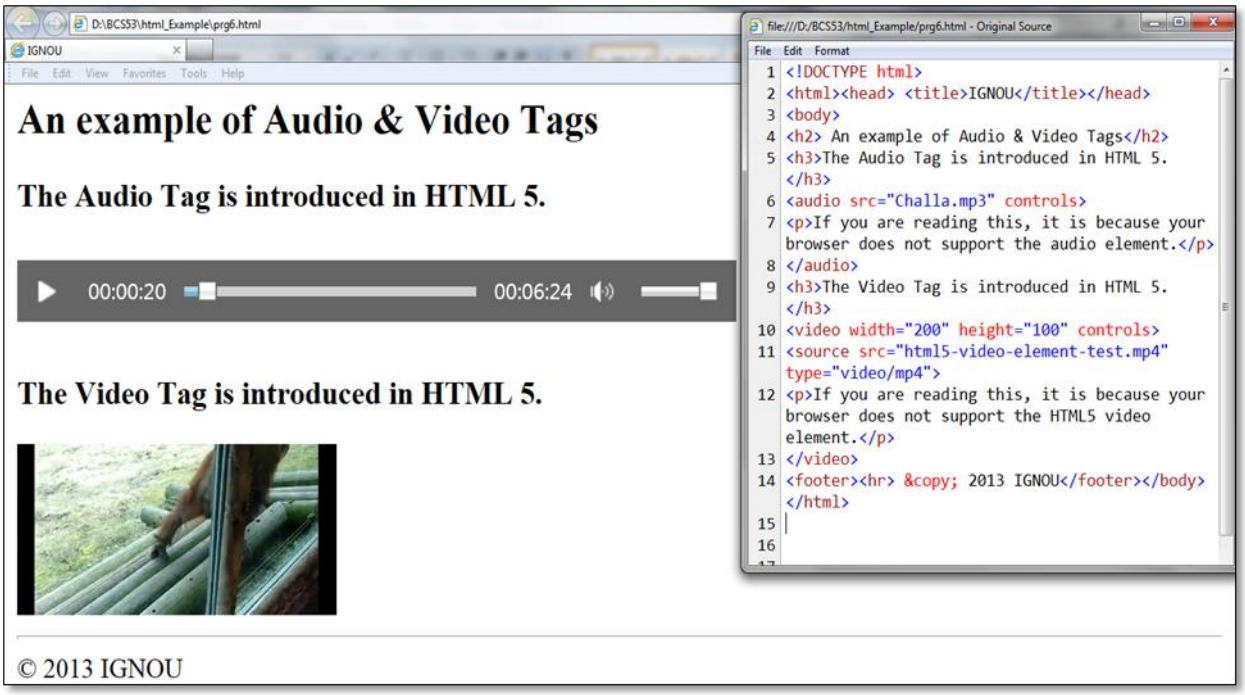


Figure 14: A web page for Audio and video tag

<SVG> Tag: Scalable Vector Graphics is used for describing 2D-graphics application. It is useful for vector type diagrams like Pie charts; Two-dimensional graphs in an X, Y coordinate system etc. For example:

```
<!DOCTYPE html>
<head><title>BCA:BCS053-SVG
</title></head>
<body>
<h3>HTML5 SVG Circle & Rectangle</h3>
<div>
<svg  >
   <circle cx="100" cy="50" r="40"
  stroke="green"  stroke-width="2"
  fill="yellow" />
</svg>
</div><div>
<svg  >
<rect  width="200" height="50" fill="cyan"
stroke="black" stroke-width="3" />
</svg></div></body></html>
```

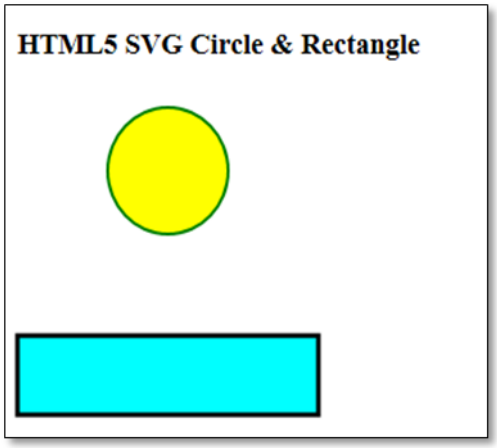


Figure 15: HTML5 program and output screen for Circle & Rectangle

The above program shows circle and rectangle which has drawn by <svg> element. For the creation of circle, a keyword 'circle' and x, y coordinate is defined. The <circle> element is written inside the <svg> element. In this example, cx attribute of circle specifies x coordinate, cy is y coordinate of the circle and r represents radius of the circle. The attribute stroke is used for fill the color and attribute stroke-width is used for defining the width size of outermost area of circle. The color of inner circle is filling using the fill attribute. In the same way for creation of rectangle, a keyword 'rect' and width, height is defined. The <rect> element is written inside the <svg> element.

<CANVAS> Tag: This element is used to for creating graphics via scripting (usually JavaScript). Canvas has several methods for drawing boxes, characters, path and adding images. Canvas is a rectangular area on which you can draw object using JavaScript.

<TIME> Tag: This tag is used for declaring the date/time within an HTML document.

<MENU> Tag: The HTML menu tag is used for specifying a list of commands.

<MARK> Tag: This tag is used for indicating text as marked or highlighted for further reference.

Consider the following code for <mark>, <time> and <menu> Tag.

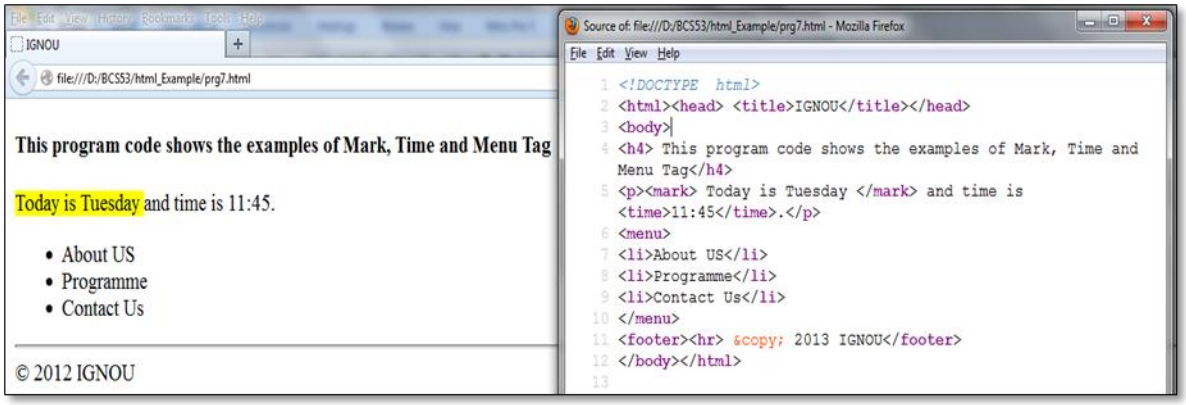


Figure 16: A HTML page with source code

<EMBED> Tag: If you want to play a video in a web page, you can upload the video to YouTube in proper HTML code to display the video.

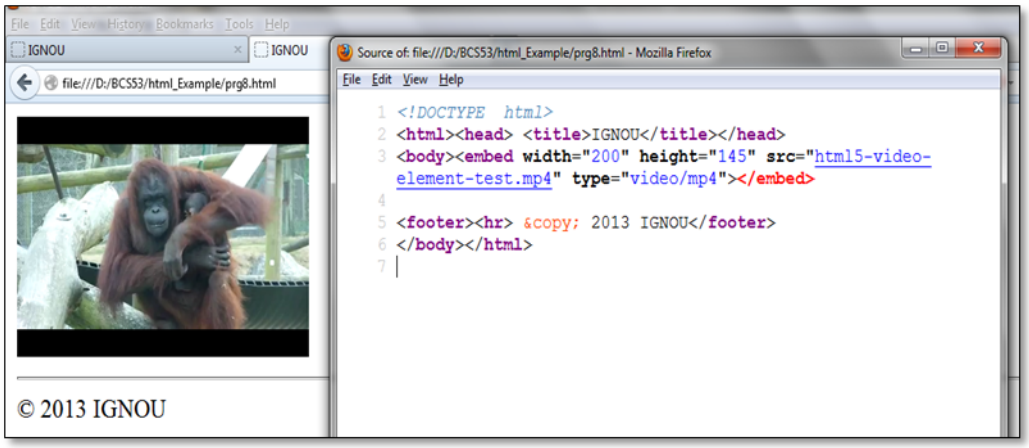


Figure 17: A web page source code and screen for embed tag

You can embed a video file in the following format:

<embed width=”300“ height = “150 “ src=”videofile.mp4” type=”video/mp4”></embed>

In the above code, src attribute define the url of the video file and using width, height attribute, you can defined the width and height of the video clip.

<DETAILS> Tag: This tag specifies additional details that the user can view or hide on demand. It can be used in conjunction with **<SUMMARY>** tag to provide a heading that can be clicked on to collapse/expand the details as required. You could use `<details>` to toggle the comments section of a blog, member profiles, and details of a download, complex forms, or in web applications. **Only browser Chrome supports the `<details>` and `<summary>` element**.

For Example:

```
<!DOCTYPE html><html>
<head><title>BCA:BCS053</title></head><body>
<details> <summary><label for="name">Name:</label></summary>
  <input type="text"  id="name" name="name" />
</details> </body></html>
```

In this program, detail element create a collapse/expend toggle mark on the page, whenever you clicked on this mark, it show/hide the text.

<ASIDE> Tag : This tag is used for defining the content aside from the page content. Basically this tag is used to represent quotation text that is related to page.

<MATHML> Tag: The MathML standard is used for mathematical notations on web page.

<BDI> Tag: This tag can be useful when the text direction is unknown such as in the case with user generated content.

<NAV> Tag : The <nav> tag defines a section of navigation links.

<FIGURE> Tag: This tag is used for annotating illustrations, photos and diagrams. The <figure> tag is used in conjunction with **<FIGCAPTION>** tag to provide a caption to the figure.

Consider the following code for <aside>, <bdi>, <figure>, <figcaption> and <nav> tags.

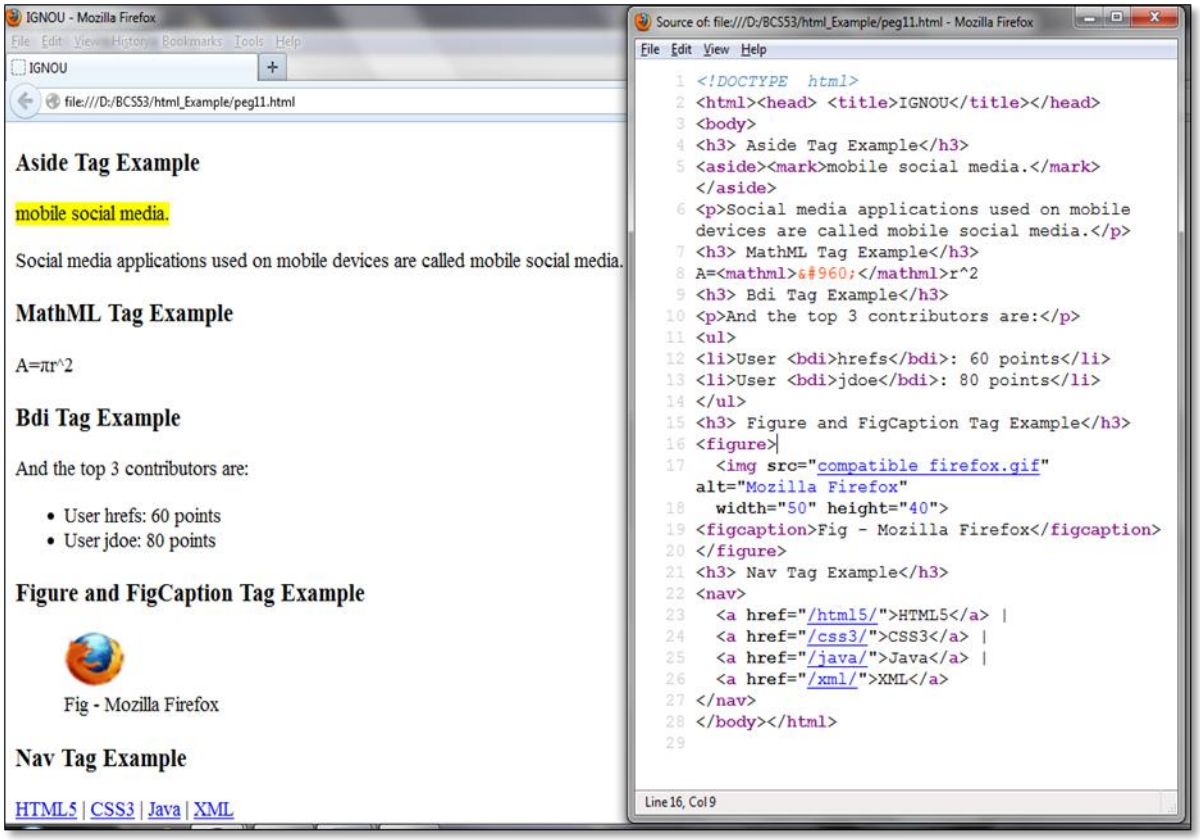


Figure 18: A web page display different tags of HTML 5

Check Your Progress 2

1. What is the use of <!DOCTYPE html> tag in HTML5?

---

2. Explain the SVG element in HTML5 with Example?

---

3. Write a HTML5 program using <details> and <summary> tag.

---

4. Differentiate between SVG and Canvas tag?

---

5. What is the purpose of aside tag in html5?

1.4 Syntactic Differences between HTML and XHTML

HTML (**H**yper**T**ext **M**arkup **L**anguage) and XHTML (E**X**tensible **H**yperText **M**arkup **L**anguage) are both languages in which web pages are written. HTML is based on SGML (Standard Generalized Markup Language) while XHTML is XML (eXtensible Markup Language) based. XML is discussed in details in the Unit 3 of this block.

There are some significant differences between the syntactic rules of HTML and XHTML. These are:

1. Case sensitivity: In HTML, it does not matter whether tags are lowercase, uppercase or a mixture of both. In XHTML, all tags must be lowercase. For example, use <head> instead of <HEAD>.

2. Closing tags: Closing tags are often optional in HTML but are always required in XHTML. For example, every paragraph must begin with tag and end with a tag. Likewise, every <tr> table row must have a closing </tr> and so forth.

3. HTML tags that do not enclose any content such as <hr>, <br> and <img>. In XHTML, it must contain a slash. For example, <hr />, <br />, . These statements are treated by XHTML interpreter as a closing tag.

4. Quoted attribute values: In XHTML, all attribute values must be enclosed in quotation marks regardless of what characters are included in the value. For example

   <img src=pic.gif border=1> is valid HTML but it must be written as
    to be valid in XHTML.

5. Explicit attribute value: All attributes must have values in XHTML. For example,

   <input type = “checkbox” checked> should be written as

   <input type=”checkbox” checked=”checked” />

6. id attribute : The id attribute is used to identify objects. For example,

   In XHTML, the use of id is encouraged and the use of name is deprecated.

7. Element nesting: Tags must be properly nested in XHTML. For example,

    _{This is bolded subscript} should be written as
    _{This is bolded subscript}.

8. Extra whitespace that appears in attribute values is stripped out. For example, If the following code is encountered

   <img src=” pic.gif “ >
   The XHTML processor will interpret it as <img src=”pic.gif” > .

1.5 Standard XHTML /HTML5 Document Structure

The previous section discussed rules for writing XHTML/HTML. In this section, you will learn about the structure of the document. Each document is structured into two parts, the HEAD, and the BODY. The head contains information about the document whereas the body contains the body of the text, and is where you place the document content to be displayed. Both of these are contained within the HTML element.

Three features in figure 19.0 are characteristic of XHTML/HTML5 documents. These are:

1. Every XHTML document must begin with an XML declaration. All XML documents should begin with the following line: <?xml version =”1.0” encoding = “utf-8” ?>

   This declaration states that the rest of the document is XML. The version attribute specify the version number, which is still 1.0. The encoding property defines the encoding in which the document is stored on a computer.

   In HTML5, every document must begin with <!DOCTYPE HTML>.

2. XHTML documents require a DOCTYPE declaration just after the XML declaration. A DOCTYPE declaration is the XML mechanism for specifying the document type declaration (DTD) relevant to the document. The DTD specifies the rules used by the markup language. You will learn more about the XML and DTD in the unit 3 of this block.

   For example,

   <!DOCTYPE html PUBLIC “-//w3c//DTD XHTML 1.1//EN”
   http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd> this declaration defines the location of the DTD specified via a URL.

   HTML5 does not define a DTD.

<?xml version =”1.0” encoding = “utf-8” ?> <!DOCTYPE html PUBLIC “-//w3c//DTD XHTML 1.1//EN” http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd> <html xmlns=http://www.w3.org/1999/xhtml> <head> …….. </head> <body> content of the document…….. …….. </body> </html>	<!DOCTYPE HTML> <html> <head> <title>Document Name</title> <link href="name_of_cssFile " rel="stylesheet" /> <script src="name of js file"></script> </head> <body> content of the document…… </body> </html>

Figure 19.0: Standard XHTML and HTML5 Document Structure

3. XHTML document must include the four main tags <html>, <head>, <title> and <body>. The <html> tag identifies the root element of the document and this element must use the xmlns attribute to declare the location of the namespace for XHTML syntax. The only allowed form is :

   <html xmlns=http://www.w3.org/1999/xhtml>

   In the same way, HTML5 document must contain <html>, <head>, <title> and <body>. Its also include <script> and <link> tag. The link tag is used to include cascading style sheet and script tag is used for java script file.

1.6 Example of HTML5 Form

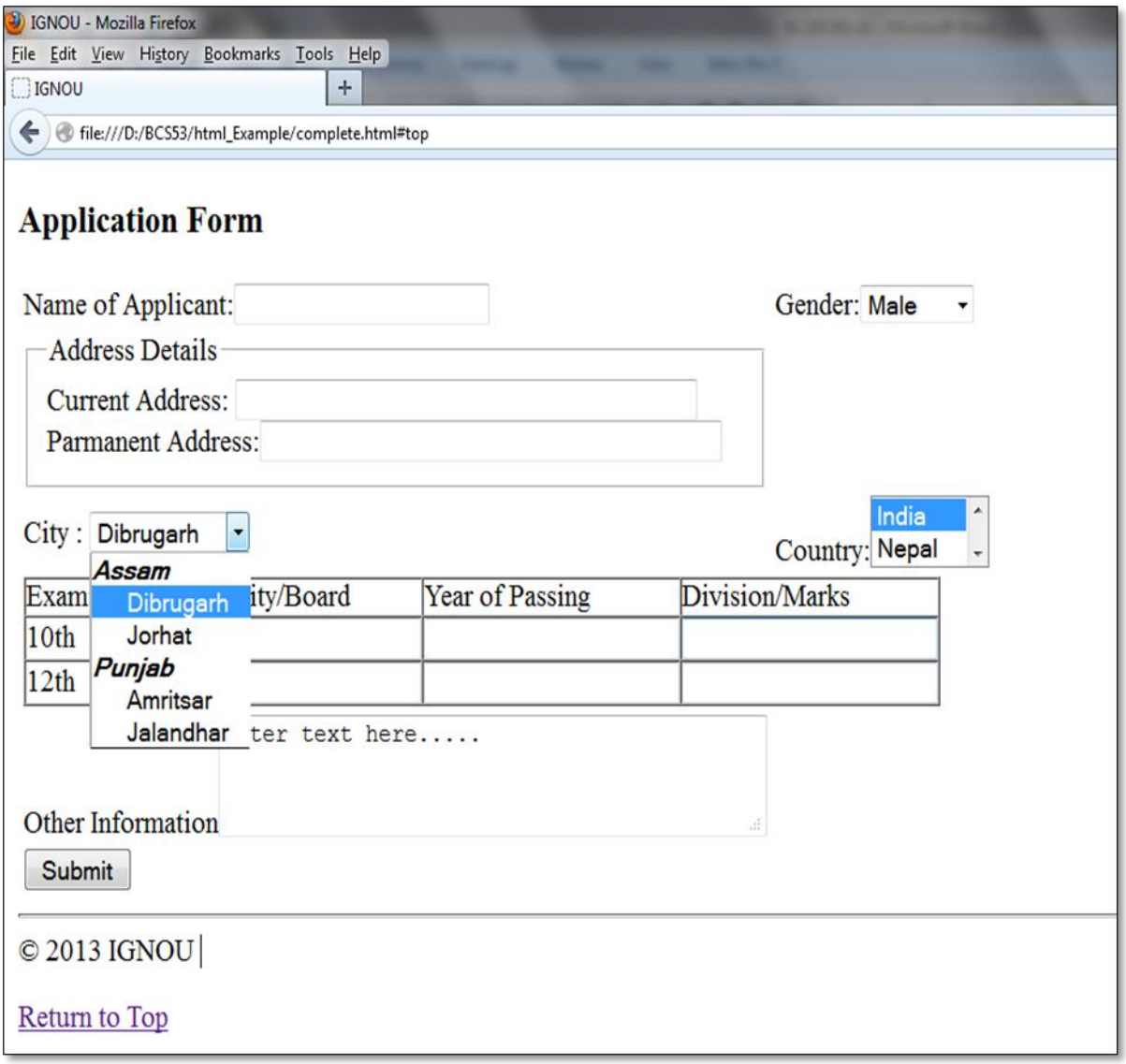
The following program creates an Application Form which includes fields such as name, gender, address, qualification and some other information.

In the example, numbers of elements are used such as title, meta, table, select, fieldset, legend, optgroup, option, input, textarea, label, footer and anchor. The title and meta elements are defined in the head portion of the HTML page and rest of the tags are defined in the body tag. The title tag tells to the users and search engine what the topic of a particular page. The meta tag is used to provide information about the page to search engine. The method, name and action attributes are defined within the form element. The name attribute is used to define a name of the form. It is used to reference form data after the form is submitted. The action attribute is used to send form data after the form submission. The get method of form element is used to get requests data from the specified resource which is defined in action attribute.

```
<!DOCTYPE  html>
<html><head> <title>IGNOU</title>
<meta name="keywords" content="HTML5, IGNOU, application">
</head><body ><h3><a name="top"> Application Form</a></h3>
<form name="HTMLForm" method="get" action="">
<table border="0">
<tr><td>Name of Applicant:<input type="text" name="FName" size="20" maxlength="50">
</td>
<td>Gender:

<select name="Gender">
<option value="">Male</option><option value="">Female</option>
</select>

</td></tr><tr><td >
<fieldset ><legend>Address Details</legend>
Current Address: <input type="text" name="caddress" size="40" maxlength="50" >
<br />
Parmanent Address :<input type="text" name="Paddress" size="40" maxlength="50">
</fieldset>
</td><td></td></tr><tr><td>City :

<select>
<optgroup label="Assam">
<option value ="">Dibrugarh</option>
<option value ="">Jorhat</option></optgroup>
<optgroup label="Punjab"><option value ="">Amritsar</option>
<option value ="">Jalandhar</option></optgroup>
</select>

</td><td>Country:
<select name="country" size="2">
<option value="">India</option><option value="">Nepal</option>
<option value="">Canada</option></select>

</td></tr>
<tr><td colspan="2"><table border="1" cellpadding="0" cellspacing="0"
width="40%"><tr><td >Examination</td><td >University/Board</td>
<td >Year of Passing</td> <td >Division/Marks</td></tr>
<tr><td >10th</td><td >
<input type="text" name="10_univ" size="20" maxlength="50"></td><td >
<input type="text" name="10_year" size="20" maxlength="50"></td><td >
<input type="text" name="10_div" size="20" maxlength="50"></td></tr>
<tr><td>12th</td><td >
<input type="text" name="12_univ" size="20" maxlength="50"></td><td >
<input type="text" name="12_year" size="20" maxlength="50"></td><td >
<input type="text" name="12_div" size="20" maxlength="50"></td></tr></table>
</td></tr><tr>
<td>Other Information<textarea  rows="3" cols="40">Enter text
here.....</textarea></td></tr><tr><td colspan="2"><input type="submit" value="Submit"
></td></tr></table></form>
<footer><hr>&copy; 2013 IGNOU </footer>
<p align="right"><a href="#top" >Return to Top</a></p></body></html>
```

The table tag is used in conjunction with <tr>, <td> tag for creating row and column. The border attribute is used to specify border of the table. The value "0" indicates the table is displayed without border. The <td> element is used to create a table cell. The table cell can span across more than one column or row by using the rowspan and colspan attributes. The <select> element is used with option group/option element to create drop down or list box. When the value of attribute type of <input> element is text, it create a text box for users input and when the value is submitted, it creates a submit button for submitting page. The <textarea> element is used for creation of multiple lines of text box for entering some other information which are not included in the form. The rows and cols attributes of <textarea> is used to create a larger section for input data.

Output of the above program is as follows:



Check Your Progress 3

1. What do you mean by HTML? Differentiate between HTML and XHTML.

---

2. Write program for header section of the page which include name of IGNOU as heading and menu bar as Unit1, Unit2, Unit3.

---

3. Write an html5 program for highlighted text "I am student of IGNOU".

---

4. Create footer for web page with copyright symbol.

---

5. Write a complete web page including header and footer.

1.7 Summary

Web 2.0 is the popular term for advanced internet technology and applications. It allow to users to actually interact with each other in the form of blogs, social networking site, social media, wikis and many more. An important part of web 2.0 is the social media, which is the way to enable people to create social networks online. This network provides online tools for sending individual messages, photo sharing and online chat. The most famous social networking site is Facebook. Related to social media is social bookmarking site. Wikis are website that allows users to add and modify content. Wiki is the multi-lingual, web-based encyclopaedia Wikipedia.

You have learned more elements and attributes about HTML5. Now, you can able to create a new webpage using this technology.

1.8 Solutions/Answers

Check Your Progress 1

Ans1: Web 2.0 is online communication platforms that facilitate online sharing and interaction. Tools like blogs, podcasts and message boards are common Web 2.0 platforms.

Ans2: Search engines are programs that search web page for given keywords and returns a list of the web pages where the keywords were found. It enables users to search for web pages on the Internet. The search engines are Google, Yahoo, Bing and many more.

Ans3: Social media is type of online media where it provides online conversations between users. It also delivers content but does not allow users to participate in creation or development of the content. It provides a platform where users are interact by sharing photos and videos and commenting on them. For example, YouTube, Flickr.

Ans4: Mashups is a web application that retrieved data from two or more external sources to create entirely new and innovative services. This is a new breed of web based data integration application

Ans5: Web widget is a component of web and it require web browser. On the other side, desktop widget is a small application that resides on computer desktop and does not require web browser.

Check Your Progress 2

Ans1: Every HTML5 program must begins with <!doctype> declaration. It is an instruction to browser about what version of HTML page is written in.

Ans2: SVG is used to create scalable vector graphics. SVG is container element that can have other element to describe the shape within itself.

Ans3:
```
<!DOCTYPE html><html><head><title>BCA:BCS053</title><head><body>
<details open="open"><summary>Name</summary><p>Dr. A.P.J. Kalam</p>
<p>If your browser supports this element, it should allow you to expand and collapse these details.</p></details>
```

Ans4: Canvas is used to create a raster graphics whereas SVG is used to create vector graphics. As you know that the raster graphics is loses quality image when enlarged and vector graphics retains it. Canvas is rectangular area on which you can create something using JavaScript, on the other hand, SVG is container element that can have other element to describe the shape within itself. SVG can have CSS control over it.

Ans5: The aside tag is used to represent text aside from main article. It is appears in bold typeface in box or as a quotation text on the page.

Check Your Progress 3

Ans1: Hypertext Markup Language (HTML) is a language. It consists of a variety of elements called tags, which is used for everything from defining a title to including audio/video clip, from creating a heading to inserting a background image in a web page.

Both languages HTML/XHTML are used for displaying web pages. XHTML is an extended version of HTML. XHTML contains all the html tags that follow the rules of XML because it is a family of XML. The main differences between XHTML and HTML are as follows:

1. XHTML provides a shorthand notation for empty elements. For example, you could use <br/> instead of <br> or </br> which HTML does not.
2. XHTML elements must always be closed whereas closing tags are often optional in HTML.
3. Other differences between XHTML and HTML are case sensitivity. In XHTML, all tags must be lowercase. In HTML, it does not matter whether tags are lowercase, uppercase or a mixture of both.
4. XHTML documents must have one root element.
5. XHTML elements must be properly nested**.**

Ans2:
```
<!DOCTYPE html><html><head><title>BCA:BCS053</title></head><body>
 <header><hgroup><h1>Indira Gandhi National Open University</h2>
 <h2>SOCIS </h2><nav>
<a href="">Unit 1</a>| <a href="">Unit 2</a>| <a href="">Unit 3</a></nav>
</hgroup></header></body></html>
```

Ans3:
```
<!DOCTYPE html><html><head><title>BCA:BCS053</title></head>
<body><p><mark>I am student of IGNOU</mark></p></body></html>
```

Ans4:
```
<!DOCTYPE html><html><head><title>BCA:BCS053</title></head>
<body><footer><hr> &copy; 2013 IGNOU</footer></body></html>
```

Ans5:
```
<!DOCTYPE  html><html><head><title>BCA:BCS053</title></head><body>
 <header><hgroup><h1>Indira Gandhi National Open University</h2>
  <h2>SOCIS </h2><nav>
 <a href="">Unit 1</a>| <a href="">Unit 2</a>| <a href="">Unit 3</a></nav>
 </hgroup></header>
 <p><mark>I am student of IGNOU</mark></p>
 <footer><hr> &copy; 2013 IGNOU</footer>
 </body></html>
```

1.9 Further Readings

1) Beginning Web Programming with HTML, XHTML and CSS by Jon Duckett. Publisher: Wrox.
2) Internet and the World Wide Web by Harvey M. Deitel, Paul J. Deitel
3) Programming the World Wide Web by Robert W. Sebesta. Publisher:Pearson
4) HTML 4 Unleashed by Rick Darnell. Publisher: Techmedia
5) XHTML 1.0 by S. Graham. Publisher: Wiley
6) http://www.w3schools.com

UNIT 2 USING STYLE SHEETS

Structure

2.0 Introduction

In the previous Unit, you were introduced to some of the tags of HTML5. In each tag of HTML, you can define a number of attributes that may define the display characteristics, such as font, color, etc. of the content in the tag. However, this model of putting the display characteristics inside the tags makes a web site inflexible, inconsistent and difficult to change. There was a need of separating content of a web page from its presentation (display in browser). This was made possible by Cascading Style Sheets (CSS). CSS allows separation of content of web pages from their presentation. The content can be part of an HTML page, whereas the presentation related information can be moved to CSS. In fact, a CSS can be used to control the style and layout of multiple html pages all at once. Cascading Style Sheet is a mechanism for adding style such as background color, font color to web pages or documents. The W3C, an organization that oversees the development of web technologies, has released three versions of CSS, namely CSS1 in 1996, CSS2 in 1998 and the latest version CSS3.

This unit introduces you to CSS and their types. It also explains how style sheet can be created and used to control the display of web pages on a browser. An external style sheet is a file that contains only CSS code and it will allow you to change the formatting style in multiple pages at once. This unit provides a complete example to demonstrate this concept. You must use CSS while designing web pages as they allow the web pages - to be portable

across browsers; to have a uniform look and feel; to have simpler design. The Websites that uses CSS are also easier to maintain. Please note that a complete discussion on CSS is beyond the scope of this Unit, you may refer to further readings and the site http://www.w3schools.com for more details on CSS.

2.1 Objectives

At the end of this Unit, you will be able to:

- write CSS rule for different elements of a web page
- link an external CSS to a web page
- control the presentation of text using CSS
- define a box model and how you set different properties for this model
- show how simply changing the CSS can alter the display of a web page

2.2 Cascading Style Sheet

Cascading Style Sheets (CSS) are used to control the presentation of a single or many web pages. In the earlier web pages, the display attributes were put in the particular tags. One of the major problems of such a situation was that if you have to insert same style to number of occurrences of that tag in same or different pages, you have to enter that style in every tag. Not only this was tedious and time consuming but also are difficult to change. But, if you use Cascading style sheet, you can set same formatting style to all such tags or elements using a single rule. This rule can then be applied to all the occurrences of that element.

A style sheet consists of *rules* that specify how formatting should be applied to particular elements in a document. In general, a style sheet contains many such rules. Each CSS rule contains two parts. The first part is *selector* and the other part is *declaration*. The selector indicates which element or elements the declaration applies. In the selector part, the elements can be separated by commas. The declaration specifies formatting properties for the element.

Figure-1 shows you how to use CSS rule within your document:

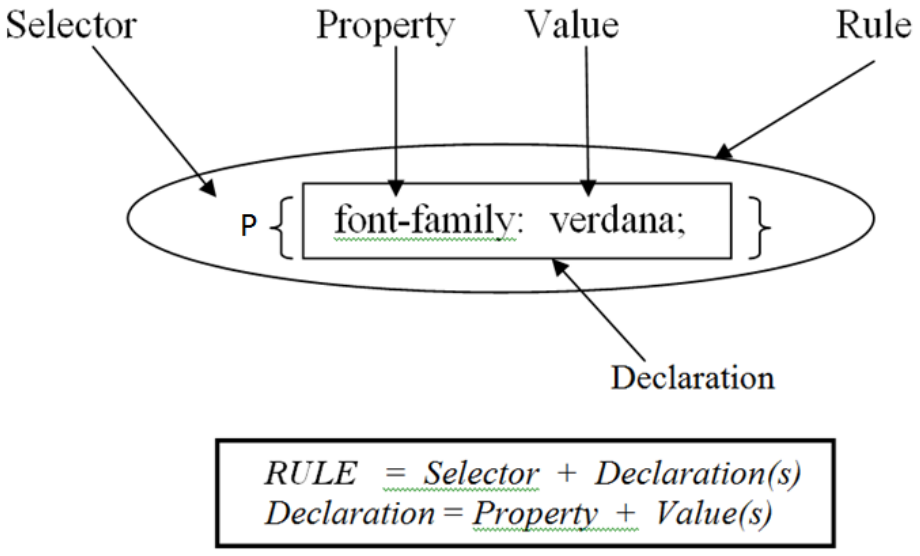


Figure 1: Format of CSS rules

In above example, the rule says that all the paragraphs, that is, <p> elements should appear in Verdana font. Notice that the declaration part is split into two parts namely property and value separated by colon. The declaration part is always written inside the curly brackets { }. Semicolon marks the end of a declaration.

A property defines the property of the selected element(s); in this example the property that is defined is *font-family* property, which is used for defining a set of fonts that may be used for display of an element. A value which specifies the value of this property, in this example, it is *Verdana* fontface is the value.

The CSS rule given in Figure-2 applies to several elements such as <h1>, <h2> and <h3>elements. Note that in the example, selector is h1, h2, h3 (separated by comma). The declaration of the rule specifies several properties for these elements with each property-value pair separated by a semicolon and all properties are kept inside the curly brackets.

```
h1, h2, h3{ font-family: verdana, tahoma, arial; font-weight: bold; font-style:italic;}
```

Figure 2: A CSS rule applicable to several elements

In Figure 2, the font-family property specifies the font for a heading element. The font-family property can hold several fonts. This rule specified that the content of each heading element <h1>, <h2> and <h3> of the related web page will be displayed in bold, italic and Verdana font. If the Veranda font is not installed on the computer then it will look for Tahoma and even if Tahoma is not installed then Arial font will be used.

CSS can be attached to a web page in three different ways as discussed in the next section. More details on CSS properties will be provided in the latter sections.

2.3 Style Sheet Types

In previous section, we have discussed about the CSS rule. A style sheet contains a number of such rules. In this section, you will learn how to use a style sheet for your documents.

There are three different types of style sheets namely Inline Style, Embedded Style and Linked External Style Sheets. The inline style sheet is used to control a single element; embedded style sheet is used to control a single page, while the linked external style sheet is used to control the entire site. This is discussed in more details in the following sub-sections.

2.3.1 Inline Style

Inline style sheet provide the finest level of control. This style sheet is written only for the single element. These style specifications appear within the opening tag of the element.

The style specification format for Inline style sheet are as follows:

```
Style = "property_1: value_1; property_2: value_2; ……;property_n:  value_n;"
```

The above syntax appears as a value of the style attribute of any opening tag. Figure 3 shows an example of an inline style.

```
<!DOCTYPE html>
<html><head></head><body>
<table border="2"><tr><td>Normal TD Cell</td>
<td style="font-family: arial; border-style: solid; background-color: red">
IGNOU:BCS053</td>
</tr></table></body></html>
```

Figure 3: An Inline style

The style given in Figure 3 is application for td tag. It will display cell data in Arial font with background color red and solid border. The output of Figure3 is as follows:

The following Inline Style Sheet of Figure 4 sets the background color of a paragraph. Please note that text color and paragraph elements background color have been specified using hexadecimal numbers.

```
<!DOCTYPE html>
<html>
<head></head>
<body>
<p style="text-transform: capitalize; background-color:#00FFFF; color:#3300CC;">
css examples for inline style sheet</p>
</body></html>
```

Figure 4: Example of Inline style for a paragraph

Output of HTML code of Figure 4:

Using Inline definition, paragraph's data is converted from lowercase to sentence case form style with cyan background and blue text color.

2.3.2 Embedded Style Sheets

This style is placed within the document. For this style sheet, CSS rule can appear inside the <head> element or between the <body> tag contained with a <style> element. The format for this embedded style is as follows:

```
<style type="text/css">…..  rule list ……..</style>
```

The *type* attribute of the <style> element tell the browser the type of style specification which is always *text/css* and rule is written in the following form:

```
Selector {property_1:value_1; property_2:value_2;……;property_n:value_n;}
```

Figure 5 shows an HTML program using Embedded style sheet.

```
<!DOCTYPE html><html><head>
<style type="text/css">
body { background-color: #FF0000; color: #000000;}
 h2, h3 {font-family:  arial, sans-serif; color: #ffffff;}
p { color: #3300FF; font-size:20px;}
</style>
</head><body>
<header> HTML: Version 5.0
<hr><hgroup>
<h2>New features should be based on HTML, CSS and JavaScript</h2>
<h3>Reduce the need for external plugins (like Flash) </h3>
</hgroup></header>
<article>
<p>HTML5 should be device independent</p>
<p>The development process should be visible to the public</p>
</article>
<footer><hr>&copy; 2013 IGNOU</footer>
</body></html>
```

Figure 5: HTML with Embedded stylesheet

Output of the program of Figure 5:

The program in Figure 5 demonstrates how you can override the color of one element by other elements. The color property is defined for body, heading and paragraph elements. The heading is defined within the section heading element named <header> in white color, content of paragraph element is displayed in blue color and footer is in black color. The color of heading and paragraph element overrides the body color. If it is not defined then whole body text is displayed in black color. The <h2> and <h3> are defined within the <hgroup> tag. The <hgroup> tag is used to group heading elements when the heading has multiple levels. The paragraph element is defined within <article> tag. The article tag is used to write text in article form like newspaper and blog. The <hr> element is used to write a horizontal row or line as a separator. The copyright symbol is defined using ampersand sign (‘&’) and copy.

2.3.3 Linking External Style Sheets

As discussed earlier the Inline style are applied within the tag and embedded style apply at document level. In those cases, when style specifications is required in more than one documents then it must be copied in each document. This is not a good practice. Instead, for such cases Linked External Style Sheet can be used. Linked External style exists as a separate file that is linked to a page with the <link> element inside the <head> tag. This file has .css extension and is referenced with URL. Inside the .css file, all style specifications are written which are applicable for the document(s). For linking an external style sheet in a web page, the following command is included:

```
<link rel=”stylesheet”  href=”name_of_file.css”  type=”text/css” >
```

The <*link*> element is used to create a link to css style sheets. The *rel* attribute specifies the relationship between the document containing link and the document being linked to. The *href* attribute specify the URL of the css file.

How to Create an External Style Sheet

External style sheets are created with a similar syntax as document level style sheet.

```
Selector {property_1:value_1; property_2:value_2;…….;property_n:value_n;}
```

Figure 6 contains an external style sheet which can be linked to an HTML document.

```
body { color: #333333;
        background-color: #FFFFFF;  //it indicate white background
        background-image: url(innovation_files/innovation_bg.gif);
      }
  h1 { font-size: 17px;
        color: #ff0000; // red color
        font-family: "Times New Roman", Times, serif;
        font-weight: bold; }
```

Figure 6: An example of External Style Sheet

Save these rules into a text file with the extension .css. In this way, you can write many more elements in .css file. Now, you need to link it to your Web pages.

In example given above, there are defined style sheet rule for only two elements i.e. body and h1. You can define in .css file as many rule as per your requirements. When you include this external sheet in your web page, the heading text is displayed in red bold color and times new roman font with size in 17 pixels. The background-color property is used to display the body color of the web page, in this example, it is white color. The background-image property is used to display an image in background of the body of the page. For this, you can define a path with name of image as a URL of the background-image property.

Precedence of styles

There are several rules that apply to determine the precedence of style sheets. The level of priority depends on the style definition which is closer to the tag. For this order, linked external style sheets have lower priority than embedded style sheets, which are of lower priority than inline style sheets. If the browser did not find any style specification then it uses default property values of its own.

Figure 7 shows a CSS file and related HTML code as well as display of HTML file in a browser window.

CSS File name: ignou.css

```
p{color:blue;}
```

HTML file

```
<!DOCTYPE html>
<html><head>
<link rel="stylesheet"  href="ignou.css"  type="text/css" >
<style>p{ color:green;}</style>
</head><body>
<p style="color:red;">BCA:BCS053</p>
</body></html>
```

Figure 7: An example to show precedence of style sheet

In the above code, style definition is given at inline, embedded and external level for <p> element. The paragraph data 'BCA:BCS053' is displayed in red color. Inline styles override the embedded and external style.

In the above code, if you delete the style attribute from paragraph <p> element, then paragraph data will be display in green color. If you will delete both the styles from inline and embedded level then color of paragraph will be displayed in blue color.

Style Sheet for more specific tags has priority over general tags. For example, if a page has <body> tag with a certain style definition and an <h1> tag with the same properties and different value then the style defined in <h1> tag will have the priority, even though it is also part of the body.

Consider the following example for the above definition. In this case, content of heading will be display in red with light grey background.

```
body{color: #000000; background-color: #FFFFFF; }
h1{color: #FF0000; background-color: #D3D3D3;}
```

For more rules, you may please refer to the website http://www.w3schools.com

Check Your Progress 1

1. Explain the term CSS.

---
2. Define the term ‘Rule’ in respect of CSS.

---
3. What are different ways to apply styles to a Web page?

---
4. What is embedded style sheet? Explain with example.

---
5. What is external Style Sheet? How to link?

2.4 Some Basic Properties in CSS

In the previous sections, you have gone through the basics of style sheet and how to use them along with the web document. This section and next section provides details on some of the properties relating to font, lists, color and alignment of text. A detailed discussion on all the properties is beyond the scope of this Unit. However, you may refer to http://www.w3schools.com for more details on various properties.

2.4.1 Font Properties

The font properties are most commonly used style sheet property. It allows you to control the appearance of text in a web document.

font-family Property:

The font-family property allows you to specify typefaces that should be used for the text. The typeface is the name of font.

For example:

```
font-family: arial, verdana, sans-serif
```

In this case, the browser will use *arial* if it support that font. If not, it will use *verdana* if it supports it. If the browser supports neither arial nor verdana, it will use *sans-serif*. If the browser does not support any of the specified fonts then it will use a default font of its own. If font name has more than one word like Times New Roman, the whole name should be delimited by single quotes as ‘Times New Roman’.

font-size Property:

The font-size property is used to specify a size of the text. You can specify a size value for this property in several ways:

- Absolute Size: xx-small, x-small, small, medium, large, x-large, xx-large
- Length: px, pt, em, cm, mm

- Relative Size: smaller, larger
- Percentage: 2%, 10%, 25%, 50%

You can use any of the above value for this property. The most common unit is px for pixels. The pt (points) may also be used. For other ways you may refer to further readings.

The following example shows font-size property using different units for heading <h1> to <h4>.

```
h1 { font-size: medium; }
h2 { font-size: 18px; }
h3 { font-size: 14pt; }
h4 { font-size: 10%; }
```

font-weight Property:

This property is used to define a degree of boldness of text. Some value for this property is normal, bold, bolder, lighter, 100 and 200.

For example: The following example shows font-weight property using normal and bold as an value for heading element <h4> and <h5>.

```
h4{font-weight: normal ;}
h5{font-weight: bold ;}
```

font-style Property:

The font-style property is used to specify a normal and italic font for the text.

In following example, normal font style is defined for heading 5 and italic font style for heading 6.

```
h5{ font-style: normal; }
h6{ font-style: italic; }
```

font Property:

This font property is used as shorthand to declare all of the font properties at once. Consider the following specifications:

```
<p style="font: bold  italic 12pt verdana">BCS053</p>
```

This specifies that the font weight should be *bold*, font style should be *italic*, font size should be *12 points* in *verdana* typeface.

2.4.2 List Properties

As you know, there are two types of lists namely ordered lists and unordered lists. In ordered lists, the list items are marked with bullets and in unordered lists, list items are marked with numbers or letters.

Using CSS rules, the lists can be more styled and images can be used in place of bullets and numbers as the list item marker. The list properties allow you to control the presentation of list item markers. The different list properties are:

list-style-type Property:

The list-style-type property is used for specifying the 'list style type'. Using this property, you can set different list item marker for ordered and unordered lists.

For example: In the following example, there are two classes are defined i.e. 'a' and 'b' which is used with ordered list and another two classes 'c' and 'd' is used with <li> element of unordered list. When you are used class 'a', the list item of ordered list is displayed in lower-roman form. Using class 'b', list item are marked as upper-alpha. For unordered list, using class c and d, list markers are displayed in square and circle form.

```
<!DOCTYPE html><html><head><style type="text/css">
ol.a {list-style-type: lower-roman ;}
ol.b {list-style-type: upper-alpha ;}
li.c {list-style-type: square ;}
li.d {list-style-type: circle ;}
</style></head>
<body><h3>List style type property for ordered list : lower roman</h3>
<ol class="a"><li>Maruti Zen</li>
<li>Maruti Wagan R</li></ol>
<h3>List style type property for ordered list : upper alpha</h3>
<ol class="b"><li>Maruti Dezire</li>
<li>Maruti Alto</li></ol>
<h3>List style type property for Unordered list : square & circle</h3>
<ul><li  class="c">Honda Activa</li>
<li  class="d">Honda DIO</li></ol></body></html>
```

Figure 8: Example for unordered lists using classes

Output of the above program is Figure 8:

List style type property for ordered list : lower roman

i. Maruti Zen
ii. Maruti Wagan R

List style type property for ordered list : upper alpha

A. Maruti Dezire
B. Maruti Alto

List style type property for Unordered list : square & circle

- Honda Activa
- Honda DIO

list-style-position Property:

The list-style-position property indicates position of the list item marker. This property contains two values: inside and outside. The inside value places the marker inside the display

box and outside places them outside the box. By default, the value of list-style-position property is outside.

The following example shows that the list style positions for ordered and unordered list. For unordered list, list style position of item marker are placed inside the box and for ordered list, position of item marker are outside the box.

```
ul {list-style-position: inside ;}
ol {list-style-position: outside ;}
```

list-style-image Property:

Using this property, you can specify an image as list item marker for unordered list. This property takes precedence over the type of marker specified by the *list-style-type* property.

For example: When you are applied the following rule in your page, list item marker are displayed as an image for unordered list.

```
ul { list-style-image: url("Img.gif"); }
```

list-style Property:

This property is used to set all the properties of list in one declaration.

For example:

```
ul { list-style: circle  inside  url("Img.gif"); }
ol { list-style: upper-roman  outside ; }
```

2.4.3 Color Property Value Forms

The style definition includes different property values in different categories such as colors, fonts, list, alignments of text and many more. The complete property values for all the categories are beyond the scope of this Unit. You can refer to http://www.w3schools.com for complete details. In this section, you will be introduced to one of the most used property value – color value. You can specify a color property as color names such as Red, Yellow etc or a six-digit hexadecimal number with 2 hex digits each for Red Green Blue (RGB) or using RGB form.

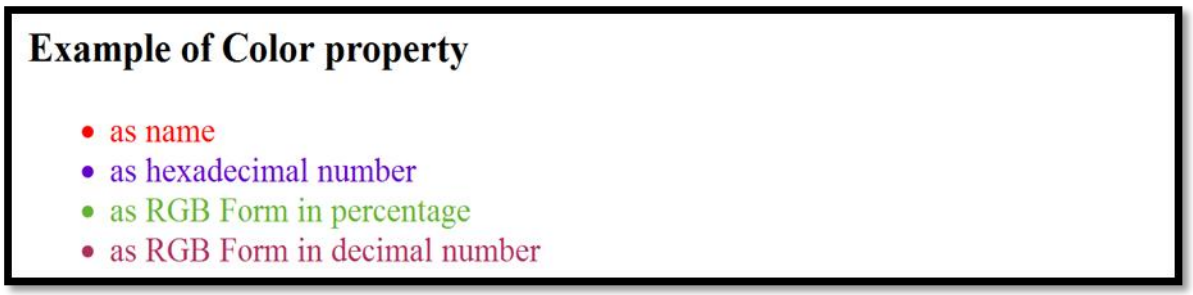
Besides the standard color names, there are a wide variety of names used for colors. For example, Blue color has many shades with different names i.e. light blue, dark blue. Some web browsers may not recognize these names. The most reliable way to specify a color in style sheets is hexadecimal code. This is because any Web browser, even an old one, recognises the hexadecimal notation as a color code. The hexadecimal notation is preceded by hash character '#'. As the base of hexadecimal is 16, you could use any one of 16^6 = 16,777,216 values for a color. In RGB form, you can write *rgb* followed by decimal number between 0 to 255 or as percentages within parenthesis. Here are the depictions of different form of color notation:

```
<!DOCTYPE html>
<html><head><title>BCA:BCS-053</title>
</head><body>
<h3> Example of Color property</h3>
<ul><li style="color:red"> as name</li>
<li style="color:#6600CC"> as hexadecimal number</li>
<li  style="color:rgb(40%,70%,20%)"> as RGB Form in percentage</li>
<li  style="color:rgb(176,48,96)"> as RGB Form in decimal number</li>
</ul></body></html>
```

Figure 9: HTML code specifying colors

Output:

Figure 9 uses the color property using inline style sheet for unordered list. The color code is defined in different forms such as simple color name, hexadecimal number and RGB form.



2.4.4 Alignment of Text

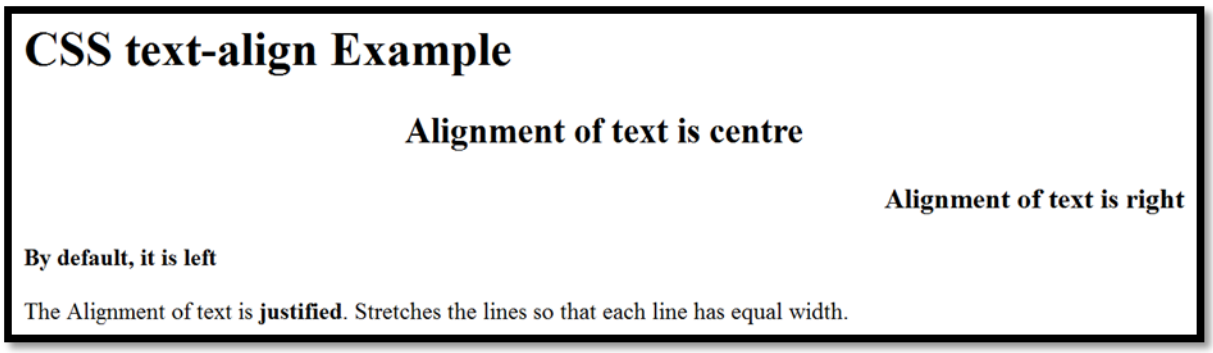
The ***text-align*** property specifies the horizontal alignment of text in an element. By default, it is left. You can define the alignment of the text by the following manner:

For example: The following program contains the different form of text alignment.

```
<!DOCTYPE html>
<html><head><title>BCA:BCS-053</title><style>
h2 {text-align:center}
h3 {text-align:right}
p {text-align:justify}
</style></head><body>
<h1>CSS text-align Example</h1>
<h2> Alignment of text is centre</h2>
<h3>Alignment of text is right</h3>
<b>By default, it is left</b>
<p>The Alignment of text is <span style="font-weight:bold;"> justified</span>.
stretches the lines so that each line has equal width. </p>
</body></html>
```

Figure 10: Styles of Text Alignment

Output of the above program:



The above CSS text-align property example shows the all forms of text alignment such as left, right, centre and justify.

2.5 Selector Forms

As the names indicate that the selectors are used to 'select' element or group of elements on an HTML page. In other words, you can say that selector is a pattern which can apply to different element(s). By using the selector, you can select elements in several ways. The general form of selector is as follows:

```
Selector{property: value;}
```

Here are depictions of the different types of selectors:

2.5.1 Simple Selector

The simple selector form is used for a single element, such as <p> element. It is a normal form of writing CSS rule. In this case, rule is applied to all the occurrences of named element. For example,

```
p {font-weight: bold; color:#336633}
```

In the above case, rule is written for all the <p> elements in a page. The text is displayed in a bold and green color. The following rule is described for grouped selectors in which rule is written once but it apply to more than one selector. This is way of shorthand writing of CSS rule. The rule is applied to both the heading elements.

```
h1, h2 {font-family: arial, book old style, book antiqua;}
```

2.5.2 Class Selector

Class selector is used to apply same style specifications to the content of more than one kind of elements with HTML class attribute. In style specification definition, the class attribute is defined with a name which is preceded with a period. Then, in the body of pages, you refer to these attribute name to activate the element tag. The general syntax for a Class selector is:

```
.ClassSelectorName {Property:Value;}
```

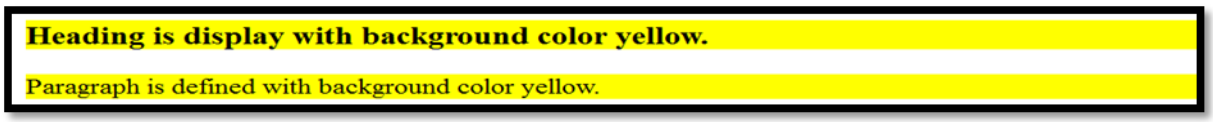
For example shows a simple class selector namely *bgcolor_tag*:

Source code for class selector:

```
<!DOCTYPE  html>
<HTML><HEAD><title>IGNOU:BCS-053</title>
<style type="text/css">
.bgcolor_tag{background-color:yellow;}
</style>
</HEAD><BODY>
<h3 class="bgcolor_tag">Heading is display with background color yellow.</h3>
<p class="bgcolor_tag">Paragraph is defined with background color yellow.</p>
</BODY></HTML>
```

Figure 11: Use of Class selector

Result:



In Figure 11, same class selector is used for heading and paragraph element. You have seen both the elements <h3> and <p> have background-color in yellow. Class selector is also used when you want to specify the style definition to a specific element then you can use element name followed by dot and class selector name. Consider the following example,

```
<!DOCTYPE  html>
<HTML><HEAD><title>IGNOU:BCS-053</title>
<style type="text/css">
p.format {color:red; text-decoration:underline;font-size:20px; font-family:algerian; }
</style></HEAD>
<BODY>
<p>This is normal paragraph</p>
<p class="format">This paragraph is defined with style specification</p>
</BODY></HTML>
```

Figure 12: Use of Class selector for a specific element

The output of the above program is:

2.5.3 id Selector

The id selector is used to select any element in an html page with specific or unique id. In style definition, id attribute is defined with hash character "#". ID selectors are similar to class selectors. The difference between an ID and a class is that an ID can be used to identify one element, whereas a class can be used to identify more than one element. The general syntax for an ID selector is:

```
#idSelectorName{Property:Value;}
```

For example:

Source code for id selector:

```
<!DOCTYPE html>
<html><head><style>
#para1
{
background-color:yellow;
color:red; font-weight:bold;
text-transform:uppercase;
}
</style></head>
<body><p id="para1">This is an example of id selector.</p>
</body></html>
```

Figure 13: Use of ID Selector

Result: The paragraph text is displayed in red color with yellow background and text-transform property is converted the lowercase letter to uppercase.

Check Your Progress 2

1. What are the different possible values in text-align property?

---

2. What is font-family? How you can define font-family property for the <p> element in Inline style definition?

---

3. Create an HTML document to display the text in the title bar of the browser. "Welcome to 5th semester".

---

4. Create a page using BOLD, ITALICS, SMALL, SUPERSCRIPT, SUBSCRIPT, UNDERLINE element.

---
5. Explain different forms of color notation in External Style Sheet definition.

2.6 The Box Model

In a document, each element is represented as a rectangular box. In CSS, each of these rectangular boxes is described using the box model. Each box has four edges: the **margin** edge, **border** edge, **padding** edge, and **content** edge. Each edge may be broken down into a top, right, bottom, and left edge. This model is shown in the Figure 14.

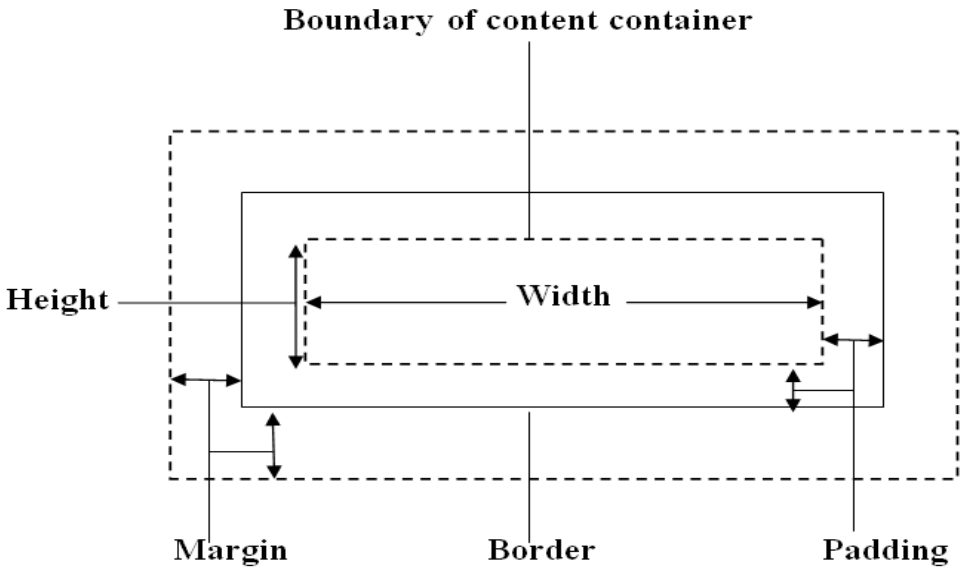


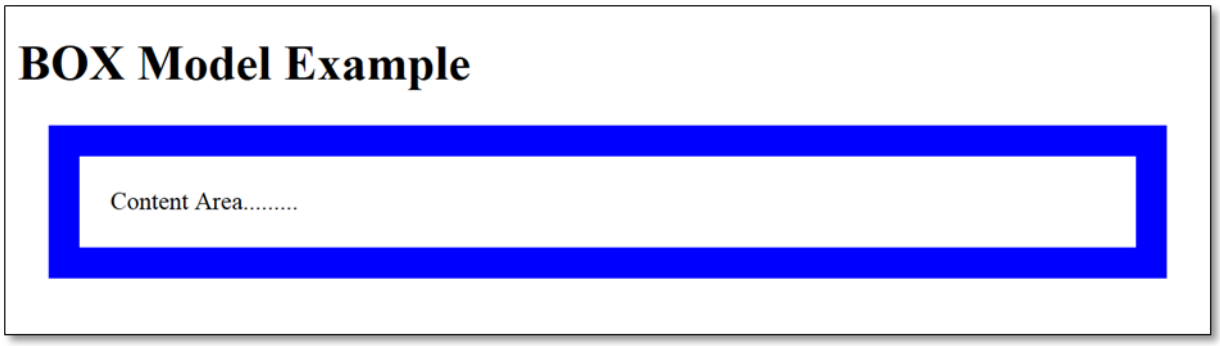
Figure 14: The Box Model of HTML

The innermost area of box model is content area where the text and images appears. The content area is measured by width and height in terms of pixel and percentages. The padding is the space between the content area of the box and its border which is affected by the background color of the box. The border is surrounding area of padding and content. The border is affected by the background color of the box. Every element has a property, ***border-style***. You can set the four sides of element with ***border-left-style***, ***border-right-style***, ***border-top-style*** and ***border-bottom-style***. The outermost area of box model is margin which is completely invisible. It has no background color. Like border edge, margin edge has the properties ***margin-top***, ***margin-bottom***, ***margin-left***, ***margin-right*** and padding have ***padding-left, padding-right***, ***padding-top*** and ***padding-bottom*** which is applies to all four sides of an element.You can set these properties in following manner for any element:

```
<!DOCTYPE html>
<html><head><title>BCA:BCS-053</title>
<style>
p { background: white;      border: 20px solid blue;
margin-top: 20px;          margin-right: 20px;
margin-bottom: 20px;     margin-left: 20px;
padding-top: 20px;          padding-right: 20px;
padding-bottom: 20px;   padding-left: 20px; }
</style></head><body>
<h1>BOX Model Example</h1>
<p> Content Area.....</p>
</body></html>
```

Figure 14: An example of text style specification using Box Model

Output:



You can set all four values using the following shorthand:

padding: [top] [right] [bottom] [left]

For example :

padding: 0 0 0 0;

p{padding:15px 5px 15px 10px;}

The above style code indicate top padding is 15px, right padding is 5px, bottom padding is 15px and left padding is 10px.

p{padding:10px 5px 15px;}

It shows padding from top(10px), right and left(5px), bottom(15px).

p{padding:20px 15px;} means top and bottom padding are 20px, right and left padding are 15px. In shorthand, you can write p**{padding:15px;}** means all four paddings are 15px.

In the similar way, you can define the margin property also.

2.7 Background Image

Background image, as the name implies, are part of the BACKGROUND of a web page. In HTML page, the most common place to add a background image is the body tag.

The ***background-image*** property is used to place an image in the background of an element. You can set the background image for a page in CSS like this:

```
body {background-image:url('logo.gif');}
```

By default, the image is repeated so it covers the entire element. This repetition is called tiling. Tiling is controlled by the **background-repeat property** which take the value ***repeat***(by default), ***repeat-x***, ***repeat-y*** *and* ***no-repeat***. The repeat-x value means that the

image is to be repeated only horizontally, repeat-y means to be repeating vertically and no-repeat specifies that only one copy of the image is to be displayed. You can also set the position for an image by using the **background-position property** which includes values top, center, left, right and bottom. You can set these values in different combinations i.e. right top, top left, bottom right and top center.

Example Source Code for background image:

```
<!DOCTYPE html>
<html><head><title>BCA:BCS-053</title>
<style>
body
{
background-image:url('logo.jpg');
background-repeat:no-repeat;
background-position: top left;
}
h1{text-align:center}
</style></head><body>
<h1>CSS Background Example</h1></body></html>
```

Figure 15: Specifying Background Image

Output:

In the above example, only one copy of image named 'logo.jpg' will be display in top left corner of the screen. You can also use the shorthand property for the above CSS code:

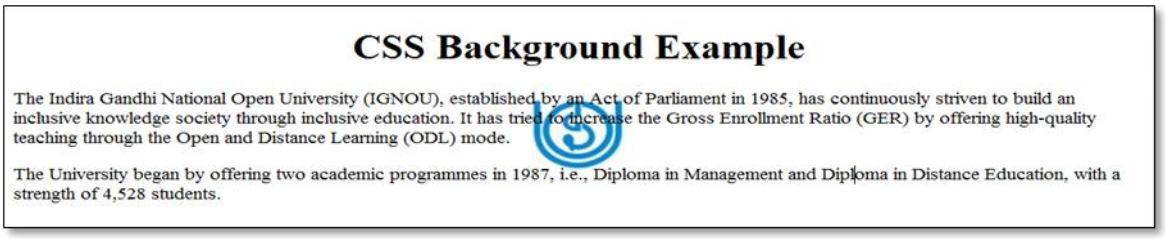
```
body {background: url('logo.gif') no-repeat top left;}
```

Consider the following example for background image:

```
<!DOCTYPE html>
<html><head><title>BCA:BCS-053</title>
<style>
body
{
background-image:url('logo.jpg');
background-repeat:no-repeat;
background-attachment:fixed;
background-position:center;
}
h1{text-align:center}
</style>
</head><body>
<h1>CSS Background Example</h1>
<p> The Indira Gandhi National Open University (IGNOU), established by an Act of Parliament in 1985, has continuously striven to build an inclusive knowledge society through inclusive education. It has tried to increase the Gross Enrollment Ratio (GER) by offering high-quality teaching through the Open and Distance Learning (ODL) mode.
</p>
<p>The University began by offering two academic programmes in 1987, i.e., Diploma in Management and Diploma in Distance Education, with strength of 4,528 students
```

Figure 16: Using Background Image

Output of the above program:



In the above example of background, only one copy of image will be display in the centre of the page. The background-attachment property is used to fix the position of the image. If the length of content is large, then page will be scroll but the image position is fixed in the centre.

2.8 The and Tags

Both the elements and are used to group and structure an HTML page and will often be used together with the selector attributes *class* and *id.*

The **div** (short for Division) elements define logical divisions in a web page. The tag is used as a container for a group of elements. It is not seen on a webpage, but works behind the scene to organize the layout of a webpage a certain way. It acts a lot like a <p> element, by placing new lines before and after the division. A division can have multiple paragraphs in it.

The **span** element has very similar properties to the element, in that it changes the style of the text it encloses. But without any style attributes, the element will not change the enclosed items at all. To use the element, simply surround the text that you want to add styles to with the and tags.

The element is used to define the style of whole sections of the HTML page whereas the element is used to style a part of a text. Div is a block-level element, while span is an inline element. Just to remind you that a block level html element is displayed from a new line like <p>, whereas an inline element is appended to the same line.

For example:

```
<!DOCTYPE html><html><head><title>BCA:BCS-053</title>
<style>
  .format{color:#FF0000; background-color:#99FFCC;}
   #edit{background-color:#9966CC; color:#FFD700;}
</style>
</head><body>
<h2>Div& Span Tags Example</h2>
<div id="edit">
<ul><li><span class="format">Republic Day </span>
of India which is celebrated on 26<sup>th</sup> January. </li>
<li> India gained its independence on
<span class="format">15<sup>th</sup> August 1947</span>. </li>
```

Figure 17: Example of use of and tags

Output of the above example is as follows:

In the above program, the highlighted text '*Republic Day*' and '*15th August 1947'* are displayed in red text color and cyan background by element and rest of the yellow text with purple background by element. Div tags can be used for organising different sections of a web page using different position attributes. A detailed discussion on this is beyond the scope of this Unit. However, we explain the essence of this with the help of example in the next section.

Check Your Progress 3

1. How do you include comments in CSS?

---
2. Explain the box model and its utility with figure.

---
3. Write CSS code for including a background image without repeating in a page?

---
4. What is the difference between and tag?

---
5. Define the CSS padding property.

2.9 USE OF CSS - A Complete Example

In earlier websites, frames and tables were used to organise contents of different web pages. However, div tag and float property provides an alternative to such web page design. The

following example explains such a design in more details. Sometimes it is referred to as table less web layout. It is a method of web design and development without the use of HTML tables for page layout control purposes. Using the DIV-based layout, you can change the entire look of the website by just changing the definitions in the CSS file. The CSS float property is a very important property for layout. You cannot float every element on a Web page. It can be used only for block-level elements such as images (<img>), paragraphs (<p>), divisions ( ) and lists (<ul>).

The float property is used to specify an element to float either left or right in a web page. It has different property values such as left - the element floats to the left and right - the element floats to the right. Another CSS property, clear is used to clear the floated elements. It has three options such as left – clears all left-floated elements, right – clears all right–floated elements and both - clears all floated elements.

In the given example, the web page contains five sections i.e. Header - display a heading, Left – used for menu like navigation, Content - text of the web page, Right – display events details and Footer - for copyright information. The external style sheet is created for this example named CSSLayout.css. Another file main.html is also created for the web page. The CSSLayout.css file is included in main document using link element. Inside the CSSLayout.css file, all style specifications are written which applicable for the main document.

Source Code for CSSLayout.css

```
body{color:#000; font-size:18pt; margin:0; padding: 0; }

.container{width:100%; }

.left{float:left; background:#ffcc00; width:300px; height:500px; padding-left:
20px;}

.right{float:right; background:#ff00cc; width:300px; height:500px; padding-left:
20px;}

.header{background:#ff0000; text-align:center;font-size:30pt; height:100px;}

.content{ float:left;width:600px; text-align:justify;}

.footer{clear:both; background:lightblue; font-size:15pt;}
```

Figure 18: A CSS file to be used with main.html

Please note that .container, .left, .header, .content and .footer are all classes. The property of these classes has been defined in the CSS given above. Please also note how are they refered in the main.html file.

Source Code for main.html

```
<!DOCTYPE html><html><head>
<link type="text/css" rel="stylesheet" href="CSSLayout.css" /></head><body>
<div class="container">
<div class="header">Indira Gandhi National Open University<br>SOCIS</div>
<div class="container">
<div class="left"><menu>
<li><a href="About.html">About Us</a></li>
<li>Programme</li>
<li>Results</li>
<li>Contact Us</li>
</menu></div>
<div class="content"><h3>Table Less DIV Based Layout using Float</h3>
Content within a div tag will take up as much space as it can, namely 100% of
the width of the current location, or the page. So, to affect the location of a
section without forcing it to a fixed width, you can change the padding or the
margin elements. </div>
<div class="right">Event<br>
<details><summary>BCS:053</summary>
<p>Unit 1: Web 2.0 and HTML5<br>
Unit 2: Style Sheets<br>Unit 3:  XML<br>
Unit 4: JavaScript<br>Unit 5: WAP & WML</p></details>
<details><summary>Datesheet</summary>
<p>Datesheet for BCA students</p></details>
<details><summary>Prospectus</summary>
<p>Prospectus for BCA students</p></details>
</div></div>
<div class="footer"><footer><hr>&copy; 2013 SOCIS, IGNOU. Design&
developed by PoonamTrikha</footer></div>
</div></body></html>
```

Figure 19 : HTML File - main.html

The output screen-1 for the above program is as follows:

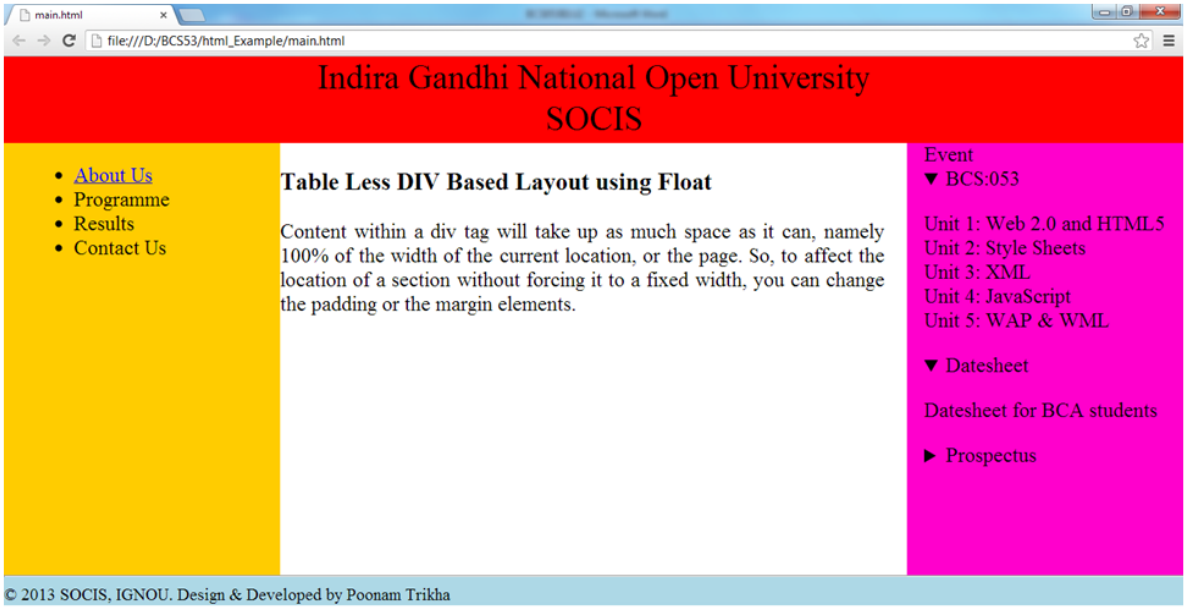


Figure 20: Output of Figure 19 HTML file using CSS of Figure 18

In the above output screen-1, header part shows the heading of the page and left column show the navigation item names such as About Us, Programme, Results and Contact Us. The navigation names are displayed by using the menu element. (You can remove the dots in this list using a CSS property; it is left as an exercise for you.) Please note that each of these navigation elements is pointing to another html file. The centre part is for writing the text of the web page and right column is displayed the events & schedule for the students by using details and summary tags. The <summary> tag defines a visible heading for the <details> element. You can click on the heading to view/hide the details. In this example program, the heading BCS053 and Date sheet is displaying the summary information and prospectus heading hide the details. The second output screen shows the same information regardless the position of the left and right column. For the second program, you can write the same program and just exchange the position of left and right column.

The difference between the following program and the above program is that the position of left column is replaced with the right column and vice versa. You can run the above and following program to see differences. Please note that the program is run only in Google Chrome due to the <summary> and <details> tags.

```
<!DOCTYPE html><html><head>
<link type="text/css" rel="stylesheet" href="CSSLayout.css" />
</head><body>
<div class="container">
<div class="header">Indira Gandhi National Open University<br>
SOCIS</div>
<div class="container">
<div class="left">Event<br>
<details><summary>BCS:053</summary>
<p>Unit 1: Web 2.0 and HTML5<br>Unit 2: Style Sheets
<br>Unit 3:  XML<br>Unit 4: JavaScript
<br>Unit 5: WAP & WML</p></details>

<details><summary>Datesheet</summary>
<p>Datesheet for BCA students</p></details>

<details><summary>Prospectus</summary>
<p>Prospectus for BCA students</p></details>

</div>
```

Figure 21: An HTML program

The output screen-2 for the above program is as follows:

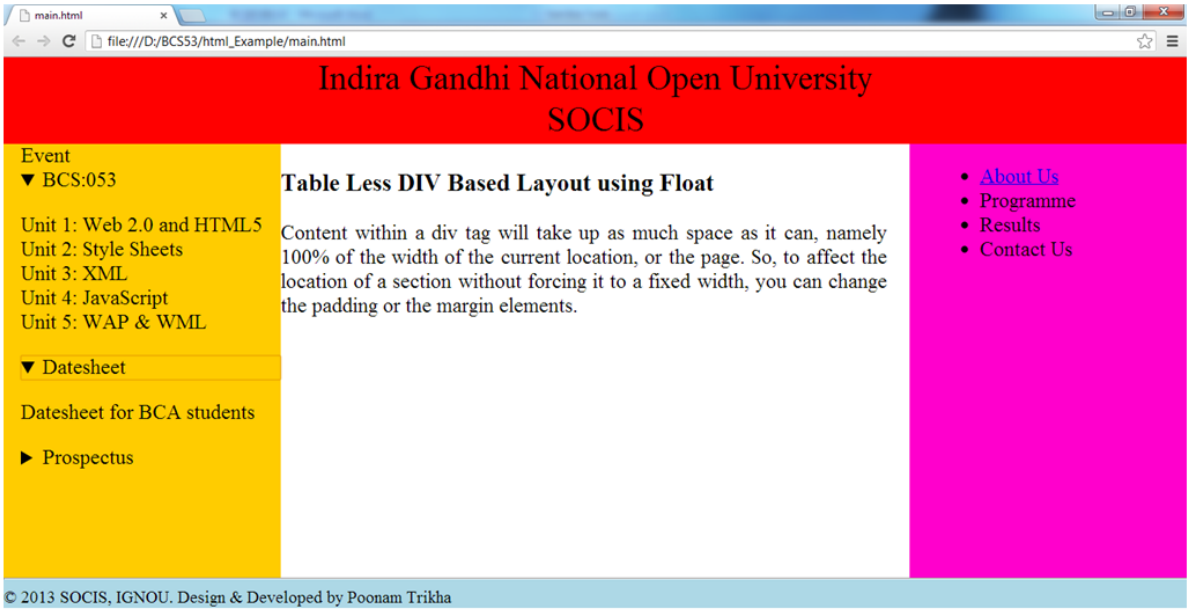


Figure 22: Display of HTML code of Figure 21

2.10 Summary

You have learned more about the Cascading Style Sheet. It is a mechanism for adding styles such as font, color, images in background to web pages. The CSS specifications are maintained by the World Wide Web Consortium (W3C). There are three ways to include CSS code in your web pages viz., inline, embedded and external. In inline style definition, you can include the style definition within the tag. Inline styles are easy and quick to add.

You do not need to create a whole new document as with external style sheets. The inline style sheet has high precedence over the other embedded and external style sheet. But, Inline style is not the best way to design a web page because it can be more difficult to find out where a style definition or rule is being set. Instead of using Inline style, you can use embedded style definition in your page. It is placed within the head section of the page. But the best way to include style in your web pages is external style sheet. In external style sheet, you can write style rules once and it is reflected in all the web pages in your site. You have also learned about the font, list, color property and box model. In CSS, box model is used for the layout of the web page. The div is most of the important element in html because in combination with CSS properties, it can give more effect in web pages. You can use any element in div tag. By using div tag, you can create a block level section in the web page. Each section can have its own formatting. Span is similar to div element. Both divide the content into individual sections. The difference is that span goes into a finer level, so you can span to format a single character if needed. In this unit, there is more exercise for you in form of questions.

You have learned HTML5 (in previous unit) and CSS rules. Now, you can design simple web page using CSS. Try to develop web sites by linking various web pages.

2.11 Solutions/Answers

Check Your Progress 1

Ans.1:
CSS is a web standard that describes style for XML/HTML documents. Cascading style sheet is used to control the layout of multiple pages at once. The CSS code is written in a separate file with extension .css. It is a mechanism to write and modify style definition in css file, and the changes will be reflected in all the web pages in which the css file is linked with link element.

Ans.2:
The CSS rule is a way of writing a code that allows web designers to define styles definition easily and effectively for HTML pages. For example, <p style=”font-weight: bold”>some text</p>.

Ans.3:
There are three ways to insert CSS into web pages by using inline, document or embedded level and external style sheet. In inline style definition, you can include the style definition within the tag. In document level, it is placed within the head section of the page. But the best way to include style in your web pages is external style sheet. In external style sheet, you can write style rules once and it is reflected in all the web pages in your site.

Ans.4: body {background: url('logo.gif’) no-repeat top left;}

Ans.5:
External style sheet is a file containing only style definition code which can be linked with any number of HTML documents. This is the way of formatting the entire site. External style sheet must have extension .css i.e. style.css. The file is linked with HTML documents using the LINK element inside the HEAD element. The following is the way to use external style sheet in a page. <link rel=”stylesheet” href=”name_of_file.css” type=”text/css” >

Check Your Progress 2

Ans.1:
The text-align property is used for horizontal alignment of the text. By default, it is left. Text can either be aligned to the left, right, centred and justify.

Ans.2:
The font-family property specifies the font for an element. The font-family property can hold several font names such as arial, Tahoma and many more. If the browser does not support the first font, it tries the next font and so on.

Ans.3:
<!DOCTYPE html><html><head><title>Welcome to 5th Semester</title>
</head><body></body></html>

Ans.4:<!DOCTYPE html><html><head><title>BCA:BCS053</title></head><body>
This line is BOLD
<u>This line is UNDELINED</u>
This line is ITALICS
<small>The font of this line is small </Small>
H₂o 2³=8 </body></html>

Ans.5:
You can use different color notation for the web page such as color name, hexadecimal form and RGB form. You can use the following program for the color notation. You can write external style sheet for the program as the following named style.css:

```
p.a{color:blue;}
p.b{color:#ff0000;}
p.c{color:rgb(40%,60%,40%);}
p.d{color:rgb(120,60,96);}
```

Now, you can include the above style.css file in the following program:

```
<!DOCTYPE html>
<html><head><title>BCA:BCS-053</title>
<link rel="stylesheet"  href="style.css"  type="text/css" >
</head><body><h3> Example of Color property</h3>
<b><p class="a">Color property as name</p>
<p class="b">Color property as hexadecimal number</p>
<p class="c">Color property as RGB Form in decimal number</p>
<p class="d">Color property as RGB Form in percentage</p></b>
</body></html>
```

Check Your Progress 3

Ans.1:
You can placed anything between /* and */ in CSS is considered as a comment. Comments are ignored by the web browser.

Ans.2:
The term 'box model' is used in CSS-based layout and design. Every element in HTML can be considered as a box, so the box model applies to all html and xhtml elements. Each box has four edges: the margin edge, border edge, padding edge, and content edge. Each edge may be broken down into a top, right, bottom, and left edge.

Ans.3:
body { background-image:url('logo.jpg'); background-repeat:no-repeat;
background-attachment:fixed; background-position:center; }

Ans.4:
The element is used to define the style of whole sections of the HTML page whereas the element is used to style a part of a text. Div is a block-level element, while span is an inline element.

Ans.5:
The CSS padding properties define the space between the element border and the element content. In CSS, it is used to specify different padding to all four sides of an element. For example :

```
padding-top:25px;
padding-bottom:25px;
padding-right:50px;
padding-left:50px;
```

2.12 Further Reading

1) Beginning Web Programming with HTML, XHTML and CSS by Jon Duckett. Publisher: Wrox.
2) HTML & CSS :design and build website by Jon Duckett.
3) Programming the World Wide Web by Robert W. Sebesta. Publisher:Pearson
4) HTML 4 Unleashed by Rick Darnell. Publisher: Techmedia
5) http://www.w3schools.com

Unit 3: Introduction to XML

Structure

3.0 Introduction

XML is widely used for the definition of platform-independent method of storing and processing texts in electronic form. It is also the interchange and communication format used by many applications on the Internet. In this Unit we introduce its basic concepts and attempt to make you independent to use its various applications.

In the first two Units of this Block you have gone through the concepts of HTML and Stylesheets. HTML is a Markup Language that uses standard tags, XML extends those tags to the domain of user defined tags. Therefore, this Unit will help you to go deeper into the aspects of topics covered in Unit1 and Unit 2. To design a cleaner HTML pages, knowledge of XML is useful. This unit will help you to write the valid xml, with formatting features using style sheets, that you have studied in the Unit 2.

3.1 Objectives

- Define the use of XML
- Define the basic terminology used in XML
- Create an XML document
- Create XML Schema and DTD
- Verify XML documents
- Create a simple display style for XML document

3.2 What is XML?

XML stands for eXtensible Markup Language. Very much like HTML, XML is also a markup language. The essence of XML is in its name:

Extensible

It lets you define your own tags. Tags are displayed in the same order as they are defined in your xml document.

Markup

The basic building block of any xml document is its tags, also called as elements. These are very much similar to the ones which are being used at html.

Language

XML is a language similar to HTML, though more flexible giving you option to create your own custom tags. Basically we can call it as meta-language also as it allows us to define or create other languages.

3.2.1 Need for XML

As you know that, HTML is designed to display documents in a web browser. It becomes cumbersome if you want to display the documents in some other language or upgrade the document for dynamic data display. XML is just suited for this purpose. It can be used in a variety of contexts. XML is used to store data in plain text format using various Unicode formats. XML data can be used by various other programming languages. This makes it easy to carry data irrespective of any platform. XML is also used as the base

language for the communication protocols such as XMPP (eXtensible Messaging and Presence Protocol). XMPP protocol, called as Jabber protocol also sometimes, functions with servers to facilitate near real-life operation, and may even allow the internet users to send instant messages (IM) to anyone else on the internet irrespective of their operating systems and browsers.

XML provides users the flexibility to define their own tags. Using meaningful tags you can structure the data, which makes it easier to transport later on and define their document structure

XML, as a document, does not do anything. It is the software which uses this document to connect with the Database and does the processing of data. XML is recommended by W3C as well, in Feb'98 for the purpose of storing and transporting data. XML is so flexible and powerful that it was used for creating other Internet Languages such as XHTML(Extensible HyperText Markup Language), WML(Wireless Markup Language), SMIL(Synchronized Multimedia Integration Language) etc. In HTML, users are supposed to use pre-defined tags only such as <p>, <table>, <b>, <i> etc. thus providing limited flexibility to the users.

Precisely we can say that HTML is a presentation language, and XML is a data-description language.Let's have a look at the sample program below:

```
<?xml version="1.0"?>

<test>

    <text1>My First XML</text1>

    <text2>Let's have coffee after dinner!</text2>

</test>
```

Figure 1: A Simple XML program

Write this text in simple notepad and save it with the extension ".xml". Now, open this file using Internet Explorer (or any other browser) and you will see the output as:

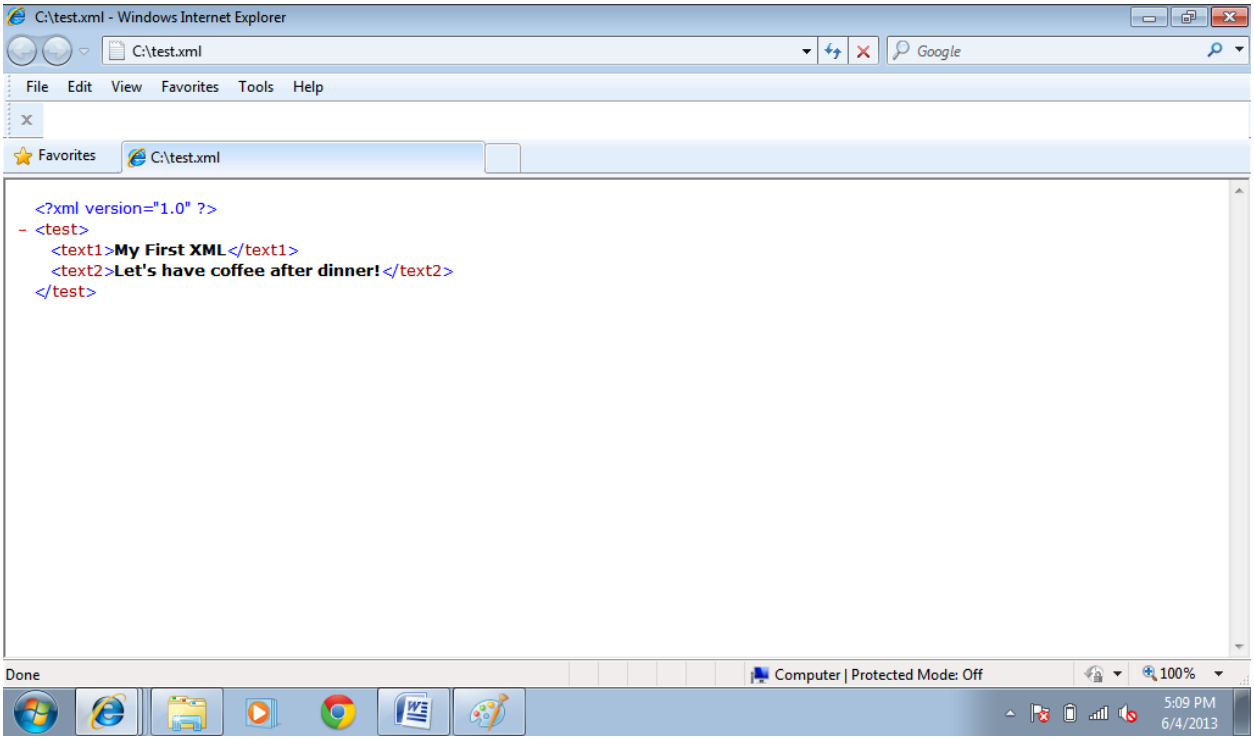


Figure 2: Display of XML program of Figure 1 in a Browser

Unlike in HTML, in XML you have to put an end tag to close open element tag, use similar capitalization in opening and closing tags, and don't use improperly nested tags. Please note that XML tags are case sensitive, that is.<text1> tag is different from <Text1> or <TEXT1> tag. If opening tag is <text1> and closing tag is </Text1>, document will be incorrect.

3.2.2 Various Terms of XML

Let us define various terms used in XML

Tags

A markup construct begins with < and ends with >. Tags are defined as:

- Start tags ; e.g. <test> in Figure 1
- End tags ; e.g. </test> in Figure 1
- Empty-element tags; for example: <line-break /> or
- <car attribute = "value" />

An element with no content is said to be empty. The representation of an empty element is either a start-tag immediately followed by an end-tag, or

an empty-element tag which is place a forward slash before the greater than symbol before the end of the tag i.e. <car />.**Element**

An element is logical part of a document that begins with start tag and ends with a matching end-tag. It can also contain an empty- element tag. The characters between start-tag and end-tag are the element's content, and may also contain other elements which are called as child elements.

For Example:

```
<document>
    <para1>sample text1</para1>
    <para2>sample text2</para2>
</document>
```

Figure 3: An XML tag with child elements

Here, <document> is the element and <para1>, <para2> are the child elements.

Attribute

An attribute is a markup which contains a name-value pair between start-tag and end-tag. For example <car type="SUV">, where type = "SUV" is the name-value pair.

Now, as we discussed earlier that XML is extensible , if we update the XML by adding or removing some tags, it immediately reflects the same on the click of refresh button on the browser. For example, if we update the example of Figure 3 as:

```
<?xml version="1.0"?>
<test>
    <text1>My First extensible XML</text1>
    <text2>Let's have coffee after dinner!</text2>
    <text3>After that, let's go for a long drive.</text3>
</test>
```

Figure 4: Modified version of Figure 3

And it will look like this, upon refresh:

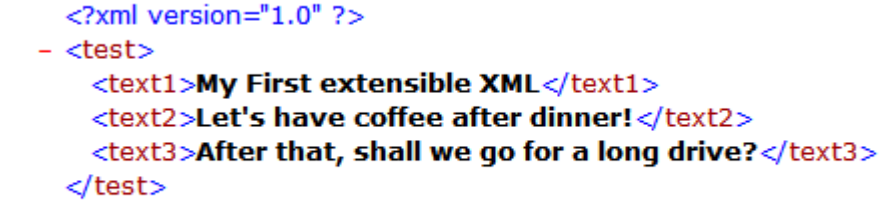


Figure 5: Display of Figure 4.

3.2.3 XML Document Structure

This section explains the document structure of an XML document. XML documents form a tree-like structure that has root and various branches. The following example shows various parts of XML document:

<?xml version="1.0" encoding=”ISO-8859-1”?>

<test>

<text1>My First XML</text1>

<text2>Let's have coffee after dinner!</text2>

</test>

Figure 6: Components of XML document

- The First line is the XML declaration. This line identifies the XML version and the encoding used. In Figure 6, XML version is 1.0 and the encoding is ISO-8859-1, a standard coding close to ASCII.
- The next line is the root element of the document. It describes that this document defines a document structure named test.
- Next 2 lines contain the 2 child elements of the root element test.
- The last line describes the end of the root element.
- Comments can be written in XML document in the same way as are written in an HTML document. For example, <!—This is a comment-->

XML ELEMENTS

All the XML documents must have the ROOT Element. An XML document can be described as a tree. The ROOT element marks the beginning and end of the document tree. Each XML document may have the nested level of the sub-elements, attributes, text. For Example:

```
<Root>
<Parent>
    <Node1>
        <subNode1></subNode1>
        <subNode2></subNode2>
    </ Node1>
    <Node2>
        <subNode1></subNode1>
            <subNode1.1>Text Here!!!<subNode1.1>
        <subNode2></subNode2>
    </ Node2>
</Parent>
</Root>
```

Figure 7: Tree Structure of XML document

Here < subNode1> and < subNode2> are the sub-elements of <Node1>.

< subNode1.1> is the 2nd level of nesting for <Node2> and,

<subNode1> and <subNode2> are the 1st level of nesting for <Node2>.

Attributes

Attributes are used for storing additional information required to be displayed for any element. Attributes do not store any data as data is

dynamic in nature and it becomes difficult to maintain as a part of the attribute. Elements are the best option for storing data in an XML document.

Another important point is that Attribute values shall always be quoted. For Example:

<Student name=”Ved Prakash Sharma”>

</Student>Here “Ved Prakash Sharma” is the attribute value for the element <Student>. Like this many more attribute values can be added to this element such as age, address, qualifications etc. Adding too much attributes to an element makes it difficult to maintain later on.

Empty-Element tag

Some elements are said to be empty like <br>, <img>. In <img> all information is contained in its attributes.
 doesn’t signify any value or attributes, it just means a line break. For empty elements, we need not specify their closing tags. For example,

<emptyElementTag/>

The / at the end signifies that the element starts and ends here. It is an efficient shortcut method to mark up empty elements.

The following example demonstrates use of elements and attributes:

If we take Automobile as Parent, we may have various elements like cars, trucks, bikes etc.

<Automobile>

<car type = “SUV”>

<name>Scorpio</name>

<make>2010</make>

<price>11,000,00<price>

<car>

<car type=”MUV”>

<name>Xylo</name>

```
        <make>2011</make>
        <price>9,000,00<price>
    </car>
    <car type="SEDAN">
        <name>Honda City</name>
        <make>2010</make>
        <price>9,500,00<price>
    </car>
</Automobile>
```

Figure 8: XML elements having *type* attribute

In this example,

<automobile> is the Root Element.

<car> is the root Element, because it contains other elements.

<car> also has one attribute - "type" having various values such as "SUV", "MUV" & "SEDAN". Based on their attribute values, text contents for the following sub-elements are mentioned:

<name>,<make>, and <price> .

Rules

While designing the XMLs, you shall follow certain rules as mentioned below:

XML Document Syntax Rules:

- XML Tags are case sensitive.
- All XML elements shall have closing tags.
- XML Elements shall be properly nested.
- XML Attribute values must be in quotes.
- Some Special characters have entity reference. It means that few special characters are already used as HTML pre-defined tags. So, if we need to

use those characters, we shall write them in a different way as mentioned below:

&lt;	<	less than
&gt;	>	greater than
&amp;	&	ampersand
&apos;	'	Apostrophe
&quot;	"	quotation mark

Now, while naming the XML documents also, you shall follow certain rules such as:

XML Document Naming Rules:

- Names can contain letters, numbers, and other characters.
- Names cannot start with a number or punctuation character.
- Names cannot start with the letters xml (or XML, or Xml, etc.).
- Names cannot contain spaces.

Other than these, any name can be used, as no words are reserved.

3.2.4 XML Namespaces

XML allows you to define your own documents freely thus providing flexibility and open scope. But it is a strong possibility that, when combining XML contents from different sources, there could be inconsistencies between codes in which the **same element name** may serve very different purposes. For example, if you are creating xml document for a bookstore, your use of <title> tag may be used for book's title. A doctor may use the <title> to track patient's formal titles (like Mr., Mrs., Dr., Ms. etc.) on the medical records. While trying to combine these xmls in one system, problem will arise.

XML namespaces provide a method to avoid element and attribute name conflicts in any XML document. XML namespaces attempt to keep different

semantic usages of the same XML elements separate and unambiguous. In this example, each person could define their own namespace and then prefix the name of their namespace to specific tags:<book:title> is different from <patientrecords:title>.

An XML namespace is declared using the reserved XML pseudo-attribute xmlns or xmlns:prefix, the value of which must be a valid namespace name. You can also use a default namespace, which will be implicit in all the child elements. It has the syntax as follows:

xmlns:prefix=”namespaceURI”

A namespace is named as Uniform Resource identifier (URI), identifying an internet resource uniquely, though it does not point to an actual location of the resource. Uniform Resource Locator (URL) is the most common form of URI used.

Any namespace declaration that is placed in a document’s root element becomes available to all the elements in that document. Please note that namespaces have scope. Namespaces affect the element in which they are declared, as well as all the child elements of that document.

For Example:

```
<root xmlns:h=”http://www.w3.org/ “ xmlns:f="http://www.w3schools.com/">

<h:title>
 <h:name>
  <h:first>Vedant</h:first>
  <h:last>Babu</h:last>
  <h:age>16</h:age>
 </h:name>
</h:title>

<f:title>
 <f:bookname>You can Win!!!</f:bookname>
 <f:price>150</f:price>
 <f:author>Shiv Kheda</f:author>
</f:title>
</root>
```

eference : http://www.w3schools.com

Figure 9: XML document with Namespaces

In this example, the xmlns attribute in the <title> tag give the h: and f: prefixes a qualified namespace. So there will not be any name conflict due to same <title> - element name, both having different contents and meanings.

When a namespace is defined for an element, all child elements with the same prefix are associated with the same namespace. Namespaces can be declared in the elements where they are used or in the XML root element. You can also define default namespaces, which will help you in not prefixing any element in the xml document, e.g.:

<root xmlns=http://www.w3.org/TR/html4/

Reference : http://www.w3schools.com

Check Your Progress 1

1. What is a well-formed XML document?

2. What is a valid XML document?

3. What are the rules for naming XML?

3.3 Validating XML Documents

An XML document can be created by you using your own tags. Therefore, there should be a way by which you can find if the created document is valid that is uses correct tags as defined by you. XML provides two major ways of validating XML documents - Document Type Definitions and XML Schema. For validations or any other processing an in-built XML parser is used. Parser is used for "reading" the xml file, analyzing the markup and getting its contents according to the structure, in a program. Let us explain this by the following example:

Here is the structure of an XML file:

<food>

<soup>tomato</soup>

<vegetable>cabbage</vegetable>

<food>

The data structure is a class, having 2 string variables (soup, vegetable) and you parse the xml with tag values such as :

menuFood obj = new menuFood()

Parse(xml, obj)

It will be interpreted as :

Obj[soup:'tomato'; vegetable:'cabbage']

Let us discuss validations in more details in the next sub sections.

4.3.1 Document Type Definitions(DTD)

A Document Type Definition (DTD) defines the legal building blocks of an XML document. It defines the document structure with a list of elements and attributes. If DTD is supplied with the XML file, then the XML parser will compare the contents of the document with the rules that are set in the DTD. If the document does not conform to the rules specified by the DTD, the parser raises an error and indicates where the processing failed.

A DTD can be declared inline inside an XML document, or as an external reference.

Internal DTD Declaration

If the DTD is declared inside the XML file, it should be wrapped in a DOCTYPE definition with the following syntax:

<!DOCTYPE root-element [element-declarations]>

OR

External DTD Declaration

If the DTD is declared in an external file, it should be wrapped in a DOCTYPE definition with the following syntax:

```
<!DOCTYPE root-element SYSTEM "filename">
```

Example DTD with Internal declaration:

```
<?xml version="1.0"?>
<!DOCTYPE test [
<!ELEMENT test (text1,text2,text3)>
<!ELEMENT text1  (#PCDATA)>
<!ELEMENT text2  (#PCDATA)>
<!ELEMENT text3  (#PCDATA)>
] >
<test>
        <text1>My First extensible XML</text1>
        <text2>Let's have coffee after dinner!</text2>
        <text3>After that, shall we go for a long drive?</text3>
</test>
```

Figure 10: Example of Internal DTD

Now, run this XML in the browser and the result would be displayed like this:

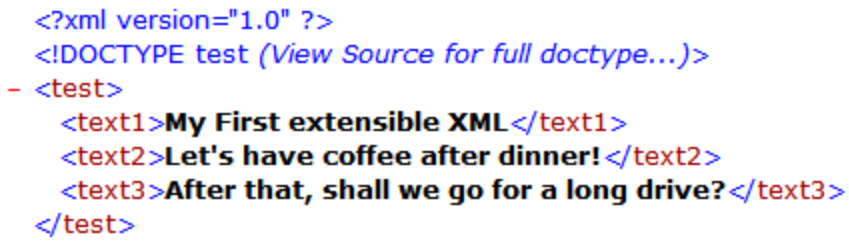


Figure 11: Display of XML file of Figure 10

You can see the source for the doctype definition. The DTD shown in Figure 10 can be interpreted as:

•*!DOCTYPE test* defines that the root element of this document is *test*

•*!ELEMENT test (test1,test2,test3)* defines that the test element contains three elements: "text1,text2,text3"

•*!ELEMENT text1*, *!ELEMENT text2* and *ELEMENT text3* define that text1, text2 and text3 elements are of the type "#PCDATA" (explained later)

The following example demonstrates the use of external DTD: <?xml version="1.0"?>

<!DOCTYPE test SYSTEM "test.dtd">

<test>

<text1>My First extensible XML</text1>

<text2>Let's have coffee after dinner!</text2>

<text3>After that, shall we go for a long drive?</text3>

</test>

The test.dtd is :

<!ELEMENT test (text1,text2,text3)>

<!ELEMENT text1 (#PCDATA)>

<!ELEMENT text2 (#PCDATA)>

<!ELEMENT text3 (#PCDATA)>

Figure 12: XML document with External DTD

DTD defines the main building block of any XML document.

Please note the use of term PCDATA in Figure 10 and 12. It means **parsed character data**. PCDATA is text that WILL be parsed by a parser. The text will be examined by the parser for entities and markup. Tags inside the text will be treated as markup and entities will be expanded.

However, parsed character data should not contain any &, <, or > characters; these need to be represented by the & < and > entities, respectively.

Some time you will find the term CDATA. It means character data. Think of character data as the text found between the start tag and the end tag of an XML element. CDATA is text that will NOT be parsed by a parser. Tags inside the text will NOT be treated as markup and entities will not be expanded.

Attribute Declaration in DTD

Attributes are declared with an ATTLIST declaration.

Syntax:

<**!ATTLIST element-name attribute-name attribute-type default-value**>

For Example, in DTD :

<!ATTLIST car type CDATA “SUV” >

In XML,

<car type= “SUV”></car>

The default value of an attribute can be:

REQUIRED

IMPLIED

EMPTY

DTD:

```
<!ELEMENT car EMPTY>
<!ATTLIST car price CDATA "100000">
```

Valid XML:

```
<car price ="1000000" />
```

In this example, the ”car” element is defined to be empty element with price attribute of TYPE CDATA . If no price is specified, it has default value of 1 L.

#REQUIRED

DTD:

```
<!ELEMENT car >
<!ATTLIST car price CDATA #REQUIRED>
```

Valid XML:

```
<car price ="1000000" />
```

Here “car’ element is defined WITH REQUIRED attribute value. If no price is specified, XML parser will throw an error. So, user is forced to specify the value.

#IMPLIED

DTD:

```
<!ELEMENT car >
<!ATTLIST car price CDATA #IMPLIED>
```

Valid XML:

```
<car price ="1000000" />
```

Here “car’ element is defined with IMPLIED attribute value. It means user is not forced to specify any price value nor any default value is available for this.

You can also used FIXED keyword in the attributes. The following is the syntax of this element.

Syntax

```
<!ATTLIST element-name attribute-name attribute-type #FIXED "value">
```

Example

DTD:

```
<!ATTLIST car manufacturer CDATA #FIXED "Mahindra">
```

Valid XML:

```
< car manufacturer ="Mahindra" />
```

Invalid XML:

```
< car manufacturer ="Hyundai" />
```

Use the #FIXED keyword when you want an attribute to have a fixed value without allowing the author to change it. If an author includes another value, the XML parser will return an error.

Benefits of using DTD:

1. DTDs allow the declaration of standard data types for publishing characters.
2. DTD support is ubiquitous due to its inclusion in XML 1.0
3. DTDs are concise compared to element-based schema and thus present more information in a single screen.
4. DTDs define a document type rather than those defined by the namespaces. It helps in grouping all document constraints in a single collection.

Limitations of DTDs:

1. They have no explicit support for namespaces. All DTD declarations are global, so you cannot define two different elements with the same name, even if they appear in different contexts.

2. DTDs have support only for simple data types.
3. DTDs are not easy to read.
4. DTDs use *regular expression syntax* to describe schema. It means DTD notations are not in XML syntax, so these are not parsed or validated the way an XML document is being treated.

3.3.2 XML Schemas

XML Schema is described as successor to DTD. It also describes the structure of an XML document, and so is also referred to as XML schema definition (XSD). XSDs are more powerful than DTDs in describing XML language. They use a rich data typing system and allow for more detailed constraints on an XML document‘s logical structure. The great strength about schemas is that they are written in XML. So, if you know XML, you may be able to write XML schemas.

The XML schema defines:

- Elements and attributes of an XML document.
- Data types for elements and attributes
- Default or fixed values for elements and attributes
- the child elements
- the number and order of child elements
- empty or non-empty elements

XML Schemas are eXtensible, because they are written in XML. So they can be used in:

- reusing schemas in other schemas
- creating your own data type derived from the standard type
- reference multiple schemas from the same document.

XMLs schemas also support data types, so with this support it is simple to:

- describe document content
- validate data correctness
- work with databases
- define data facets

- define data pattern
- convert data between different data types

With XML schemas, it is easier to send or receive data using defined convention. For example, a date is always misinterpreted in different countries based on the date format they follow. For example, sender country may follow mm/dd/yyyy for date, whereas the receiver country is using dd/mm/yyyy. In such a case a date like 06th August, 2013 may be sent as 08/06/2013 and may be interpreted at receiver as 08th June, 2013. In XML, you can define date as :

<Date type = "date">2013-08-06</date>

Here XML date data type requires the format as "yyyy-mm-dd", thus avoid any chances of confusion.

XML Schema has a lot of built-in data types. The most common types are:

- xs:string
- xs:decimal
- xs:integer
- xs:Boolean
- xs:date
- xs:time

Here are some examples of XML elements:

```
<name>Charlie</name>

<age>36</age>

<dob>1977-08-06</dob>
```

And here are the corresponding simple element definitions:

```
<xs:element name="name" type="xs:string"/>

<xs:element name="age" type="xs:integer"/>

<xs:element name="dob" type="xs:date"/>
```

Example of use of XSD:

XML documents can refer to a DTD or an XML schema, as per example below:

```
<?xml version="1.0"?>

<test>

    <text1>My First extensible XML</text1>

    <text2>Let's have coffee after dinner!</text2>

    <text3>After that, shall we go for a long drive?</text3>

</test>
```

DTD File is as follows: **test.dtd**

```
<!ELEMENT test (text1,text2,text3)>

<!ELEMENT text1  (#PCDATA)>

<!ELEMENT text2  (#PCDATA)>

<!ELEMENT text3  (#PCDATA)>
```

Now XML Schema file is as follows: test.xsd

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com"
elementFormDefault="qualified">

<xs:element name="test">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="text1" type="xs:string"/>
      <xs:element name="text2" type="xs:string"/>
      <xs:element name="text3" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

In this XSD file, <schema> element is the root element of every XML schema It has following attributes.

```
xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

it says that the data types and elements are from this default namespace.

```
targetNamespace="http://www.w3schools.com"
```

it indicates that the elements defined by this schema (test,text1,text2,text3) come from the "http://www.w3schools.com" namespace.

```
xmlns="http://www.w3schools.com"
```

it indicates that the default namespace is "http://www.w3schools.com"

```
elementFormDefault="qualified"
```

it says that any element declared in this schema, shall be used by only those XML instance document which use the same namespace i.e. "http://www.w3schools.com"

A reference to this XML schema file is in the following XML:

```
<?xml  version="1.0"?>

<test xmlns="http://www.w3schools.com"

    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

    xsi:schemaLocation="http://www.w3schools.com test.xsd">

      <text1>My First extensible XML</text1>

      <text2>Let's have coffee after dinner!</text2>

      <text3>After that, shall we go for a long drive?</text3>

</test>
```

Here following code fragment means:

```
<test xmlns="http://www.w3schools.com">
```

the default namespace is “http://www.w3schools.com”.

This fragment means :

xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance

the elements(test, text1, text2, text3) and data types used in the schema are from the “http://www.w3.org/2001/XMLSchema-instance” namespace. It also specifies that elements and attributes that come from the “http://www.w3.org/2001/XMLSchema-instance” namespace shall be prefixed with xsi:

This fragment means:

xsi:schemaLocation="http://www.w3schools.com test.xsd">

First value is the namespace used for all the elements in the xml file and second value is the location of the xml schema used for that namespace.

XML elements can contain values for various data types available.

Reference : http://www.w3schools.com and http://en.wikipedia.org/

Check Your Progress 2

1. **Which are 2 methods to validate XML files?**
2. **What are the limitations of DTD?**
3. **Write down the default built-in data types for XML schema.**
4. **Use the following XML, and write down DTD and XSD for it:**

```
<?xml version="1.0"?>

    <toyFactory>
        <toyType>Cartoon Characters</toyType>
```

```
        <toyName>McQueen</toyName>
</toyFactory>
```

3.4 Displaying XML files

In the previous section, you have gone through the methods of validating XML document. You must have noticed that an XML document is plain text file just like HTML files and is displayed in a browser, in a tree-like structure. Browser does not contain any information about the way content inside a tag may be displayed. Therefore, to enhance the readability, you can add some style to it. This can be done using css i.e. Cascading Style Sheets. CSS files are applied to HTML as well to enhance the readability. An XML document can be displayed in a web browser using a CSS and a XSLT. The following sections describe them in details:

4.4.1 XML CSS

If a browser supports css and xml, you can use css and the formatted example xml document may be displayed as:

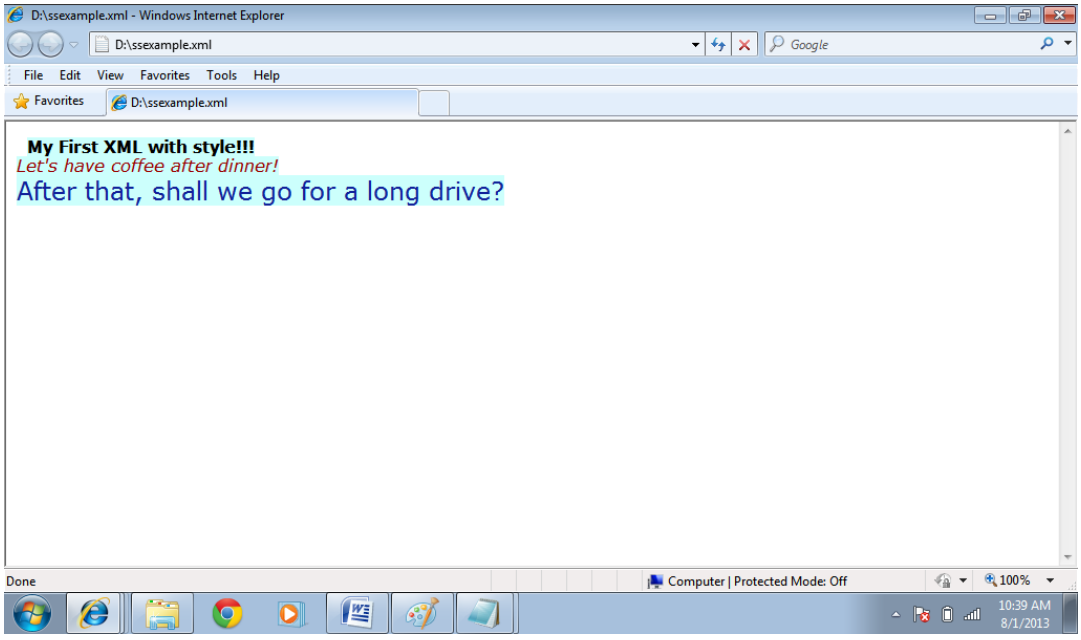


Figure 13: Display of a sample XML page

In this example, please ensure that the xml and css files are stored in the same location on the drive.

Code for the XML file:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/css" href="ss.css"?>

<test>
   <text1>My First XML with style!!!</text1>
   <text2>Let's have coffee after dinner!</text2>
   <text3>After that, shall we go for a long drive?</text3>
</test>
```

Code for the CSS file:

```
test
{
margin:10px;
background-color:#ccfffc;
font-family:verdana,sans-serif;
}

text1
{
display:block;
font-weight:bold;
}

text2
{
display:block;
color:#990000;
font-size:small;
font-style:italic;
}
```

```
text3
{
display:block;
color:#112299;
font-size:large;
}
```

You can enhance the css by adding more formatting features, and that shall reflect on the xml document. Similarly you can also try code using embedded or inline style sheets like you do in HTML. A detailed discussion of this topic is beyond the scope of this Unit. You may refer to further readings for more details on this topic.

3.4.2 XML XSLT

XSLT (eXtensible Stylesheet language Transformations) is the most recommended style sheet language for the XML documents. Using XSLT, XML document can be transformed as:

- another XML document, or
- other web objects such as HTML, plain text etc.
- This way it looks more presentable and it can be converted into PDF, PostScript and PNG files, if required. Original document is never changed, only a new document is created based on the contents of the existing ones.

An XSLT document is a valid XML document and consists of a number of elements/tags/attributes. A transformation can take place in one of the three locations:

- On the server
- On the client (for example, web browser)
- With a standalone program

The following example explains how the transformation takes place on the client, this time instead of css, you will link xml document with the xsl file. The code for the xml file is as follows:

```
<?xml version="1.0" encoding="UTF-8"
standalone="yes"?>
<?xml-stylesheet type="text/xsl" href="transform.xsl"?>

<test>
   <text1>My First XML with style using XSLT!!!</text1>
   <text2>Let's have coffee after dinner!</text2>
   <text3>After that, shall we go for a long drive?</text3>
</test>
```

Now create the xsl file, with the following code:

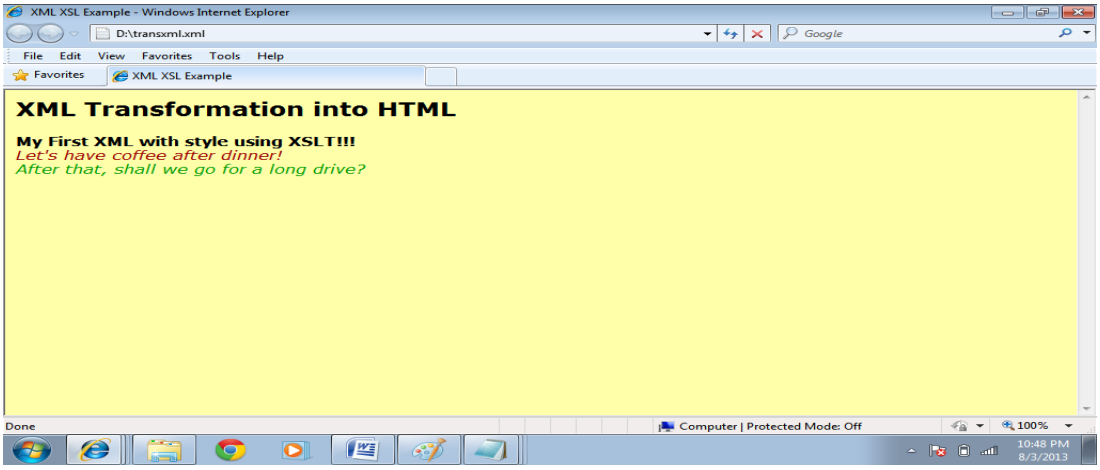
```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"

xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
<html>
<head>
<title>XML XSL Example</title>
<style type="text/css">
body
{
margin:10px;
background-color:#ffffaa;
font-family:verdana,helvetica,sans-serif;
}

.test-text1
{
display:block;
font-weight:bold;
}

.test-text2
{
display:block;
```

```
color:#990000;
font-size:small;
font-style:italic;
}
.test-text3
{
display:block;
color:#009900;
font-size:small;
font-style:italic;
}
</style>
</head>
<body>
<h2>XML Transformation into HTML</h2>
 <xsl:apply-templates/>
</body>
</html>
</xsl:template>

<xsl:template match="test">
  <span class="test-text1"><xsl:value-of select="text1"/></span>
  <span class="test-text2"><xsl:value-of select="text2"/></span>
  <span class="test-text3"><xsl:value-of select="text3"/></span>
</xsl:template>

</xsl:stylesheet>
```

Here <xsl:template> contain rules to apply when specific node is matched. The match attribute is used to associate the template with an XML element. Or to define a template for a whole branch of the XML document (i.e. match="/" defines the whole document). please note that <xsl:template> is a top-level element.

Please note that using style: display:block; makes text display in a newline.

The result shall be displayed like this:



References: 1. http://sitepoint.com

i. 2. http://www.w3.org

Check Your Progress 3

1 Why is style sheet or XSLT is needed?

2 Modify the XML below, as per er given specifications:

Back ground color – RED
Text Color – White
Font – Arial
Add <p> tag with the content as - A very Interesting example!!!
Color for the 3rd tag is blue.

```
<?xml version="1.0" encoding="UTF-8"
standalone="yes"?>
<?xml-stylesheet type="text/xsl" href="transform.xsl"?>

<test>
   <text1>XML formatting with XSLT!!!</text1>
   <text2>Let's have coffee after dinner!</text2>
   <text3>After that, shall we go for a long drive?</text3>
</test>
```

3.5 Summary

In this unit, you have studied a basic introduction of XML and its document structure. XML is a useful language that has immense potential in storing and transmitting data. An XML document needs to be checked, if it is as per some defined tagging format. You have seen how to validate and display any XML document and how are namespaces being used in these. Finally you have gone through the concept of conversion of XML document to a simple HTML document for the purpose of display. Please note that this Unit only introduces you to XML, however, if you want more details on XML, you must refer to some of the web sites specified in the further readings.

3.6 Answers to Check Your Progress

Check Your Progress 1

1. A well-formed document is syntactically correct. It conforms to XML's basic syntax rules:

- Every open tag must be closed.
- The opening and closing tag must exactly match. These tags are case-sensitive.
- All elements must be embedded within a single root element.
- Child tags must be closed before parent tags.

2. A structurally correct document is a valid XML documents. A valid XML document is implicitly well-formed, but well-formed document may not be valid.

3. XML naming rules:

- Names can contain letters, numbers, and other characters
- Names cannot start with a number or punctuation character
- Names cannot start with the letters xml (or XML, or Xml, etc)
- Names cannot contain spaces

Check Your Progress 2

1. DTD and Schema

2. Limitations of DTDs:

   - They have no explicit support for namespaces. All DTD declarations are global, so you can't define two different elements with the same name, even if they appear in different contexts.
   - DTDs have support only for rudimentary data types.
   - DTDs lack readability.
   - DTDs use regular expression syntax to describe schema. This is less accessible to programmers than an element-based syntax.

3. Common built-in data types are as follows:
   - xs:string
   - xs:decimal
   - xs:integer
   - xs:Boolean
   - xs:date
   - xs:time

4. Including XSD, XML file is updated as :

```
<?xml version="1.0"?>

        <toyFactory
xmlns="http://www.w3schools.com/2001/XMLSchema"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"

xsi:noNamespaceSchemaLocation="toyXSDExample.xsd">


        <toyType>Cartoon Characters</toyType>
```

```
        <toyName>McQueen</toyName>

        </toyFactory>
```

And the code for XSD is :

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com"
elementFormDefault="qualified">

<xs:element name="toyFactory">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="toyType" type="xs:string"/>
      <xs:element name="toyName" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

Now, including DTD, same XML file is updated as:

```
<?xml version="1.0"?>
<!DOCTYPE toyFactory SYSTEM "toyDTDExample.dtd">

        <toyFactory>

        <toyType>Cartoon Characters</toyType>
        <toyName>McQueen</toyName>

        </toyFactory>
```

And code for DTD file(toyDTDExample.dtd) is :

```
<!ELEMENT toyFactory (toyType,toyName)>
<!ELEMENT toyType (#PCDATA)>
<!ELEMENT toyName  (#PCDATA)>
```

Check Your Progress 3

1. When we want to display the XML document in a presentable and more readable manner, we prefer to use formatting, which is achieved either by css or XSLT. CSS (Cascading Style Sheet) is used to enhances, formatting features of an XML document, in terms of making few words/lines bold, adding para, newline, spaces etc. XSLT is used to transform the entire XML document itself into a presentable format in the form of another XML or HTML/XHTML document, **2. Code for the XSL file is :**

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
<html>
<head>
<title>XML XSL Example</title>
<style type="text/css">
body
{
margin:10px;
background-color:#990000;
font-family:arial,helvetica,sans-serif;
color:#999999;
}

.test-text1
{
color:#999999;
display:block;
font-weight:bold;
}
```

```
.test-text2
{
display:block;
}
.test-text3
{
display:block;
color:#000099;
font-size:small;
}
</style>
</head>
<body>

<xsl:apply-templates/>
<p>-  A very Interesting example!!!</p>
</body>
</html>
</xsl:template>

<xsl:template match="test">
  <span class="test-text1"><xsl:value-of
select="text1"/></span>
  <span class="test-text2"><xsl:value-of
select="text2"/></span>
  <span class="test-text3"><xsl:value-of
select="text3"/></span>
</xsl:template>

</xsl:stylesheet>
```

Code for the XML file is :

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<?xml-stylesheet type="text/xsl" href="testTransform.xsl"?>

<test>
```

```
    <text1>XML formatting with XSLT!!!</text1>
    <text2>Let's have coffee after dinner!</text2>
    <text3>After that, shall we go for a long drive?</text3>
</test>
```

3.7 References

2. http://www.w3schools.com
3. http://en.wikipedia.org
4. http://sitepoint.com
5. http://www.w3.org

http://searchdomino.techtarget.com

UNIT 4 DOCUMENT OBJECT MODEL

Structure

4.0 INTRODUCTION

In the first two Units of this block you have gone through the concepts of Web 2.0, HTML and CSS. Every web browser window displays a HTML document. The window object represents a window. This window has a document property which refers to a document object. The document object has objects that represent the content of the document. The HTML documents can contain text, hyperlinks, images, forms etc. All this content of the document can be directly accessed and manipulated by JavaScript code. A DOM is an API that defines how to access the objects that compose a document.

"Dynamic HTML" was the immediate ancestor of the Document Object Model. DHTML was thought of largely in terms of browsers. The DOM originated as a specification to allow JavaScript scripts and Java programs to be portable among Web browsers.

The Document Object Model (DOM) is an application programming interface (API) for valid/ well-formed HTML and XML documents. DOM defines the logical structure of documents. It also defines the way to access and manipulate a document.

This Unit explains the Document Object Model in details. It also explains the concepts of events and how JavaScript can be used to handle DOM and events.

4.1 OBJECTIVES

After going through this unit, you should be able to:

- define the Document Object Model
- write JavaScript code to handle HTML DOM
- create documents and navigate through their structures
- handle events using JavaScript
- add, modify and delete elements and content;

4.2 DOM LEVELS

There are three DOM levels:

- The Level 0 DOM – these are by all browsers which are Netscape 2 and onwards.
- The two Intermediate DOMs- these are supported by NS 4 and IE 4 and 5.
- The Level 1 DOM (W3C DOM) – these are supported by Mozilla, IE 5 and above.

"DOM is language and platform independent interface which permits programs and scripts to dynamically access/ update the content, structure, and style of a document."

There are 3 different parts of W3C DOM standard:

- Core DOM
- XML DOM
- HTML DOM

In this unit, we will be discussing about HTML DOM only.

4.2.1 HTML DOM

The HTML DOM is defined as:

- A standard object model for HTML
- A standard programming interface for HTML
- A W3C standard

The HTML DOM defines the **objects** and **properties** of all HTML elements. It also defines the methods required to access HTML elements. In other words "*The HTML DOM is a standard for how to fetch, modify, add, or delete HTML elements.*"

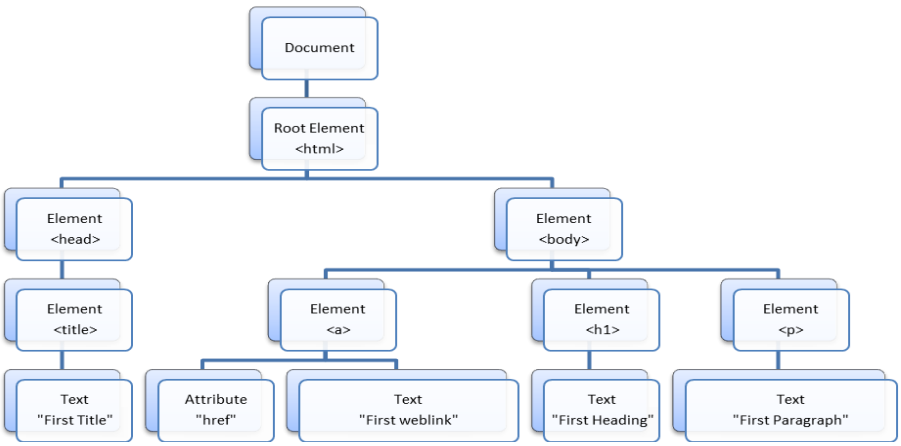
4.2.2 HTML DOM Nodes

According to the W3C HTML DOM standard, everything in an HTML document is a node:

- **document node:** The document node comprises of the whole document.
- **element node :**All HTML element are known as element nodes.
- **text nodes :** The text inside HTML elements is known as the text node.
- **attribute node :** Every HTML attribute is an attribute node.
- **comment nodes :** Every comment in the document is a comment node.

4.2.3 HTML DOM Node Tree

HTML documents are viewed as tree structures by the HTML DOM. The structure is called a **Node Tree.**



With the HTML DOM, all nodes in the tree can be accessed, modified, deleted and created by JavaScript.

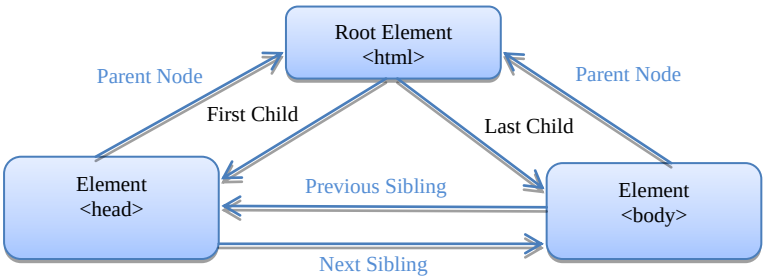
4.2.4 Node Parents, Children and Siblings

The nodes in the node tree have a hierarchical relationship with each other. The node directly above a node is the parent node of that node. The nodes at the same level and same parent are sibling nodes. The nodes that are one level directly below another node are the children of that node. So, to summarize

- The topmost node is called the root node in a node tree.

- Except the root, each node has one parent.
- There is no limit on the number of children, a node can have.
- Sibling nodes are the nodes with the same parent.

The following image illustrates a part of the node tree and the relationship between the nodes:



Look at the following HTML fragment:

```
<html>
        <head>
                <title>Welcome to  IGNOU</title>
        </head>
        <body>
                <h1>Courses Offered</h1>
                <p>Computer Science</p>
        </body>
</html>
```

From the HTML above:

- the root node: The <html> It has no parent node.
- the parent node of the <head> and <body> : <html> node
- The parent node of the "Computer Science" text node : <p> node
- The child nodes of <html> : <head> and <body>
- The child node of <head> : the <title> node.
- The child node of <title> node : the text node "Welcome to IGNOU"
- The siblings and child nodes of <body> : <h1> and <p> nodes
- The first child of the <html> element : <head> element
- The last child of the <html> element : <body> element
- The child of the <body> element : <h1> element
- The last child of the <body> element : <p> element

Check Your Progress 1

1. What is DOM and what are the objectives of DOM?

   ……

   ……

2. What are different levels of DOM?

   ……

   ……

3. Why was the name Document Object Model chosen?

   ……

   ……

4. Describe the HTML DOM with an example?

……

……

4.3 ACCESSING ELEMENT IN JAVASCRIPT

Every HTML element is defined as object. The programming interface comprises of the object methods and object properties.

4.3.1 Method

A **method** is an action, we can take. The action can be to add or modify an element. Refer to table given below for methods used to access an HTML element in different ways:

Method	Purpose
getElementById()	Returns the element with the specified Id
Example: Get the element with id as “university” from the given document. *document.getElementById("university");*	
getElementsByTagName()	Returns all elements with a specified tag name
Example 1: Get a list of all <p> elements in the given document. *document.getElementsByTagName("p");* Example 2: Retrieve a list of all <p> elements that are descendants (children, grand-children, etc.) of the element with id="university" from the given document. *document.getElementById("university").getElementsByTagName("p");*	
getElementByClassName()	returns a list of all elements with a specified class name
Example : Fetch a list of all elements with class="university" from the given document. *document.getElementsByClassName("university");*	

4.3.2 Property

A **property** is a value that we can fetch or set , e.g. it can be the name or content of a node.

Property	Description

innerHTML	• To fetch the content of an element • Useful for getting or replacing the content of HTML element

Following code gets the innerHTML from the <p> element with id="university":

```
< html>
< body>

< p id="university"> Welcome to IGNOU</p>

< script>
var txt=document.getElementById("university").innerHTML;
document.write(txt);
< /script>

< /body>
< /html>
```

nodeName	The nodeName property specifies the name of a node. • nodeName is read-only • nodeName of an element node is the same as the tag name • nodeName of an attribute node is the attribute name • nodeName of a text node is always #text • nodeName of the document node is always #document Note: nodeName always contains the uppercase tag name of an HTML element.

Example: To get the node name of the body element from a given document.

Include the following statement in the script
document.body.nodeName;

The result will be:
BODY

nodeValue	The nodeValue property specifies the value of a node. • nodeValue for element nodes is undefined • nodeValue for text nodes is the text itself • nodeValue for attribute nodes is the attribute value

Following example retrieves the text node value of the <p id="university"> tag from the given document.

```
< html>
< body>
< p id="university"> Welcome to IGNOU </p>
< script type="text/javascript">
```

```
        x=document.getElementById("university");
        document.write(x.firstChild.nodeValue);
< /script>
< /body>
< /html>
```

nodeType	The nodeType property returns the type of node. nodeType is read only. The most important node types are: Element type — NodeType Element — 1 Attribute — 2 Text — 3 Comment — 8 Document — 9

Example: To converts all Text node (descendants excluded) data to uppercase:

```
if(n.nodeType == 3  /*Node.TEXT_NODE*/
   n.data = n.data.toUpperCase();
```

4.4 TRAVERSING A DOM TREE

We can traverse the DOM tree using node relationships in HTML DOM.

4.4.1 Node List

The getElementsByTagName() method returns a **node list**. A node list is an array of nodes.

Example: Select all <p> element nodes in a document and access the second <p>

```
var x=document.getElementsByTagName("p");
y= x[1];          //as the index starts at 0.
```

4.4.2 Node List Length

The “length” property defines the number of nodes in a node-list. We can loop through a node-list by using the “length” property.

Example: Get all <p> element nodes. For each <p> element, output the value of its text node

```
var x=document.getElementsByTagName("p");
for (i=0;i<x.length;i++)  // loop through <p> elements, no. of <p> nodes =x.length
{
        document.write(x[i].innerHTML);    // output the value of text node
        document.write("<br />");
}
```

4.4.3 Node Relationship

We can use the three node properties namely parentNode, firstChild and lastChild to navigate in the document structure. Let us have a look at the following HTML fragment:

```
< html>
< body>

< p>Welcome to IGNOU</p>
< div>
< p>The courses offered by IGNOU</p>
< p>This example demonstrates node relationships.</p>
< /div>

< /body>
< /html>
```

- The first <p> element is the firstChild of the <body> element
- The element is the lastChild of the <body> element
- The <body> element is the parentNode of the first <p> element and the element

Example: This example demonstrates the use of "firstChild" property to access the text of an element. The output of this script will be Welcome to IGNOU, which is the text of the <p> element (the firstChild of the <body> element).

```
< html>
< body>
< p id="university">Welcome to IGNOU</p>
< script>
x=document.getElementById("university");
document.write(x.firstChild.nodeValue);
< /script>
< /body>
< /html>
```

4.4.4 Root Nodes

We can access the full document by using these two special properties :

- document.documentElement - The full document
- document.body - The body of the document

Example

```
< html>
< body>

< p>Welcome to IGNOU</p>
< div>
< p>The Courses offered</p>
< p>This example demonstrates the <b>document.body</b> property.</p>
< /div>

< script>
alert(document.body.innerHTML);
< /script>

< /body>
< /html>
```

4.4.5 childNodes and nodeValue

childNodes and nodeValue properties can also be used to get the content of an element, in addition to the innerHTML property.

The following code gets the value of the <p> element with id="university":

Example

```
< html>
< body>

< p id="university"> Welcome to IGNOU </p>

< script>
var txt=document.getElementById("university").childNodes[0].nodeValue;
document.write(txt);
< /script>

< /body>
< /html>
```

In the example above, getElementById is a method, while childNodes and nodeValue are properties.

4.5 MODIFYING A DOM TREE

The HTML DOM can be modified in following different ways

- By modifying the HTML content
- By modifying the HTML style
- By creating new elements
- By deleting existing elements
- By replacing elements

4.5.1 Modifying HTML Content

We can modify/change the content of an element is by using the **innerHTML** property. It is the easiest way to do so.

Example: This example demonstrates how to change the HTML content of a <p> element. Here the content *Welcome to IGNOU* has been changed to *IGNOU provides innovative Learning!*

```
< html>
< body>
< p id="university">Welcome to IGNOU</p>
< script>
document.getElementById("university").innerHTML="IGNOU provides innovative
Learning!";
< /script>
< /body>
< /html>
```

4.5.2 Modifying HTML Style

With the HTML DOM we can access the style object of HTML elements. In the following example the HTML style of a paragraph has been changed.

```
< html>
< body>
<p id="university">Welcome to IGNOU</p>
<script>
document.getElementById("university").style.color="blue";
< /script>
</body>
< /html>
```

4.5.3 Creating New Elements

To add a new element to the HTML DOM, we must create the element (element node) first, and then append it to an existing element.

Below HTML document contains a element with two child nodes (two <p> elements):

```
< div id="div1">
< p id="p1">This is first paragraph.</p>
< p id="p2">This is second paragraph.</p>
< /div>

<script>
var para=document.createElement("p");  //creates a new <p> element
var node=document.createTextNode("This is new paragraph.");  //create a text node
para.appendChild(node);  //append text node to <p> element

var element=document.getElementById("div1"); // finds an existing element
element.appendChild(para);  // appends new element to existing element
< /script>
```

The appendChild() method in the above example, appended the new element as the last child of the parent.

If we want that the child should be inserted before a specific node and not as the last child, we can use the insertBefore() method:

```
<script>
var para=document.createElement("p");
var node=document.createTextNode("This is new paragraph.");
para.appendChild(node);

var element=document.getElementById("div1");
var child=document.getElementById("p1");
element.insertBefore(para,child);
< /script>
```

4.5.4 Deleting Existing Elements

To delete/ remove an HTML element, we must know the parent of the element. In this example below HTML document contains a element with three child nodes (three<p> elements).

In this script the child node with id as p1 (*This is first paragraph.)* is removed from the parent i.e. element.

```
< div id="div1">
< p id="p1">This is first paragraph.</p>
< p id="p2">This is second paragraph.</p>
<p id="p3">This is third paragraph.</p>
```

```
< /div>
<script>
var parent=document.getElementById("div1"); //find the element with id “div1”
var child=document.getElementById("p1");    //find the <p> element with id “p1”
parent.removeChild(child);                  //remove the child from the parent
< /script>
```

OR

Find the child that is to be removed and use its parentNode property to find the parent. The following code lines also achieve the same.

```
var child=document.getElementById("p1");
child.parentNode.removeChild(child);
```

4.5.5 Replacing Elements

To replace an element to the HTML DOM, we use the replaceChild() method.

In the script below HTML document contains a element with three child nodes (three <p> elements). Here we replace the first child with id as p1 (*This is first paragraph*) with the newly created <p> element *para* having a text node *node* as a child (*This is new paragraph.*).

```
< div id="div1">
< p id="p1">This is first paragraph.</p>
< p id="p2">This is second paragraph.</p>
<p id="p3">This is third paragraph.</p>

< /div>

<script>
var para=document.createElement("p");
var node=document.createTextNode("This is new paragraph.");
para.appendChild(node);

var parent=document.getElementById("div1");
var child=document.getElementById("p1");
parent.replaceChild(para,child); // p1 gets replaced by para.
< /script>
```

4.6 DOM COLLECTIONS & STYLES

The groups of related objects on a page are known as Collections in the Document Object Model. DOM collections are accessed as properties of DOM objects such as the document object or a DOM node. The document object has properties containing the collections of images, links, forms and anchors. These collections comprise of all the elements of the corresponding type on the page. To find the number of elements in the collection, the length property is used. The variable currentLink (a DOM node

representing an a element) has a specialized href property to refer to the link's href attribute. An easy access to all elements of a single type in a page is provided by DOM collections. This is useful for gathering elements into one place and for applying changes in entire page. For example, the forms collection could be used to disable all form inputs after a submit button has been pressed to avoid multiple submissions while the next page loads.

An individual style statement is represented by the Style object. We can access the Style object from the document or from the elements on which that style is applied.

The syntax for using the Style object property is*:*
document.getElementById("id").style.property="value"

Example : The following example changes the background color of the <body> element using style object's *backgroundColor* property when a button is clicked

```
< input type="button" onclick="document.body.style.backgroundColor='yellow';"
value="Change background color" />
```

OR

```
< script>
 function ChangeBackground()
 {
   document.body.style.backgroundColor="yellow";
 }
< /script>

< input type="button" onclick="ChangeBackground()"
value="Change background color" />
```

The Style object property has various categories like: Background, Border/Outline, Generated Content, Text, List, Positioning/Layout Printing, Margin/Padding , Table Misc etc.
For a detailed description on the style object property you can visit http://www.w3schools.com/

An element's style can be changed dynamically. Such a change is often made in response to user events. Such style changes can create many effects, including mouse over effects, interactive menus, and animations. In general, CSS properties are accessed in the format node.style.styleproperty. DOM Style Sheets allow us to step through the rules of each stylesheet, change the selectors, read and write styles, and add new rules. This allows us to create or change CSS that affects several elements at the same time, instead of just one element as with traditional DHTML. It also allows us to take advantage of CSS selectors to target the desired elements, and enter rules into the CSS cascade. DOM stylesheet does not provide an exact copy of what we put in our stylesheet. It produces what the browser sees, and what it interprets. Rules that the browser does not understand are not included . Styles that are not understood are ignored. Comments are not included.

The document.styleSheets collection has all stylesheets available. An easy way to check if a browser supports some amount of DOM Style Sheets is by checking for the existence of the document.styleSheets collection. It can be checked simply by

```
if( document.styleSheets ) {
  // If the browser supports styleSheets,DOM stylesheets will be available.
}
```

Each stylesheet has a number of properties that give details of the stylesheet. These properties can be URL (href), title (title), type (type, usually 'text/css'), the media types it applies to (media), and if the property is disabled or not (disabled). The disabled property can be set to true or false. All other properties are read-only. Only the properties that are applicable to the stylesheet in question are available. For example, if a link element does not have the title attribute set, then its associated StyleSheet object will not have a title.

Check Your Progress 2

1. Write a function that converts all the Text node data of a node and its descendants to uppercase?

   ……

   ……

2. Write a function to find all the tables in a document.

   ……

   ……

3. Explain How to change HTML Attribute using HTML DOM?

   ……

   ……

4. Write a function for searching a specific table in a document and count its rows?

   ……

   ……

5. Write a function to return all name/value pairs of cookies in a document?

   ……

   ……

6. Write a script to toggle the visibility of a para element on the click of a button.

   ……

   ……

4.7 EVENTS

Events are generated by the browser when "things happen" to HTML elements. The HTML DOM allows you to execute code when an event occurs. For example clicking of the mouse, loading of the web page / image, moving the mouse over an element or submission of a HTML form etc. are HTML events.

4.7.1 How Events Work

When events happen to an HTML element in a web page, it checks to see if any event handlers are attached to it. If the answer is yes, it calls them in respective order, while sending along references and further information for each event that occurred. The event handlers then act upon the event. DOM elements can be nested inside each

other. And somehow, the event handler of the parent works even if we click on its child. The reason for this is *event bubbling*. There are two types of event order:

- Event bubbling
- Event capturing

Event bubbling: It begins by checking the target of the event for any attached event handlers, and then bubbles up through each respective parent element until it reaches the HTML element. The main principle of bubbling states that after an event triggers on the deepest possible element, it then triggers on parent s in nesting order.

Event capturing starts with the outer most element in the DOM and works inwards to the HTML element the event took place on and then out again. For example, a click in a web page would first check the HTML element for onclick event handlers, then the body element, and so on, until it reaches the target of the event.

We can choose whether to register an event handler in the capturing or in the bubbling phase. This is done through the addEventListener() method . If its last argument is true the event handler is set for the capturing phase, if it is false the event handler is set for the bubbling phase. Bubbling or capturing can be stopped by event.cancelBubble=true for (IE < 9) and event.stopPropagation() for other browsers. In all browsers, except (IE < 9), there are two stages of event processing .The event first goes down - that's called *capturing*, and then *bubbles* up. This behavior is standardized in W3C specification.

4.7.2 Reacting to Events

A JavaScript can be executed when an event occurs.

Example 1: Content of the <h1> element is changed when a user clicks on it

```
<html>
< body>
< h1 onclick="this.innerHTML='IGNOU'">Click here!</h1>
< /body>
< /html>
```

Example 2: In the following example the javascript is executed on onblur event. When the user leaves an input field the onblur event is triggered and the handler for this event (*changeToUpper()*) gets executed. This function changes the text of the input field to upper case.

```
<html>
<head>
<script>
function changeToUpper(){
var x=document.getElementById("fname");
x.value=x.value.toUpperCase();
}
</script>
</head>
<body>
Enter your name: <input type="text" id="fname" onblur=" changeToUpper()">
<p>When you leave the input field,  the input text will  change  to upper case.</p>
</body>
</html>
```

Example 3: In the following example we change the text of the <p> element from “Let us learn Event Handling” to “I have learnt Event Handling” when a button is clicked.

```
< html>
< body>
<p id="p1">Let us learn Event Handling.</p>
< script>
function ChangeText() {
document.getElementById("p1").innerHTML="I have learnt Event Handling.";
}
< /script>
< input type="button" onclick="ChangeText()" value="Change text">
< /body></html>
```

4.7.3 HTML Event Attributes

To assign events to HTML events we can use event attributes.

Example: Assigning an “onclick” event to a button element

```
<button onclick="displayMonth()">Check it yourself</button>
```

In the example above, a function named *displayMonth* will be executed when the button is clicked.

4.7.4 Assign Events using the HTML DOM

HTML DOM allows us to assign events to HTML elements using JavaScript.

Example: Assigning an “onclick” event to a button element

```
<script>
document.getElementById("myBtn").onclick=function(){displayMonth()};
</script>
```

In the example above, a function named *displayMonth* is assigned to an HTML element with the id=myBtn".

Event	Description
onload onunload	• These events are triggered when the user enter or leaves the page • onload event can be used check browser type/ version and load the proper version of web page • onload and onunload can be used to deal with cookies Example: *<body onload=”checkCookies()”*
Onchange	• often used in combination with validation of input fields Example: *< input type="text" id="fname" onchange="upperCase()">*

onmouseover onmouseout	• These events can be used to trigger a function when the user mouse over, or out of, an HTML element Example: *<html> <body> <h1 onmouseover="style.color='red'"' onmouseout="style.color='blue'"> Mouse over this text</h1> </body> </html>*
onmousedown onmouseup onclick	• onmousedown, onmouseup, and onclick events are all parts of a mouse-click • First when a mouse-button is clicked, the onmousedown event is triggered, then, when the mouse-button is released, the onmouseup event is triggered, finally, when the mouse-click is completed, the onclick event is triggered.

For more information on the events you can visit http://www.w3schools.com/

Check Your Progress 3

1. Write a program to change the HTML element using events?

   ……

   ……

2. What are the functionalities performed by onload() and onUpload()?

   ……

   ……

3. Execute a javascript before the browser closes the document.

   ……

   ……

4. Write a script to alert the keycode of the key pressed.

   ……

   ……

5. Write a function to execute a script that changes the color of an image when moving the mouse over / out of the image?

   ……

   ……

6. Write a function to return the type of event that occurred?

   ……

   ……

4.8 DYNAMIC DOCUMENTS USING JAVASCRIPT

Dynamic Document Creation is the creation of a Web document from within the JavaScript. It may be created while an HTML document is being displayed, and may

be influenced by features of the current Web page. As a technique, it is very useful for displaying information from Web pages and creating new HTML documents.

Let us create dynamic documents with **document.write()**. This method can

- insert text into the current web page
- create an entirely new document

Let us look at this example that uses write to display today's date to an otherwise static HTML document.

```
<script type="text/javascript">
 var today = new Date()
 var mon = today.getMonth() + 1
 document.write(
  "<html><head><title>Dynamic Document</title></head>\n"
  + "<body bgcolor=yellow>\n  <h1 align=center>Today is "
  + mon + "/"  + today.getDate() + "/" + today.getYear()
  + "</h1>\n</body></html>")
</script>
```

We can also use the write method in conjunction with the open() and close() methods of the Document object, to create entirely new documents in other windows or frames. The function given below opens a pop up widow to display the date the document is accessed on. Invoke this function from an event handler.

```
Function recent( ) {
      var w = window.open( );     // Create a new window with no content
      var d = w.document( );      // Get its Document object
      var today = new Date( );
      d.open( );                  //Start a new document
      d.write("<p> Document recently accessed on:  " + today.tostring( ));
      d.close( );                 // Close the document
 }
```

4.9 AJAX

AJAX stands for **Asynchronous JavaScript and XML**. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS and Java Script. It uses XHTML for content and CSS for presentation, as well as the Document Object Model and JavaScript for dynamic content display.

In standard Web applications, the interaction between the customer and the server is synchronous. This means that one has to happen after the other. If a customer clicks a link, the request is sent to the server, which then sends the results back.

With Ajax, the JavaScript that is loaded when the page loads handles most of the basic tasks such as data validation and manipulation, as well as display rendering the Ajax engine handles without a trip to the server. At the same time that it is making display changes for the customer, it is sending data back and forth to the server. But the data transfer is not dependent upon actions of the customer.

The Ajax engine works within the Web browser through JavaScript and the DOM. It renders the Web application and handles any requests of web server's client. While the Ajax engine is handling the requests, it can hold most information in the engine itself. At the same time Ajax engine allows the interaction with the application and the customer to happen **asynchronously** and independently of any interaction with the server. The key feature of Ajax application is that it uses scripted HTTP to communicate with a web server without causing pages to reload. Since the amount of data exchanged is often small and the browser does not have to parse and render a document. As a result, the response time is greatly improved and this results in making web applications feel more like desktop applications.

Here is the list of famous web applications which are using AJAX

Google Maps (http://maps.google.com/)
A user can drag the entire map by using the mouse instead of clicking on a button or something

Google Suggest (http://www.google.com/)
As you type, Google will offer suggestions. Use the arrow keys to navigate the results

Gmail (http://gmail.com/)
Gmail is a new kind of webmail, built on the idea that email can be more intuitive, efficient and useful.

Check Your Progress 4

1. What is AJAX? List out the differences between AJAX and JavaScript.

   ……

   ……

2. What are the advantages of AJAX?

   ……

   ……

3. Name the browsers that support AJAX?

   ……

   ……

4. What are the limitations of Ajax?

   ……

   ……..

4.10 SUMMARY

In this unit we have learnt how to use the HTML DOM to make our web site more dynamic and interactive. We have also learnt the ways to manipulate HTML elements in response of different scenarios and created dynamic web pages by using scripts on the client (in the browser). We got to know about AJAX which is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS and Java Script.

4.11 ANSWERS TO CHECK YOUR PROGRESS

Check your progress 1

1. The Document Object Model (DOM) is a cross-platform and language-independent convention for representing and interacting with objects in HTML, XHTML and XML documents. With the Document Object Model, programmers can build documents, navigate their structure, and add, modify, or delete elements and content.

One important objective for the Document Object Model is to provide a standard programming interface that can be used in a wide variety of environments and applications.

2. There are three DOM levels:

- The Level 0 DOM, supported from Netscape 2 onwards by all browsers.
- The two Intermediate DOMs, supported by Netscape 4 and Explorer 4 and 5.Note that the use of these DOMs is not necessary any more
- The Level 1 DOM, or W3C DOM, supported by Mozilla and Explorer 5

3. The name "Document Object Model" was chosen because it is an "object model" is used in the traditional object oriented design sense: documents are modeled using objects, and the model encompasses not only the structure of a document, but also the behavior of a document and the objects of which it is composed.

4. A standard way to access and manipulate HTML documents is defined in HTML DOM. HTML document is represented as a tree-structure in DOM.

Check your progress 2

1.
```
function uppercase(n) {
  if (n.nodeType == 3)  /* Text Node */
        n.data = n.data.toUpperCase( );
  else{
        var children = n.childNodes;
        /* loop through children, recursively call
        for ( var  i=0; i<children.length; i++)
              uppercase(children[i] );
        }
  }
```

2.
```
var tables = document.getElementByTagname( “table” ) ;
 alert( “ This document contains ( + tables.length + “tables” );
```

3. Example to change HTML Attribute by using HTML DOM

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

```
<img id =”image” src =”oldclip.jpg”>
<script type = “ javascript”>
document.getElementById(“image”).src = “newclip.jpg”
</script>
</body>
</html>
```

In the above example we load an image on HTML document by using id="image". Using DOM we get the element with id="image". This script will change the src attribute from oldclip.gif to newclip.jpg

4.
```
var table secondtable  = document.getElementById ( “ COURSES” );
   var rows = secondtable.getElementByTagname( “ tr “ );
var totalrows = rows.length;
```

5.
```
<html>
 <body>
   Cookies associated with this document:
 <script>
   document.write(document.cookie);
  </script>
 </body>
 </html>
```

6.
```
<html>
  <body>
<p id="p1"> Welcome to IGNOU Welcome to IGNOU Welcome to IGNOU.
Welcome to IGNOU </p>
<input type="button" value="Hide text"
onclick="document.getElementById('p1').style.visibility='hidden'" />
<input type="button" value="Show text"
onclick="document.getElementById('p1').style.visibility='visible'" />
</body>
</html>
```

Check your progress 3

1. Events are usually generated by the browser, when an event on some object takes place like user clicks an element. Event handlers are used to handle the events and execute the code . The example is shown below that is used to change the HTML element values:

```
<html>
<body>
<input type="button" onclick="document.body.bgColor='green';"
value="Change background color" />
</body>
</html>
```

2. onload() and onUpload() are two functions that get activated or the event gets triggered when the user enters or leaves the page. The onload() event is used to verify and check the user’s visit according to the browser’s type and it loads a version of the

event. It provides the web page information that allows easy access to the website and consists of the information regarding the event that is triggered. Onload() and onUpload() events use the cookies to hold down the values given by the users when it enters or leaves the page.

```
3. <html>
<head>
<script>
function beforeclose()
{
alert("Welcome to IGNOU!");
}
</script>
</head>

<body onunload="beforeclose ()">

<h1>Welcome to IGNOU Home Page</h1>
<p>Press F5 to reload the page OR close this window.</p>

</body>
</html>

4. <html>
<head>
<script>
function codeKey(event)
{
alert(event.keyCode);
}

</script>
</head>

<body onkeyup=" codeKey (event)">
<p>Press a key on your keyboard. An alert box will show the keycode of the key
pressed.</p>
</body>

</html>

5.<html>
<head>
<script>
function bgChange(bg)
{
document.body.style.background=bg;
}
</script>
</head>
<body>
<b>Mouse over the squares and the background color will change!</b>
<table width="200" height="100">
 <tr>
  <td onmouseover="bgChange('red')"
     onmouseout="bgChange('transparent')"
     bgcolor="red">
```

```
  </td>
  <td onmouseover="bgChange('blue')"
     onmouseout="bgChange('transparent')"
     bgcolor="blue">
  </td>
  <td onmouseover="bgChange('green')"
     onmouseout="bgChange('transparent')"
     bgcolor="green">
  </td> </tr>
</table>
</body>
</html>
```

```
6.function getTypeOfEvt(event)
{
alert(event.type);
}
</script>
</head>

<body onmousedown=" getTypeOfEvt (event)">

<p>Click in the document.
 The type of event triggered will be shown in an alert box .</p>

</body>
</html>
```

Check your progress 4

1. Ajax is abbreviated as Asynchronous Javascript and XML. It is new technique used to create better, faster and more interactive web systems or applications. Ajax uses asynchronous data transfer between the Browser and the web server. Here on sending request to the server, one needn't wait for the response. Other operations on the page can be carried out Hence, Asynchronous. On the other hand, Java script sends an HTTPRequest to the server and waits for the XML response e.g. populating State field. Using JavaScript we need to use the "Onchange" event where as using ajax, the request is just sent to populate the state list. Other operations can be carried out on the page. Ajax is a part of Java Script programming. Java Script is used to manage and control a web page once downloaded. Ajax does not need to wait for the whole page to download.

Use of Ajax can reduce connections to the server since the script has to be requested once.

2. Following are the advantages of Ajax:

- It saves memory when the data is fetched from the same page. So, bandwidth utilization is better.
- It provides more interactivity.
- Using Ajax data retrieval takes less time.

3. Following browsers support AJAX:

- Internet Explorer 5.0 and above
- Opera 7.6 and above

- Netscape 7.1 and above
- Safari 1.2 and above

4. Limitations of AJAX:

- Back functionality cannot work because the dynamic pages don't register themselves to the browsers history engine
- The page cannot be bookmarked if implemented using Ajax.
- If java script is disabled, Ajax will not work.
- Because different components of the pages are loaded at different times it may create confusion for the user.

4.12 REFERENCES & FURTHER READINGS

http://www.w3schools.com
http://www.w3schools.com/htmldom
http://www.w3.org/
http://www.w3.org/DOM/
http://www.w3.org/TR/
http://www.w3.org/wiki/Handling_events_with_JavaScript
http://simplehtmldom.sourceforge.net/
http://stackoverflow.com/
https://developer.mozilla.org/en/docs/DOM

Deitel, H. M., & J., D. a. (2008). Internet & World Wide Web How to Program, 4/e. *Pearson Education*

Sebesta, R. W. (2011). Programming the World Wide Web, 6/E. *Addison-Wesley / Prentice Hall*

Unit 5: Introduction to WAP and WML

Contents

5.0 Introduction to WAP

Internet's penetration and expansion into fields such as finance, research, medicine, education, business etc. has initiated new ways of providing services to customers and conducting business. **WAP (Wireless Application Protocol)** is simply acommunication protocol by which a wireless device talks to a server installed in a wireless network. It is an open, global specification, which gives mobile users with access to wireless devices the opportunity to easily access the Internet or other computer applications, defined by the WAP forum.

WAP Forum, founded in 1997, is the industry association of carriers, handset manufacturers, infrastructure providers, software developers etc. dedicated to make internet other information services available to handheld wireless devices.

WAP is an emerging industry standard which aims at providing wireless internet services and information access using handheld devices having limited display capabilities and over limited bandwidth wireless channels. How successful this effort has been can be seen by the fact today more than 95% of the global hand set makers are members of the WAP Forum.

5.1WAP Protocol Stack

WAP stack has inherited most of the characteristics of the ISO OSI reference model

It consists of five layers, namely: application layer, session layer, transaction layer, security layer and datagram layer

Application Layer (Wireless Application Environment, WAE)

WAE provides an application environment intended for the development and execution of portable applications and services. WAE User Agents are designed to understand text in encoded WML (Wireless Markup Language) and compiled WMLScript. WAE consists of WML and WMLScript as the two main building blocks of the user agents located on the client side

Session Layer (Wireless Session Protocol, WSP)

WSP supplies methods for the organized exchange of content between client/ server applications.

Transport Layer (Wireless Transport Protocol WTP)

WTP is responsible for providing reliable transmission of WSP data packets between the client and the server over a wireless medium.

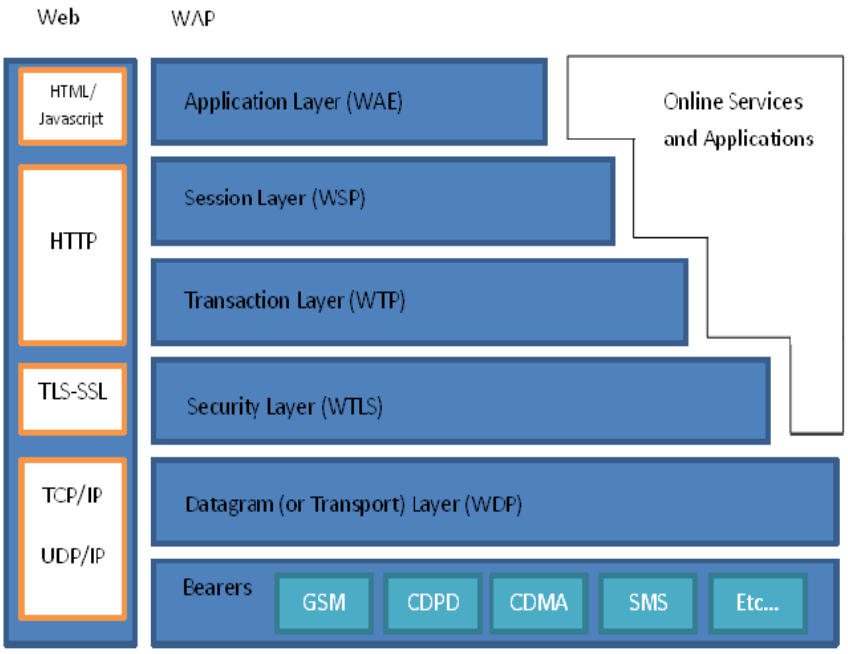
Security Layer (Wireless Transaction Layer Security WTLS)

WTLS is an optional layer and is the solution to the security issue, provided by the WAP Forum. WTLS is based on SSL(Secure Socket Layer)and when present provides services that ensure privacy, server authentication, client authentication and data integrity.

Network Layer (Wireless Datagram Protocol WDP)

WDP is the bottom most layer of the WAP stack. It is modelled after User Datagram Protocol (UDP) and is a datagram oriented protocol.WDP shields the upper layers from the bearer services like SMS, CMD etc. provided by the network and hence allows applications a transparent transmission of data

over different bearers. WDP must have bearer specific implementation since it is the only layer that has to interface various bearer networks.

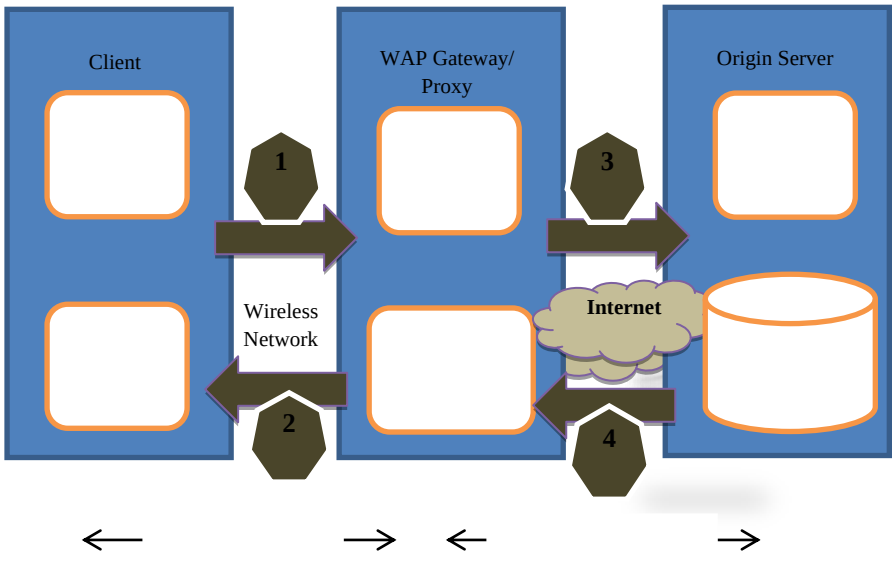


5.1.1 Objectives of WAP

Standard HTML content cannot be effectively displayed on the small-size screens of pocket-sized mobile phones and pagers.WAP utilizes binary transmission for greater compression of data and is optimized for long latency and low bandwidth. The lightweight WAP protocol stack is designed to minimize the required bandwidth and maximize the number of wireless network types that can deliver WAP content.

5.1.2 The WAP Model

Below figure shows the WAP programming model. You can clearly see the similarities with the Internet model i.e. the two models would have been practically identical without the WAP Gateway/Proxy.



WAP Gateway/Proxy is the entity that connects the wireless domain with the Internet. We should make a note that the request that is sent from the wireless client to the WAP Gateway/Proxy uses the Wireless Session Protocol (WSP). In its essence, WSP is a binary version of HTTP.

A markup language - the Wireless Markup Language (WML) has been adapted to develop optimized WAP applications. In order to save valuable bandwidth in the wireless network, WML can be encoded into a compact binary format. Encoding WML is one of the tasks performed by the WAP Gateway/Proxy.

5.1.3 Working of WAP Model

When it comes to actual use, WAP works like this:

1. The user selects an option on their mobile device that has a URL with Wireless Markup language (WML) content assigned to it.
2. The mobile device uses phone network to send the URL request to a WAP gateway, using the binary encoded WAP protocol.
3. This WAP request is translated into a conventional HTTP request for the specified URL, and sent to the Internet by the WAP gateway.
4. This HTTP request is now picked up by the appropriate Web server.
5. The request is processed by the server. If the URL refers to a static WML file, the server delivers it. If a CGI script is requested, it is processed and the content is returned as usual.
6. The HTTP header is added to the WML content and returned to the gateway by the Web server.
7. The WML is compiled intobinary form by WAP gateway.
8. The WML response is sent back to the phone by WAP gateway.
9. The WML is received by the mobile device via the WAP protocol.
10. The WML is processed and the content is displayed on the screen by micro-browser.

5.1.4 Benefits of WAP

- Wireless network operators can decrease churn, cut costs and increasesubscriber base.
- Content providers can have access to customers who seek enhanced services.
- End users can have more value and functionality from their mobile devices.

5.1.5 Limitations of WAP

- Device limitations that arise from limitations of power and formfactor.
- Bearer limitations that arise from limitations of bandwidth andmobility.
- Use case limitations that arise from the consumer nature of mobiledevices.

Check Your Progress 1

1. What is the WAP? Why do we need WAP?

   ……

   ……

2. Name the various layers of WAP stack.

   ……

   ……

3. Which layer of the WAP stack should have a bearer specific implementation and why?

   ……

……

4. What are the advantagesand limitations of WAP ? Give some example of WAP.

……

……

5.2 WML - Overview

The topmost layer in the WAP (Wireless Application Protocol) architecture is made up of WAE (Wireless Application Environment), which consists of WML and WML scripting language. WML is the markup language defined in the WAP specification. WML is an application of XML, which is defined in a document-type definition. WAP sites are written in WML, while web sites are written in HTML. WML is based on HDML and is modified so that it can be compared with HTML. WML is very similar to HTML. Both of them use tags and are written in plain text format. WML takes care of the small screen and the low bandwidth of transmission.

5.2.1 WML Versions:

WAP Forum has released a latest version WAP 2.0. Most of the new mobile phone models released are WAP 2.0-enabled. The markup language defined in WAP 2.0 is XHTML Mobile Profile (MP). The WML MP is a subset of the XHTML. A style sheet called WCSS (WAP CSS) has been introduced along with XHTML MP. The WCSS is a subset of the CSS2.WML 1.x is an earlier technology and a lot of wireless devices that only supports WML 1.x are still using it. Latest version of WML is 2.0 and it is created for backward compatibility purposes.

5.2.2 WML Components(WML Decks and Cards):

Developing in WML is slightly different from developing for the web. As far as the development for the web is concerned, each HTML file constitutes one HTML page. In WML, since each page or screen is very small, it does not make much sense for each page to constitute a separate file. WML pages – content viewed on separate screens – are *cards* and the cards are all placed within a *deck* of related pages that constitute one single file. So, a main difference between HTML and WML is that the basic unit of navigation in HTML is a page, while that in WML is a card. A WML file can contain multiple cards and they form a deck.

When a WML page is accessed from a mobile phone, all the cards in the page are downloaded from the WAP server. So if the user goes to another card of the same deck, the mobile browser does not have to send any requests to the server since the file that contains the deck is already stored in the wireless device.Logically, a user navigates through a set of cards. WML decks can be stored in ‘static’ files on an origin server, or the content generator that is running on an origin server can dynamically generate them. Each card, in a deck, contains a specification for a particular user interaction. We can put links, text, images, input fields, option boxes and many other elements in a card. WML comments use the same format as HTML comments.

Note that comments are not compiled or sent to the user agent, and thus have no effect on the size of the compiled deck.

5.2.3 WML Program Structure:

A WML program is typically divided into two parts: the documentprolog and the body. Consider the following code:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card id="one" title=" My First Card">
<p> Welcome to IGNOU!</p>
</card>
<card id="two" title=" My Second Card">
<p>Let us learn to make a Wireless World...</p>
</card>
</wml>
```

The first line of this text says that this is an XML document and the version is 1.0. The second line selects the document type and gives the URL of the document type definition (DTD).One WML deck (i.e. page) can have one or more cards as shown above.

WML Document Prolog:

The first line of this text says that this is an XML document and the version is 1.0. The second line selects the document type and gives the URL of the document type definition (DTD). The DTD referenced is defined in WAP 1.2, but this header changes with the versions of the WML. The header must be copied exactly so that the tool kits automatically generate this prolog.

The prolog components are not WML elements and they should not be closed, i.e. we should not give them an end tag or finish them with />

WML Document Body:

The body is enclosed within a <wml></wml> tag pair. The body of a WML document can consist of one or more of these, namely Deck,Card,Content to be shown and Navigation instructions.

5.2.4Testing Program

Firstly the above code is put in a file called test.wml file, and then this WML file is put locally on the hard disk. Now it can be viewed using an emulator.

This is by far the most efficient way of developing and testing WML files. If our aim is, however, to develop a service that is going to be available to WAP phone users, we should upload our WML files onto a server once WML files have been developed locally and test them over a real Internet connection. Before accessing any URL, we should make sure WAP Gateway Simulator is running on the PC.

When WAP program is downloaded then only the first card is seen at the mobile. Following is the output of the above example on Nokia Mobile Browser 4.0. This mobile supports horizontal scrolling. The text can be seen off the screen by pressing the "Left" or "Right" button.

First Card
Welcome to IGNOU!
OptionsBack

When the right button is pressed then second card will be visible as follows:

Second Card
Let us learn to make a Wireless World...
Options Back

Check Your Progress 2

1. Short Note on WML Component?

……
……

2. In what aspects is the WML development different from web development?

---

3. What are the Characteristic Features of WML?

5.3 WML Elements - formatting and links

WML has a set of *elements* that specify all markup and structural information for a WML deck. Elements are identified by tags, which are each enclosed in a pair of angle brackets. Unlike HTML, WML strictly adheres to the XML hierarchical structure and hence elements must contain a start tag and an end tag. WML is a case sensitive language. The Card and card are different things. Elements have one of the following two structures:

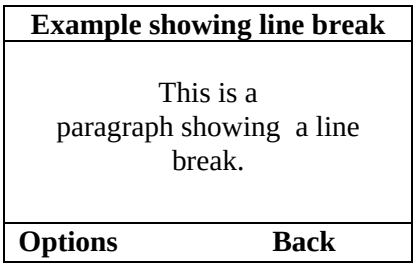
> **<tag> content </tag>** : This form is identical to HTML.
> **<tag />:** This is used when an element cannot contain visible content or is empty, such as a line break. WML document's prolog part does not have any element which has closing element.

5.3.1Line Break:

The <br /> element defines a line break and almost all WAP browsers support a line break tag. Following is the example showing usage of <br /> element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card title="Example showing line break">
<p align="center">This is a <br /> aparagraph showing a line break.
</p>
</card>
</wml>
```

This will produce following result:



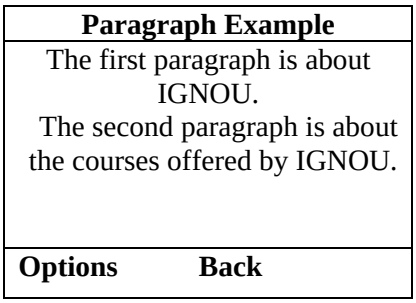
5.3.2Text Paragraphs:

The element defines a paragraph of text and WAP browsers always render a paragraph in a new line.In WML, all text to be displayed on the main part of the screen must be inside a paragraph element.A element is required to define any text, image or a table in WML.

Following is the example showing usage of element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card title="Paragraph Example">
<p align="center">The first paragraph is about IGNOU.</p>
<p align="right">The second paragraphis about the courses offered by IGNOU.</p>
</card>
</wml>
```

This will produce following result:



5.3.3 WML Tables:

The <table> element alongwith<tr> and <td> is used to create a table in WML. WML does not allow the nesting of tables.A <table> element should be put with-in ... elements.

Following is the example showing usage of <table> element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
```

```
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card title="WML Tables">
<p><table columns="3" align="LCR">
            <tr>
            <td>Col 1</td><td>Col 2</td><td>Col 3</td>
            </tr>
            <tr>
            <td>Course1</td><td>Course2</td><td>Course3</td>
            </tr>
            <tr>
            <td>Course4</td><td>Course5</td><td>Course6</td>
            </tr>
</table></p>
</card>
</wml>
```

This will produce following result:

WML Tables

Col 1	Col 2	Col 3
Course1	Course2	Course3
Course4	Course5	Course6

Options **Back**

5.3.4 Preformatted Text:

The <pre> element is used to specify preformatted text in WML. Preformatted text is text of which the format follows the way it is typed in the WML document. This tag preserves all the white spaces enclosed inside this tag. We have to make sure that we are not putting this tag inside ...

Following is the example showing usage of <pre> element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card title="Preformatted Text">
<pre>
  This preformatted
Text<b>Welcome to IGNOU</b>will appear
as it is.
</pre>
</card>
</wml>
```

This will produce following result:

Preformatted Text
This preformatted Text **Welcome to IGNOU** will appearas it is.
OptionsBack

5.3.5 WML Fonts - <b>, <big>,,<italic>

WML Elements	Purpose
<b>	Defines bold text
<big>	Defines big text
	Defines emphasized text
<i>	Defines italic text
<small>	Defines small text
	Defines strong text
<u>	Defines underlined text

5.3.6 WML – Images

'A picture is worth a thousand words'. With such small screens, small images instead of text can be very useful, allowing us to get more information across to the user in the small space available. Images can be particularly useful when used as links. For example an application could be navigated using small icons instead of lots of text.The <img> element is used to include an image in a WAP card.WAP-enabled wireless devices only supported the Wireless Bitmap (WBMP) image format.The file extension of WBMP is ".wbmp" and the MIME type of WBMP is "image/vnd.wap.wbmp".

Following is the example showing usage of <img> element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="main" title="Image">
<p>
What a beautiful image of the<br/>
<imgsrc="rainbow.wbmp" alt="rainbow"/><br/>
on a rainy day
</p>
</card>
</wml>
```

This will produce following result:

Image
What a beautiful image of the  on a rainy day
Options **Back**

5.3.7 WML Navigational Elements - <do>, <a>, <anchor>and <go>

Implementing good navigation around a WAP application is important since very little can be displayed on the small screen that the device has. In WML the elements that are used for navigation between WML cards are <do>, <a>, <anchor>and <go>. These are discussed below.

The do Element

The do element gives the user a general mechanism for performing navigation between cards. The do element establishes a soft key which gives the user the ability to alter the path or jump to a different card without having to navigate a full list of options.The do element may appear at both the card-level and deck level.

Card-Level

The do element may appear inside a card and may be located anywhere in the text flow. If the user agent intends to render the do element inline, it should use the element's anchor point as the rendering point.

Deck-Level

The do element may appear inside a template, indicating a deck level do element. A deck level do element applies to all the cards in the deck. It is equivalent to specifying the do element with in each card.

A card level do element overrides a deck-level do element if they have the same name. For a single card, the active do element are defined as the do elements specified in the card, plus any do elements specified in the deck's template and not overridden in the card.

Syntax

<do type="type" label="label" name="name">

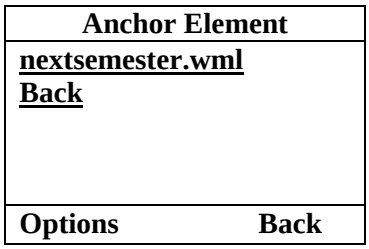
WML <anchor> Element:

The <anchor>...</anchor> tag pair is used to create an anchor link. It is used together with other WML elements called <go/>, <refresh> or <prev/>. These elements are called task elements and tell WAP browsers what to do when a user selects the anchor link. We can enclose Text or image along with a task tag inside <anchor>...</anchor> tag pair.

Following is the example showing usage of <anchor> element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card title="Anchor Element">
<p><anchor>
<go href="nextsemester.wml"/>
</anchor></p>
<p><anchor>
<prev/>
</anchor></p>
</card>
</wml>
```

This will produce following result:

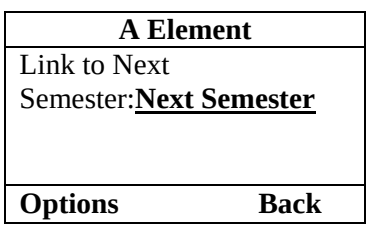


WML <a> Element:

The <a>... tag pair can also be used to create an anchor link and always a preferred way of creating links.We can enclose Text or image inside <a>... tags.Following is the example showing usage of <a> element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card title="A Element">
<p> Link to Next Semester:
<a href="nextsemester.wml">Next Semester</a>
</p>
</card>
</wml>
```

This will produce following result:



The go element is discussed later in WML Tasks.

Check Your Progress 3

1 List the various WML navigational elements.

2.Explain how Images are linked in WML?

5.4 WML - Inputs

WML provides various options to let a user enter information through WAP application.

First of all, let us look at the different options for allowing the user to make straight choices between items. These are usually in the form of menus and submenus, allowing users to drill down to the exact data that they want.

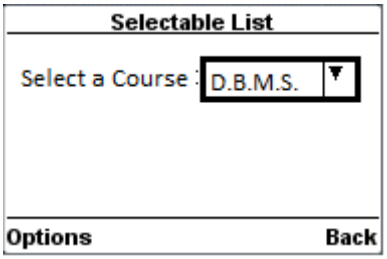
5.4.1 WML <select> Element:

The <select>...</select> WML elements are used to define a selection list and the <option>...</option> tags are used to define an item in a selection list. Items are presented as radio buttons in some WAP browsers. The <option>...</option> tag pair should be enclosed within the <select>...</select> tags.

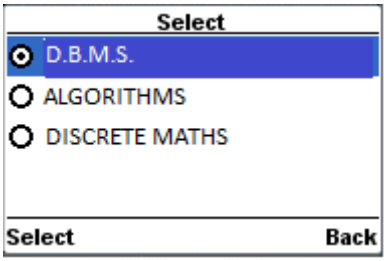
Following is the example showing usage of these two elements.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>

<card title="Selectable List">
<p> Select a Course :
<select>
<option value="dbms">D.B.M.S</option>
<option value="algo">ALGORITHMS </option>
<option value="dm">DISCRETE MATHS </option>

</select>
</p>
</card>

</wml>
```

When we will load this program it will show us following screen:



Once we highlight and enter on the options it will display following screen:



We want to provide option to select multiple options then set *multiple* attribute to *true* as follows:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM
//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>

<card title="Selectable List">
<p> Select a Subject :
<select multiple="true">
<option value="dbms">D.B.M.S</option>
<option value="algo">ALGORITHMS </option>
<option value="dm">DISCRETE MATHS </option>
</select>
</p>
</card>

</wml>
```

This will give us a screen to select multiple options.

5.4.2 WML <input> Element:

The <input/> element is used to create input fields and input fields are used to obtain alphanumeric data from users. This element supports the following attributes:

Attribute	Value	Description
Name	Text	The name of the variable that is set with the result of the user's input

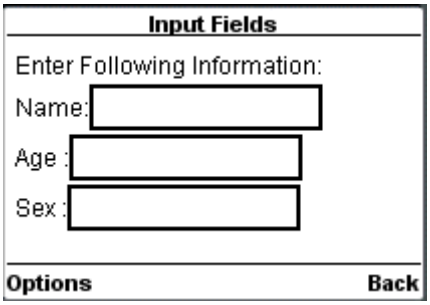
maxlength	Number	Sets the maximum number of characters the user can enter in the field
emptyok	true false	Sets whether the user can leave the input field blank or not. Default is "false"
format	A a N X x M m *f *nf*	Sets the data format for the input field. Default is "*M". A = uppercase alphabetic or punctuation characters a = lowercase alphabetic or punctuation characters N = numeric characters X = uppercase characters x = lowercase characters M = all characters m = all characters **f* = Any number of characters. Replace the *f* with one of the letters above to specify what characters the user can enter *nf* = Replace the *n* with a number from 1 to 9 to specify the number of characters the user can enter. Replace the *f* with one of the letters above to specify what characters the user can enter
Size	Number	Sets the width of the input field
tabindex	Number	Sets the tabbing position for the select element
Title	Text	Sets a title for the list
Type	text password	Indicates the type of the input field. The default value is "text". Password field is used to take password for authentication purpose.
Value	Text	Sets the default value of the variable in the "name" attribute
xml:lang	language_code	Sets the language used in the element
Class	class data	Sets a class name for the element.
Id	element ID	A unique ID for the element.

Following is the example showing usage of this element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>

<card title="Input Fields">
```

```
<p> Enter Following Information:<br/>
 Name: <input name="name" size="12"/>
 Age :  <input name="age" size="12" format="*N"/>
 Sex :  <input name="sex" size="12"/>
</p>
</card>

</wml>
```

This will provide us following screen to enter required information:



5.4.3 Setvar Element

The setvar element specifies the variable to be set in the current browser context as a side effect of executing a task. The element is ignored if the name attribute does not evaluate to a legal variable name at runtime.

Syntax

<setvar name=”vname” value=”value”>

Name specifies the variable name and Value specifies the value to be assigned to the variable. Both the attributes are required.The following element would create a variable named *a* with a value of 1000:

```
<setvar name="a" value="1000"/>
```

5.4.4 WML <fieldset> Element:

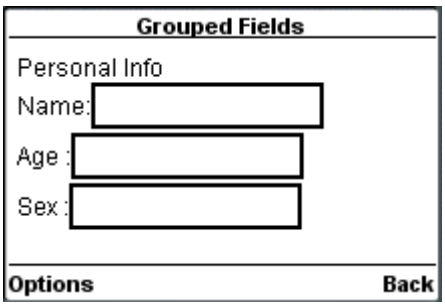
The <fieldset/> element is used to group various input fields or selectable lists.

Following is the example showing usage of this element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>

<card title="Grouped Fields">
<p>
```

```
<fieldset title="Student Info">
 Name: <input name="name" size="12"/>
 Age :  <input name="age" size="12" format="*N"/>
 Sex :  <input name="sex" size="12"/>
</fieldset>
</p>
</card>

</wml>
```

This will provide us following screen to enter required information. This result may differ browser to browser.



5.4.5 WML <optgroup> Element

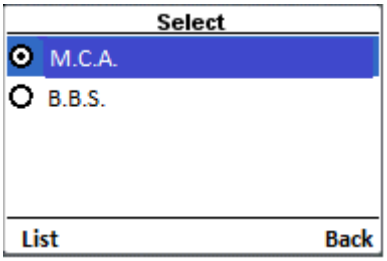
The <optgroup/> element is used to group various options together inside a selectable list.

Following is the example showing usage of this element.

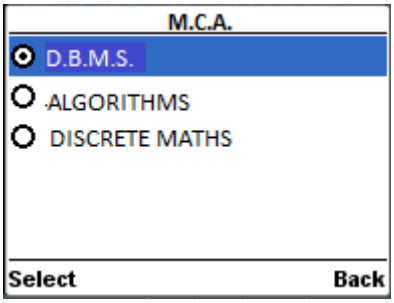
```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>

<card title="Selectable List">
<p> Select a Course :
<select>
<optgroup title="M.C.A">

<option value="dbms">D.B.M.S</option>
<option value="algo">ALGORITHMS </option>
<option value="dm">DISCRETE MATHS </option>
</optgroup>
<optgroup title="B.B.S">
<option value="marketing">Marketing</option>
<option value="ob">Organizational Behaviour</option>
<option value="hrm">Human Resource Management</option>
</optgroup>

</select>
```

```
</p>
</card>

</wml>
```

When a user loads above code then it will give two options to be selected:



When a user selects any of the options then only it will give final options to be selected. So if user selects India then it will show us following options to be selected:



5.4.6 WML - Submit Data to Server

Similar to *HTML Form* WML also provide a mechanism to submit user data to web server. To submit data to the server in WML, we need the <go>...</go> along with <postfield/> tags. The <postfield/> tag should be enclosed in the <go>...</go> tag pair. To submit data to a server, we collect all the set WML variables and use <postfield> elements to send them to the server. The <go>...</go> elements are used to set posting method to either POST or GET and to specify a server side script to handle uploaded data.

CHECK YOUR PROGRESS 4

1. What are Input Elements? Explain briefly.

---

2. Discuss about Select Element?

---

3. What is the use of DO Element?

---

4. Define setvar element?

5.5 WML – Tasks

A WML task is an element that specifies an action to be performed by the browser, rather than something to be displayed. For example, the action of changing to a new card is represented by a <go> task element and the action of returning to the previous card visited is represented by a <prev> task element. Task elements encapsulate all the information required to perform the action. WML provides four elements to handle four WML tasks called go task, pre task, refresh task and noop tasks.

5.5.1 go task

A go executes a push operation on the history stack. As the name suggests, the <go> task represents the action of going to a new card.Following is the example showing usage of <go> element.

```
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card title="GO Element">
<p>
<anchor>
      Semester 1 : <go href=semester1.wml"/>
</anchor>
</p>
</card>
</wml>
```

5.5.2 The <prev> Task:

The <prev> task represents the action of returning to the previously visited card on the history stack. When this action is performed, the top entry is removed from the history stack, and that card is displayed again, after any <setvar> variable assignments in the<prev> task have taken effect.If no previous URL exists, specifying <prev> has no effect. Following is the example showing usage of <prev> element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card title="Prev Element">
<p>
<anchor> Previous Page :<prev/></anchor>
</p>
</card>
</wml>
```

5.5.3 The <refresh> Task:

The <refresh> task performs the variable assignments specified by its <setvar> elements and then redisplays the current card with the new values. It is most often used to perform some sort of "reset" action on the card. The <go> and <prev> tasks perform the same action just before displaying the new card. Following is the example showing usage of <refresh> element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card title="Referesh Element">
<p>
<anchor> Refresh this page:
<go href="test.wml"/>
<refresh>
<setvar name="x" value="100"/>
</refresh>
</anchor>
</p>
</card>
</wml>
```

5.5.4 The <noop> Task:

The purpose of the <noop> task is to do nothing (no operation). This attribute is useful for overriding deck-level do elements.The only real use for this task is in connection with templatesFollowing is the example showing usage of <noop> element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card title="Noop Element">
<p>
<do type="prev" label="Back">
<noop/>
</do>
</p>
</card>
</wml>
```

Check Your Progress 5

1. Write a WML application to navigate between cards.

---

2. Discuss one situation where it can be useful to include variables in a <prev> task.

5.6 WML - Events

Events can be used to make WAP applications dynamic. Event in ordinary language as the name suggests, is simply the happening of something notable.In programming **event** is identical in meaning, but with one major difference. When something happens in a computer system, the system itself has to **(1)** detect that something has happened and **(2)** know what to do about it.WML language also supports events and we can specify an action to be taken whenever an event occurs. This action could be in terms of WMLScript or simply in terms of WML. WML supports following four event types:

- onenterbackward: This event occurs when the user hits a card by normal backward navigational means. That is, user presses the Back key on a later card and arrives back at this card in the history stack.
- onenterforward: This event occurs when the user hits a card by normal forward navigational means.
- onpick: This is more like an attribute but it is being used like an event. This event occurs when an item of a selection list is selected or deselected.
- ontimer: This event is used to trigger an event after a given time period.

These event names are case sensitive and they must be lowercase.

5.6.1**WML – Timer**

WML provides us**ontimer** event to handle if user wants something to happen without the user explicitly having to activate a control.

We can bind a task to this event with the <onevent> element. Here is the syntax:

```
<onevent type="ontimer">
  A task to be performed.
</onevent>
```

Here a task could be <go>, <prev> or <refresh>.

5.6.2 WML <timer> Element:

A timer is declared inside a WML card with the <timer> element. It must follow the <onevent> elements if they are present. (If there are no <onevent> elements, the <timer> must be the first element inside the <card>.) No more than one <timer> may be present in a card.

Following is the example showing usage of <timer> element.

```
<<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="card1" title="First card"
ontimer="#card2">
<timer value="10"/>
<p>
Welcome to IGNOU...<br/>
This is First Card
<do type = "accept" label="Next">
```

```
<go href="#card2"/>
</do>
</p>
</card>
<card id="card2" title="Second card"
ontimer="#card1">
<timer value = "20"/>
<p>
Let us learn to make a Wireless World <br/>
This is the Second Card
<do type = "prev" label="Back">
<prev/>
</do>
</p>
</card>
</wml>
```

The first card Welcome to IGNOU... would be visible on the screen for 10 seconds. Automatically, the second card comes on view as soon as thetimer expires. The second card,Let us learn to make a Wireless World would be displayed for 20 seconds. At the expiry of the timer, the user is taken back to the first card.

5.7 WML Variables

When a user switch from card to card in a deck, we need to store data.This mechanism is provided via WML variables.WML variables arecase sensitive.

5.7.1 Naming the Variable

Specify a Variable with the Setvar Command

When someone executes a task (like go, prev, and refresh), the setvar element can be used to set a variable with a specified value.

The following example will create a variable named k with a value of 345:
<setvar name=”k” value =”345”/>
The name and value attributes are required.

Specify a Variable through an Input Element

Variables can also be set through an input element (like input, select, option, etc.). A variable is automatically created that corresponds with the named attribute of an input element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
<card id="first" title="First Card">
<p align="center">
Select Any one... <br/>
<select name="coursename">
```

```
<option value="no">Select </option>
<option value="MCA">MCA Course</option>
<option value="MBA">MBA Course</option>
</select>
<do type="access" label="next">
<go href="#card2"/>
</do>
</p>
</card>
<card id="card2" title="Second Card">
<p>You selected: $(coursename)</p>
</card>
</wml>
```

The use of variable that we created in the example above:

You selected: $(coursename)

5.8 WML Examples

A WML deck with two cards - one for user input and one for displaying the result - can be set up, like demonstrated in this example:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
<card id="card1" title="Course">
<do type="accept" label="Answer">
<go href="#card2"/>
</do>
<p>
<select name="name">
<option value="MCA">MCA Course</option>
<option value="BBS">BBSCourse</option>
<option value="BFIA">BFIA Course</option>
</select>
</p>
</card>
<card id="card2" title="Answer">
<p>
You selected: $(name)
</p>
</card>
</wml>
```

Example Explained

The Prolog <?xml version="1.0"?>

<!DOCTYPEwml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

The first lines in the WML document is called the prolog: The prolog defines that this is an XML document. It defines the XML version, and the DTD to be referenced.

The Deck <wml> .....</wml>

The deck is the WML document itself. It is embedded within <wml> tags

The Cards

<card> ..... </card>

Card are always displayed one at the time. The two cards named "card1" and "card2" are defined between <card> tags

The <do> element

<do> ... </do>

The first card has a <do> element that defines an event to be triggered. The type="accept" attribute of the do element causes the label="Answer" to be displayed in the lower left corner of the display.

The Event

The go element triggers when the user clicks the do label. The href="#card2" attribute of the go element causes card2 to be displayed on the screen.

The Variable

Card2 displays the $(name) variable from card1, because variables are valid across cards.

Check Your Progress 6

1. Discuss briefly about WML Event?

---

2. How to use Timer in WML?

---

3. What are the WML variables? How to use them?

5.9 Summary

WAP is simply aprotocol – a standardised way by which a wireless device talks to a server installed in a wireless network.WAP provides a method to communicate across wireless networks quickly, securely and efficiently. WAPprovides the opportunity to integrate databases, dynamic content, e-commerce and secure information trafficking through an WAP-enabled device. It takes a client sever

approach. It incorporates a relatively simple microbrowser into a mobile phone requiring only limited resources on the mobile phone. This makes WAP suitable for thin clients and early smart phones. It is aimed at turning a mass- market mobile phone into a "network based smart phone". The philosophy behind the WAP approach is to utilise as few resources as possible on the handheld device and compensate for the constraint of the device by enriching the functionality of the network. WML is a markup language, which is used to interface with the WAP browser. It allows a programmer to simultaneously develop web-based applications that can be viewed both within a traditional web browser and within a hand held device. WML supports several elements to solicit user input. The elements can be combined into one or more cards. All requests for user input are made in abstract terms, allowing the user agent the freedom to optimize features for the particular device. WML includes a small set of input controls. For example, WML includes a text entry control that supports text and password entry. Text entry fields can be masked preventing the end user from entering incorrect character types. WML also supports client-side validation by allowing the author to invoke scripts at appropriate times to check the user's input. It includes an option selection control that allows the author to present the user with a list of options that can set data, navigate among cards, or invoke scripts. WML supports both single and multiple option selections. WMLalso includes task invocation controls. When activated, these controls initiate navigation or a historymanagement task such as traversing a link to another card (or script) or popping the current card off of thehistory stack. The user agent is free to choose how to present these controls. It may for example, bindthem to physical keys on the device, render button controls in a particular region of the screen (or inlinewithin the text), bind them to voice commands, etc.WML allows several navigation mechanisms using URLs. It also exposes a first –class history mechanism.Navigation includes HTML-style hyperlinks, inter-card navigation elements, as well as history navigationelements.

5.10 Answers to Check Your Progress

Check Your Progress 1

Ans.1 WAP (Wireless Application Protocol) is simply a communication protocol by which a wireless device talks to a server installed in a wireless network. It is an open, global specification, which gives mobile users with access to wireless devices the opportunity to easily access the Internet or other computer applications, defined by the WAP forum. Standard HTML content cannot be effectively displayed on the small-size screens of pocket-sized mobile phones and pagers. WAP utilizes binary transmission for greater compression of data and is optimized for long latency and low bandwidth. The lightweight WAP protocol stack is designed to minimize the required bandwidth and maximize the number of wireless network types that can deliver WAP content.

Ans.2 It consists of five layers, namely: application layer, session layer, transaction layer, security layer and datagram layer.

Ans.3 Network layer of WAP i.e. WDP(Wireless Datagram Protocol) shields the upper layers from the bearer services like SMS, CMD etc. provided by the network and hence allows applications a transparent transmission of data over different bearers. WDP must have bearer specific implementation since it is the only layer that has to interface various bearer networks.

Ans.4 Increase in subscriber base, enhanced services, low cost, more value and functionality from mobile devices are some of the advantages of WAP. Limitations of WAP areimposed by device limitations and by the bandwidth and mobility limitations. Some basic examples of WAP are given below:

1. Using WAP we can get information of train time-table.
2. Using WAP can purchase tickets like: movie,journey ticket etc.
3. Using WAP we perform task like flight check in
4.We can also view traffic information.
5.We can get information about current weather condition.
6.Using WAP we can also do trading of shares.
7.We can also display sport results on our small wireless devices.

Check Your Progress 2

Ans.1: WML Components consists of WML Decks and Cards. WML pages – content viewed on separate screens – are *cards* and the cards are all placed within a *deck* of related pages that constitute one single file.

Ans.2: In WML, since each page or screen is very small, it does not make much sense for each page to constitute a separate file.So, a main difference between HTML and WML is that the basic unit of navigation in HTML is a page, while that in WML is a card..

Check Your Progress 3

Ans.1: In WML the elements that are used for navigation between WML cards are <do>, <a>, <anchor>and <go>.

Ans.2: The do element gives the user a general mechanism for performing navigation between cards. These buttons give the user the ability to alter the path or jump to a different card without having to navigate a full list of options.The do element may appear at both the card-level and deck level.

Ans.3: The <img> element is used to include an image in a WAP card.WAP-enabled wireless devices only supported the Wireless Bitmap (WBMP) image format.The file extension of WBMP is ".wbmp" and the MIME type of WBMP is "image/vnd.wap.wbmp".

Following is the example showing usage of <img> element.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="main" title="Image">
<p>
What a beautiful image of the <br/>
<imgsrc="rainbow.wbmp" alt="rainbow"/><br/>
on a rainy day
</p>
</card>
</wml>
```

Check your progress 4

Ans.1: The <input/> element is used to create input fields and input fields are used to obtain alphanumeric data from users.

Ans.2: The <select>...</select> WML elements are used to define a selection list and the <option>...</option> tags are used to define an item in a selection list. Items are presented as radio buttons in some WAP browsers. The <option>...</option> tag pair should be enclosed within the <select>...</select> tags.

Ans.3: The setvar element specifies the variable to be set in the current browser context as a side effect of executing a task. The element is ignored if the name attribute does not evaluate to a legal variable name at runtime.

Syntax

<setvar name=”vname” value=”value”>

Check Your Progress 5

Ans.1:

```
<wml>
<card id="main">
<do type="accept" label="Enter">
<go href="#inbox"/>
</do>

<p><b>Messages</b>
</p>
</card>
<card id="inbox">
<do type="prev"><prev/></do>
<p><anchor><go href="#c1"/>1) Inbox</anchor><br/>
<anchor><go href="#c2"/>2) Outbox</anchor><br/>
</p>
</card>
<card id="c1">
<do type="prev"><prev/></do>
<p>
NO SPACE.
</p>
</card>
<card id="c2">
<do type="prev"><prev/></do>
<p>
NO MESSAGES.
</p>
</card>
</wml>
```

Ans.2: One situation where it can be useful to include variables in a <prev> task is a login page, which prompts for a username and password. In some situations, we may want to clear out the password field when returning to the login card, forcing the user to reenter it. This can be done with a construct such as:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
```

```
"http://www.wapforum.org/DTD/wml12.dtd">

<wml>
<card title="Prev Element">
<p>
<anchor>
<prev>
<setvar name="password" value=""/>
</prev>
</anchor>
</p>
</card>
</wml>
```

Check Your Progress 6

Ans.1:WML language supports events and we can specify an action to be taken whenever an event occurs. WML supports following four event types:

- **onenterbackward:** This event occurs when user presses the Back key on a later card and arrives back at this card in the history stack.
- onenteforward: This event occurs when the user hits a card by normal forward navigational means.
- onpick: This event occurs when an item of a selection list is selected or deselected.
- ontimer: This event is used to trigger an event after a given time period.

Ans.2: We can set Timer in WML.We express time unit of timer as 1/10 of a second.

Below given is the example which will display text on WML page for 6 seconds.

Example:

```
<?xml version="1.0"?>
<!DOCTYPEwml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card ontimer="timetest.wml"><timer value="60"/>
<p>Write some text here</p>
</card>
</wml>
```

Ans.3: Use of variables in WML is that when client wants to switch over card to card in a deck, then client stores the data into variables.

We can use variables in WML in two ways.

1. Can set a variable using setvar Command: Suppose that if client want to execute go,prev and refresh type task than they can use setvar command and create a variable with a specific value.

Example:

```
<setvar name="x" value="9888"/>
```

2. Can set a variable with an input element: If we want to set variable with an input,select, option etc input element.

Example:

```
<card id="card1">
<select name="course">
<option value="DBMS">DBMS </option>
<option value="CSA">CSA</option>
<option value="DISCRETE">DISCRETE MATHS</option>
</select>
</card>
```

We can use this created variable course into the card2.

```
<card id="card2">
<p>You have selected: $(course)</p>
</card>
```

In the above example we create a variable course in card1 and passed the selected course into card2 by variable course.

5.11 References

www.wapforum.org

www.webopedia.com/TERM/W/WML.html

www.w3professors.com/Pages/Courses/WAP/Wap-Wml-Programs.html

www.wirelessdevnet.com/channels/wap/

http://www.tutorialspoint.com/wml/wml_useful_resources

http://www.w3schools.com/xml/xml_usedfor.asp