Unit 1 (Introduction to Web)

History of the Web

1: Tim Berners-Lee, a British scientist, invented the World Wide Web (WWW) in 1989, while working at CERN.
2: The Web was originally conceived and developed to meet the demand for automated information-sharing between scientists in universities and institutes around the world.
3: He developed the first web server, the first web browser, and a document formatting protocol, called Hypertext Markup Language (HTML).
4: Websites for use by the general public began to emerge in 1994.
5: Following the complete removal of commercial restrictions on Internet use by 1995, commercialization of the Web amidst macroeconomic factors led to the dot-com boom and bust in the late 1990s and early 2000s.

Evolution of www:

1989-1990: Birth of the Web:

- Tim Berners-Lee, a British computer scientist, proposed the concept of the World Wide Web in 1989.
- The first website, *http://info.cern.ch*, went live in 1991.

1990s: Early Growth:

- The web saw rapid growth during the 1990s as more websites and browsers emerged.
- Commercialization of the web began, and businesses started creating an online presence.

Late 1990s: Dot-com Boom and E-commerce:

- The late 1990s saw the dot-com boom, with numerous internet-based companies emerging.
- Introduction of CSS (Cascading Style Sheets) for better web design.

Early 2000s: Web 2.0:

- The term "Web 2.0" was coined to describe a shift from static web pages to dynamic, interactive content.

Mid-2000s: Rise of Social Media:

- Platforms like Facebook, Twitter, and YouTube gained widespread popularity, changing how people interacted online.

2010s: Cloud Computing and Progressive Web Apps:

- Cloud computing has become integral to web development, enabling scalable and flexible solutions.

2020s: Continued Advancements:

- Continued emphasis on web performance, security, and accessibility.
- Artificial intelligence and machine learning integration into web applications.

Web development cycle

1:Information Gathering:

In this phase, the development team gathers essential information about the project's objectives, target audience, scope, and requirements.The goal is to establish a clear understanding of what the website should achieve.

2:Planning:

During the planning phase, the team outlines the project's roadmap. This includes defining the project scope, setting timelines, allocating resources, and establishing a budget.

3:Design:

In the design phase, the project's visual and user experience (UX) aspects are created.

4:Development:

In the development phase, the actual coding and implementation of the website take place.

5:Testing and Delivery:

Testing is a crucial phase where the website's functionality, performance, and security are rigorously tested.

6:Maintenance Phase:

This involves monitoring the website's performance, addressing any issues that arise, and making updates as necessary.

THE PROCESS OF WEB PUBLISHING

1: Web publishing is the process of publishing original content on the Internet.

2:The process includes building and uploading websites, updating the associated webpages, and posting content to these webpages online.

3: Web publishing comprises of personal, business, and community websites in addition to e-books and blogs.

4:The content meant for web publishing can include text, videos, digital images, artwork, and other forms of media.

5: Publishers must possess a web server, a web publishing software, and an Internet connection to carry out web publishing.

6: Web publishing is also known as online publishing.

WEB CONTENT

1:Two primary types of web content exist: text and multimedia.

2: Textual content is presented in text blocks or images, often containing internal links for added information.

3: Multimedia encompasses non-textual elements such as animations, images, audio, and videos.

4: Animations, achieved through Flash, Ajax, or GIFs, enhance visual appeal. **5:**Images, including clip art and photos, are common, while audio files enhance website attractiveness.

6: Videos are popular but need compatibility across browsers for seamless integration.

7: Publishers should ensure efficient performance when incorporating multimedia to create an engaging and accessible website experience.

Static website

1: Static website is the basic type of website that is easy to create.

2: You don't need web programming and database design to create a static website.

3: Its web pages are coded in HTML.

4: The codes are fixed for each page so the information contained in the page does not change and it looks like a printed page.

Advantages

1: No programming skills are required to create a static page.

2: No particular hosting requirements are necessary.

3: Can be viewed directly by a web browser without needing a web server or

application server.

Disadvantages

1: Any personalization or interactivity has to run client-side (i.e. in the browser), which is restricting.

2: Maintaining large numbers of static pages as files can be impractical without

automated tools.

Application areas of Static Website:

1: Brochure Websites: Static sites are great for creating simple online brochures that provide information about a company, product, or service.

2: Documentation: Technical documentation or user guides can be presented effectively on static websites, providing structured information to users.

3: Prototyping: Designers and developers often create static prototypes of websites before implementing them with dynamic functionality.

Dynamic website

1: Dynamic website is a collection of dynamic web pages whose content changes dynamically.

2: It accesses content from a database or Content Management System (CMS).

3: Therefore, when you alter or update the content of the database, the content of the website is also altered or updated.

4: Dynamic website uses client-side scripting or server-side scripting, or both to generate dynamic content.

5: Client side scripting generates content at the client computer on the basis of user input.

Applications of dynamic website

1: Dynamic websites power online stores, enabling users to browse products, add items to carts, and complete transactions.

2: Social networking Platforms like Facebook, Twitter, and Instagram rely on dynamic websites to facilitate user interactions, posts, comments, and real-

time updates.

3: Hotels, airlines, restaurants, and event organizers use dynamic websites to manage bookings and reservations.

Static vs Dynamic difference

SL.NO	Static Web Page	Dynamic Web Page
1.	In static web pages, Pages will remain same until someone changes it manually.	In dynamic web pages, Content of pages are different for different visitors.
2.	Static Web Pages are simple in terms of complexity.	Dynamic web pages are complicated.
3.	In static web pages, Information are change rarely.	In dynamic web page, Information are change frequently.
4.	Static Web Page takes less time for loading than dynamic web page.	Dynamic web page takes more time for loading.
5.	In Static Web Pages, database is not used.	In dynamic web pages, database is used.
6.	Static web pages are written in languages such as: HTML, JavaScript, CSS, etc.	Dynamic web pages are written in languages such as: CGI, AJAX, ASP, ASP.NET, etc.
7.	Static web pages does not contain any application program .	Dynamic web pages contains application program for different services.
8.	Static web pages require less work and cost in designing them.	Dynamic web pages require comparatively more work and cost in designing them.

Unit 2 (Languages and Technologies for Browsers)

HTML

1: HTML, short for Hypertext Markup Language, is the main markup language for web pages.
2: It employs HTML elements, enclosed in angle bracket "tags," to structure content.
3: This allows embedding images, objects, and interactive forms.
4: It organizes text with headings, paragraphs, lists, links, and quotes. **4:** Additionally, HTML enables the inclusion of scripts like JavaScript, impacting webpage behavior.
6: It's also utilized for incorporating Cascading Style Sheets (CSS) that define text appearance and layout.
7: The W3C, responsible for HTML and CSS standards, promotes CSS for layout instead of direct presentational markup.

Brief history of HTML

1. **Genesis (1989-1991):** Tim Berners-Lee develops the concept of HTML while working at CERN in 1989 and officially proposes the World Wide Web in 1990. The first HTML document is created in 1991.

2. **HTML 1.0 (1991-1995):** The initial release of HTML with 18 tags. Its purpose was to structure and link documents.

3. **HTML 2.0 (1995):** The first standardized version, introducing new features like forms and text alignment.

4. **HTML 3.2 (1997):** Brought enhancements such as tables and applets, aiming for improved layout and interactivity.

5. **HTML 4.01 (1999):** Focused on separating content and presentation, promoting cleaner code.

6. **XHTML (2000s):** XHTML aimed for stricter syntax and compatibility with XML. However, it didn't gain as much traction as expected.

7. **HTML5 (2014):** A significant evolution, introducing new elements, attributes, and APIs. Improved support for multimedia, better document structure, and compatibility with various devices.

8. **Ongoing Development (2020s):** HTML continues to evolve, with ongoing efforts to enhance its capabilities and keep up with the changing landscape of web development. The latest specifications and best practices are followed to create modern and interactive web pages.

Minimum HTML5 Document

Below is a simple HTML5 document, with the minimum of required tags:

```
<!DOCTYPE html>
<html>
<head>
<title>Title of the document</title>
</head>
<body>
The content of the document......
</body>
</html>
```

DHTML

1: Dynamic HTML, or DHTML, refers to a blend of technologies like HTML, JavaScript, CSS, and the Document Object Model that collaborates to produce interactive web pages with animations.

2: DHTML empowers scripting languages to modify variables in a page's definition language, influencing the appearance and functionality of ostensibly "static" HTML content.

3: Unlike generating a unique page for each load, DHTML's dynamic aspect lies in its operation during page viewing.

4: Conversely, a dynamic web page encompasses various instances, including those tailored through client-side and server-side scripting, producing personalized content based on users, load events, or variables.

---

four parts to DHTML

1: DOM

1: Document Object Model; The DOM or DocumentObject Model is the API

that binds JavaScript and other scripting languages together with HTML and other markup languages.

2: It is what allows Dynamic HTML to be dynamic.

3: The DOM is what allows you to access any part of your Web page to change it with DHTML.

4: Every part of a Web page is specified by the DOM and using its consistent

naming conventions you can access them and change their properties.

2: Scripts

1: Scripts written in either JavaScript or ActiveX are the two most common scripting languages used to activate DHTML.

2: You use a scripting language to control the objects specified in the DOM.

3: CascadingStyleSheets (CSS)

1: CSS is used in DHTML to control the look and feel of the Web page.

2: Style sheets define the colors and fonts of text, the background colors and images, and the placement of objects on the page.

3: Using scripting and the DOM, you can change the style of various elements

4: XHTML

1: XHTML or HTML 4.x is used to create the page itself and build the elements for the CSS and the DOM to work on.

2: There is nothing special about XHTML for DHTML but having valid XHTML is even more important, as there are more things working from it than just the browser.

Features of DHTML

There are four primary features of DHTML:

1: Changing the tags and Properties

1: This is one of the most common uses of DHTML.

2: It allows you to change the

qualities of an HTML tag depending on an event outside of the browser (such as a

mouse click, time, or date, and so on). **3:**You can use this to preload information onto a page, and not display it unless the reader clicks on a specific link.

2: Real-time postioning

1: When most people think of DHTML this is what they expect.

2: Objects, images, and text moving around the Web page.

3: This can allow you to play interactive games with your readers or animate portions of your screen.

3: Dynamic Fonts

1: This is a Netscape only feature. **2:** Netscape developed this to get around the problem designers had with not knowing what fonts would be on a reader's system.

3: With dynamic fonts, the fonts are encoded and downloaded with the page, so that the page always looks how the designer intended it to.

4: Data binding

1: This is an IE only feature. Microsoft developed this to allow easier access to

databases from Web sites.

2: It is very similar to using a CGI to access a database, but uses an ActiveX control to function.

3: This feature is very advanced and difficult to use for the beginning DHTML writer.

XHTML

XHTML (Extensible Hypertext Markup Language) is a familyof XML markup

languages that mirror or extend versions of the widely used Hypertext Markup Language (HTML), the language in which web pages are written.

Changes in XHTML from HTML

1: All tags must be in lower case.

2: All documents must have a doctype.

3: All documents must be properly formed All tags must be closed.

4: All attributes must be added properly. 5: The name attribute has changed.

6: Attributes cannot be shortened All tags must be properly nested.

Doctype for XHTML

1: In XHTML, there are three Document Type Declarations (DOCTYPEs) commonly used:

•Strict

•Transitional

•Frameset

2: These DOCTYPEs define the rules for validating the structure of XHTML documents.

3: The "Strict" version enforces stricter markup rules, while the "Transitional" and "Frameset" versions allow for some deprecated elements for compatibility purposes.

4: Keep in mind that HTML5 has largely replaced XHTML in modern web development.

General Rules for converting HTML to XHTML

Textbook page no: 21 to 23

JAVASCRIPT

1: A scripting language developed by Netscape to enable Web authors to design interactive sites.

2: Although it shares many of the features and structures of the full Java

language, it was developed independently.

3: JavaScript can interact with HTML source code, enabling Web authors to spice up their sites with dynamic content.

4: JavaScript is endorsed by a number of software companies and is an open language that anyone can use without purchasing a license.

5: It is supported by recent browsers from Netscape and Microsoft, though Internet Explorer supports only a subset, which Microsoft calls Jscript.

What can JavaScript do?

1: JavaScript provides HTML designers with a simple programming tool.

2: Despite HTML authors not typically being programmers, JavaScript's uncomplicated syntax allows anyone to embed code snippets into HTML pages.

3: This scripting language can insert dynamic text using statements like "document.write("<h1>" + name + "</h1>")" to display variable content. **4:** Additionally, JavaScript can respond to events such as page loading completion or user clicks on HTML elements.

5: It empowers developers to read and modify HTML element content and validate form data before submission, reducing server load.

6: Moreover, JavaScript aids in identifying visitors' browsers and can even create cookies to store and retrieve data on visitors' computers.

CGI

1: Common Gateway Interface is one of the older standards on the internet for moving data between a webpage and a web server.

2: CGI is by far and away the most

commonly used method of handling things like guestbooks, email forms, message boards and so on.

3: CGI is actually a standard for passing data back and forth and not a scripting language at all.

4: In fact, CGI routines are commonly written in interpreted languages such as PERL or compiled languages like C.

CSS

1: Cascading Style Sheets to format your web pages anyway that you want.

2: CSS is a language that describes the style of an HTML document.

3: CSS describes how HTML elements should be displayed.

JSP

1: Java Server Pages (JSP) is a server-side programming technology that

enables the creation of dynamic, platform-independent method for building Web-based applications.

2: JSP have access to the entire family of Java APIs, including the JDBC API

to access enterprise databases.

JavaScript for HTML Design

1: JavaScript provides HTML designers with a simple programming tool.

2: Despite HTML authors not typically being programmers, JavaScript's uncomplicated syntax allows anyone to embed code snippets into HTML pages.

3: This scripting language can insert dynamic text using statements like "document.write("<h1>" + name + "</h1>")" to display variable content.

4: Additionally, JavaScript can respond to events such as page loading completion or user clicks on HTML elements.

5: It empowers developers to read and modify HTML element content and validate form data before submission, reducing server load.

6: Moreover, JavaScript aids in identifying visitors' browsers and can even create cookies to store and retrieve data on visitors' computers.

ASP and *ASP.net/*

1: ASP is a technology developed by Microsoft that allows developers to build web pages with server-side scripts.
2: These scripts can be written in languages like VBScript or JScript.
3: ASP enables the execution of code on the web server before sending the resulting HTML to the client's browser.
4: *ASP.NET/*, on the other hand, is an evolution of ASP and provides a more robust and scalable framework for building web applications.
5: It supports multiple programming languages such as C#, *VB.NET/*, and F# and offers a more structured and object-oriented approach to development.

Microsoft Office

1: The Microsoft Office suite includes a number of tools, including Word, Excel, Access and Powerpoint.

2: Each of these tools has the ability to save in HTML format and has special commands for the internet.

3: This is especially useful, for example, if you work in an office where people are trained in Excel and you don't want to retrain them to create web pages.

4: On the other hand, if you are creating internet web sites (as opposed

to intranet sites) you probably would be better off using web specific products to edit your web pages.

Perl

1: A great scripting language which makes use of the CGI standard to allow work to be done on the web server.

2: PERL is very easy to learn (as programming languages go) and straightforward to use.

3: It is most useful for guestbooks, email forms and other similar, simply tasks.

4: PERL's primary disadvantage is the overhead on the server is very

high, as one process is created each time a routine is called, and the language is interpreted, which means the code is recompiled each time it is run.

5: For complex tasks, a server-side scripting language such as PHP or ASP is much preferred.

PHP

1: This language is, like ASP, used to get work done on the server.

2: PHP is similar in concept to ASP and can be used in similar circumstances. **3:** PHP is very efficient, allows

access to databases using products such as MySQL, and can be used to create very dynamic web pages.

SSI

1: Server-side includes are instructions and directives included in a webpage that the web server may analyze when the page is provided.

2: SSI refers to the servlet code that is embedded into the HTML code.

3: Not all web servers can handle SSI.

4: so you may read documents supported by a web server before utilizing SSI in your code.

VBScript

1: Microsoft VBScript (Visual Basic Script) is a general-purpose, lightweight and active scripting language developed by Microsoft that is modeled on Visual Basic.

2: Nowadays, VBScript is the primary scripting language for Quick Test Professional (QTP), which is a test automation tool.

Unit 3 (Introduction to HTML)

HTML FUNDAMENTALS

1: HTML – HyperText Markup Language – The Language of Web Pages on the World Wide Web.
2: HTML is a text formatting language. Its collection of ―TAGS, that are used to make web documents that are displayed using browsers on internet.
3: HTML is a platform independent language.
4: To display a document on the web it is essential to mark-up the different elements (headings, paragraphs, tables, and so on) of the document with the HTML tags.
5: To view a mark-up document, the user has to open the document in a browser.
6: A browser understands and interprets the HTML tags, identifies the structure of the document and makes decision about presentation of the document.

HTML tags

1: HTML markup tags are usually called HTML tags.

2: HTML tags are keywords surrounded by angle brackets like <html>

3: HTML tags normally come in pairs like and

4: The first tag in a pair is the start tag, the second tag is the end tag

5: Start and end tags are also called opening tags and closing tags

6: Tags are used to represent various elements of webpage like Header, Footer, Title, Images etc.

Tags are of two types:

Container Tags: These tags are always paired with closures tags are called

container tags.

Empty Tags: Tags, which have only opening and no ending, are called empty

tags/ standalone tag.

HTML Attributes

1: HTML attributes are additional pieces of information you can add to HTML elements to provide extra details about the element or modify its behavior. **2:** Attributes are added within the opening tag of an HTML element and are written in the format attribute_name="value".

3: Attribute values should always be enclosed in quotes.

4: Double style quotes are the most common, but single style quotes are also allowed.

STRUCTURE OF HTML CODE

1: HTML documents are structured into two parts, the HEAD, and the BODY.

2: Both of these are contained within the HTML element – it simply denotes its HTML document

3: The head contains information about the document that is not generally displayed with the document, such as its TITLE.

4: The BODY contains the body of the text Elements allowed inside the HEAD, such as TITLE, are not allowed inside the BODY, and vice versa.

Creating a Basic Starting Document:

page1.html

```
<HTML>
<HEAD>
<TITLE>My First Page</TITLE>
</HEAD>
<BODY>
Hello World !!!<br>
I am learning Web Technology.
</BODY>
</HTML>
```

Detailed structure of the HTML tags

Textbook page no: 32 to 40

INSERTING IMAGE

<IMG>

Tag type: Stadalone Function:

Images can be placed in a web page by using <IMG> tag.

The gif format is considered superior to the jpeg format for its clarity and ability to maintain the originality of an image without lowering its quality.

Example:

<img src="smiley.gif" alt="Smiley face" height="42" width="42" border=3>

Image Maps

1: Image maps are images, usually in gif format that have been divided into regions;

2: clicking in a region of the image cause the web surfer to be connected to a new URL.

3: Image maps are graphical form of creating links between pages.

4: There are two types of image maps:

Client side and server side.

5: Both types of image maps involve a listing of co-ordinates that define the mapping regions and which URLs those coordinates are associated with.

6: This is known as the map file.

Types of Shapes

Rect : used for squares and ordered shapes.

Circle : used for circles.

Poly : used for unordered shapes.

Number of coordenations for each shape:

Rect : 4 numbers for two corners

Circle : 3 numbers for the center & R

Poly : depends on the number of corners of the shape(2 numbers for each corner)

HOW TO MAKE A Hyper LINK

Anchor tag

1: The tags used to produce links are the <A> and </A>.

2: The <A> tells where the link should start and the </A> indicates where the link ends.

3: Everything between these two will work as a link.

4: The example below shows how to make the word here work as a link to yahoo.

Click <A HREF="http://www.yahoo.com">here</A> to go to yahoo.

Internal Links

1: Links can also be created inside large documents to simplify navigation.

2: Today‘s world wants to be able to get the information quickly. Internal links can help us meet these goals.

3: Select some text at a place in the document that we would like to create a link to, then add an anchor to link to like this:

<A NAME=“bookmark_name”></A>

4: The Name attribute of an anchor element specifies a location in the document that we link to shortly.

5: All NAME attributes in a document must be unique.

6: Next select the text that you would like to create as a link to the location created above.

<A HREF=“#bookmark_name”>Go To Book Mark</A>

HTML frameset Tag

1: The <frameset> tag in HTML is used to define the frameset.
2: The <frameset> element contains one or more frame elements.
3: It is used to specify the number of rows and columns in frameset with their pixel of spaces.
4: Each element can hold a separate document.
5: Note: The <frameset> tag is not supported in HTML5.
Syntax:

```
<frameset cols = "pixels|%|*">
```

Attributes: The list of frameset attributes are given below:

cols: The cols attribute is used to create vertical frames in a web browser. This attribute is basically used to define the no. of columns and their size inside the frameset tag.

Rows: The rows attribute is used to create horizontal frames in the web browser. This attribute is used to define the no. of rows and their size inside the frameset tag.

border: This attribute of frameset tag defines the width of the border of each frame in pixels. Zero value is used for no border.

frameborder: This attribute of frameset tag is used to specify whether a three-dimensional border should be displayed between the frames or not for this use two values 0 and 1, where 0 defines no border and value 1 signifies for yes there will be a border.

framespacing: This attribute of frameset tag is used to specify the amount of spacing between the frames in a frameset. This can take any integer value as a parameter which basically denotes the value in pixel.

HTML form

1: An HTML form is a document section containing content, labels, and special controls like checkboxes and menus.

2: Users modify controls (text, menus), then submit to an agent (web server).

3: Controls are named and defined by a name attribute within the FORM element.

4: Each control has an initial and current value. Initial value is set by attributes (except TEXTAREA, OBJECT).

5: Resetting the form reverts current values to initial.

6: On submission, controls with name/value pairs are "successful controls."

7: These pairs are sent with the form for processing to an agent, like a web server.

Text Fields

```
<input type = "text" /> defines a one-line input field that a user can
enter text into:
<form>
First name: <input type = "text" name = "firstname" /><br /> Last
name: <input type =
"text" name = "lastname" /></form>
```

Password Field

```
<input type = "password" /> defines a password field:
<form>
Password: <input type = "password" name = "pwd" /></form>
```

Radio Buttons

1: The HTML <input type=”radio”> is used to define a Radio Button.

2: Radio Buttons are used to let the user select exactly one option from a list of predefined options.

3: Radio Button input controls are created by using the “input” element with a type attribute having the value as “radio”.

example /

<Checkbox> Tag

1: The HTML <checkbox> tag is used to define the square boxes.

2: It is a form element which allows users to select one or more options from the given options.

3: It is created by the type attribute of the <input> element as shown in the following syntax:

```
<input type="checkbox" name="field name" value="Initial value"
checked="yes">
```

Example

Table tag

A table can be inserted in a wep page using table tag.

The <TABLE></TABLE> element has four sub-elements:

1. Table Row<TR></TR>.

2. Table Header <TH></TH>.

3. Table Data <TD></TD>.

4. Caption <CAPTION></CAPTION>.

The table row elements usually contain table header elements or table data elements.

Tables Attributes

BGColor: Some browsers support background colors in a table.

Width: you can specify the table width as an absolute number of pixels or a

percentage of the document width. You can set the width for the table cells as well.

Border: You can choose a numerical value for the border width, which specifies the border in pixels.

CellSpacing: Cell Spacing represents the space between cells and is specified

in pixels.

CellPadding: Cell Padding is the space between the cell border and the cell

contents and is specified in pixels.

Align: tables can have left, right, or center alignment.

Background: Background Image, will be titled in IE3.0 and above.

Table Data and Table Header Attributes

Colspan: Specifies how many cell columns of the table this cell should span.

Rowspan: Specifies how many cell rows of the table this cell should span.

Align: cell data can have left, right, or center alignment.

Valign: cell data can have top, middle, or bottom alignment.

Width: you can specify the width as an absolute number of pixels or a

percentage of the document width.

Height: You can specify the height as an absolute number of pixels or a

percentage of the document height.

Special Things to Note

1: TH, TD and TR should always have end tags: Although the end tags are formally optional, many browsers will mess up the formatting of the table if you omit the end tags. In particular, you should always use end tags if you have a TABLE within a TABLE -- in this situation, the table parser gets hopelessly confused if you don't close your TH, TD and TR elements.

2: A default TABLE has no borders:

By default, tables are drawn without border lines. You need the BORDER

attribute to draw the lines.

3: By default, a table is flush with the left margin:

TABLEs are plopped over on the left margin. If you want centered tables, You

can either: place the table inside a DIV element with attribute ALIGN="center".

Most current browsers also support table alignment, using the ALIGN attribute.

Allowed values are "left", "right", or "center", for example: <TABLE ALIGN="left">.

The values "left" and "right" float the table to the left or right of the page, with text flow allowed around the table. This is entirely equivalent to IMG alignment

Unit 4 (Cascading Style Sheets)

CASCADING STYLE SHEETS

1: CSS (Cascading Style Sheets) is a coding language used to control the visual presentation of web content written in HTML and XML.
2: It enables web designers to define how elements like text, images, and layout should be displayed on a webpage.
3: CSS allows for precise control over aspects such as colors, fonts, spacing, and positioning.
4: By separating design from content, it promotes consistency and easier maintenance across a website.
5: Styles can be applied inline within HTML tags, embedded in HTML documents, or linked externally to multiple pages.
6: This separation of concerns enhances the flexibility and adaptability of web designs.

<style> element

1: The `<style>` element in HTML is used to define internal CSS (Cascading Style Sheets) within an HTML document.

2: It allows you to embed CSS rules directly within the HTML file, rather than linking to an external CSS file.

3: This can be useful for small projects or when you want to encapsulate styles specific to a particular HTML page.

4: The `<style>` element is typically placed within the `<head>` section of the HTML document.

5: Inside the `<style>` element, you write CSS rules that will apply to the HTML elements within the same document.

6: This approach helps maintain the separation between content (HTML) and presentation (CSS).

---

6 rules of style definitions

1: Every CSS definition has to have a selector and a declaration. The declaration follows the selector and uses curly braces.

2: The declaration consists of one or more properties separated with a semicolon.

3: Every property has a name, colon and a value.

4: A property can have multiple values separated with a comma (e.g. "Verdana, Arial, Franklin Gothic Book").

5: Along with a value, can also be a unit of measure (e.g. "9pt", where "pt" stands for points). No space is allowed between the value and the unit.

6: When writing CSS code, you can use whitespaces as needed – new lines, spaces, whatever makes your code more readable.

Inline style

1: Inline styles are styles that are applied to a specific element within the body section of the webpage.

2: The style will be applied to that individual element only rather than to the entire page (internal style) or across all linked pages (external style sheet).

3: Here's an example:

```
<p style="color: red; font-size: 18px;">This is a red text with font size 18px.</p>
```

4: In this example, the style attribute is added to the <p> (paragraph) element, and the styles are defined directly within the attribute using CSS property-value pairs.

---

embedded style

1: It allows us to define styles for a particular HTML document as a whole in one place.

2: This is done by embedding the **<style> </style>** tags containing the CSS properties in the head of our document.

3: Embedded style sheets are particularly useful for HTML documents that have unique style requirements from the rest of the documents in our project.

4: However, if the styles need to be applied across multiple documents, we should link to an external style sheet instead of using individual embedded style sheets.

5: Using embedded stylesheets holds a distinct advantage over inline styles which only allow us to address one HTML element at a time.

Example:

```
<!DOCTYPE html>
<html>
<head>
    <style>
        p {
            color: blue;
            font-size: 16px;
        }
    </style>
</head>
<body>
    <p>This is a blue text with font size 16px.</p>
</body>
</html>
```

In this example, the <style> element is placed within the <head> section of the HTML document. The CSS rules defined within the <style> element will be applied to all <p> (paragraph) elements in the <body> section.

External CSS

1: An external style sheet is ideal when the style is applied to many pages.

2: The style information is placed in a separate document with the file extension .css.

3: With an external style sheet, we can change the look of an entire Web site by changing the corresponding style information file.

4: Each page must link to the style sheet using the <link> element, which usually goes within the <head> section.

5: An external style sheet can be written in any text editor and should be saved with the file extension .css.

Naming font families using CSS

1: CSS provides a mechanism for rendering font families in a browser if those fonts are installed on a user‘s system.

2: This is accomplished by creating either an inline style on an element such as a td or span element, or by creating a class rule selector within the style element.

3: Either way, the syntax is the same, with a list of font family names, each separated by a comma,contained within a set of braces:

```
font-family {Arial, Helvetica, sans-serif;}
```

4: The browser will look first for the Arial font in the preceding example, then the Helvetian font, then the —"default" sans-serif font, which is whatever sans-serif font the user‘s operating system defaults to.

Establishing font styles in html

1: To establish font styles in HTML, you can use CSS to control the appearance of text on your webpage.

2: There are various ways to set font styles, including inline styles, embedded styles, and external stylesheets.

3: Here's an example of using embedded styles to establish font styles:

```
<!DOCTYPE html>
<html>
<head>
    <style>
```

```
QQ q        body {
            font-family: Arial, sans-serif;
        }

        h1 {
            font-size: 24px;
            color: #333;
        }

        p {
            font-size: 16px;
            line-height: 1.5;
        }
    </style>
</head>
<body>
    <h1>Welcome to My Website</h1>
    <p>This is a paragraph of text with established font styles.</p>
</body>
</html>
```

4: The font-family property in the body selector sets the default font for the entire webpage.
5: In this case, it's set to Arial or a generic sans-serif font.
6: The font-size property in the h1 and p selectors sets the font size for headings and paragraphs.

Bolding fonts by changing font weight

1: Font weight determines the thickness of a font's stroke.

2: A font's weight is denoted by values ranging from 100 (thin) to 900 (heavy), with intermediate weights in between.

3: These values are used numerically or with keywords like bold or normal.

4: The keyword bold equates to the numeric value 700.

5: For instance, in CSS rules within a style element, p {font-weight: normal} sets default weight, while `p.bold {font-weight: 900}` defines a heavy weight. Font weight influences text appearance and can be manipulated using specific values or keywords, offering a range of design possibilities.

Background color and image

1: The CSS background element allows us to add a background color or image to our Webpage.

2: For example, we may like to use a dark color to set a background against light-colored paragraphs to create an effect of sidebars or offsetting text for

emphasis.

3: The CSS element <background-color> takes the general format <b style="background-color:#rrggbb">

Positioning a background image

1: You can further control the position at which a background image begins to tile on your Webpage.

2: This is all done by the CSS element background-position.

3: It takes the general form <body style="background-image:url (bg_image.gif) background-position: x y">

4: where x y represents the position of the image.

CSS PSEUDO-CLASSES

1: CSS pseudo-classes are used to select and style elements that are not directly present in the HTML structure.

2: They begin with a colon (:) and target specific states or positions of elements.

3: Examples include `:hover`, `:active`, `:focus`, `:nth-child()`, `:first-child`, and `:last-child`.

4: These classes allow us to apply styles based on user interactions or element positions, enhancing the visual experience and interactivity of our webpage.

Anchor pseudo classes

Anchor pseudo-classes in CSS are used to style links (<a> elements) based on their states or positions. Here are a few common anchor pseudo-classes:

:link: Targets unvisited links.

:visited: Targets visited links.

:hover: Targets links when the mouse pointer hovers over them.

:active: Targets links when they are clicked or activated.

Pseudo-classes and CSS Classes

1: Combining pseudo-classes and CSS classes allows us to apply specific styles to elements based on both their inherent characteristics (pseudo-classes) and their assigned classes.

2: For example, we can target a specific state of an element, such as when it's being hovered over or clicked (pseudo-class), and then further refine the styling by specifying a class that the element belongs to.

3: This combination provides powerful and precise control over how elements are displayed on our webpage.

The id Selector

1: The id selector is used to specify a style for a single, unique element.

2: The id selector uses the id attribute of the HTML element, and is defined with a "#".

3: The style rule below will be applied to the element with id="para1":

Example

```
#para1
{
text-align:center;
color:red;
}
```

The class Selector

1: The class selector is used to specify a style for a group of elements.

2: Unlike the id selector, the class selector is most often used on several elements.

3: This allows you to set a particular style for any HTML elements with the same class.

4: The class selector uses the HTML class attribute, and is defined with a "."

5: In the example below, all HTML elements with class="center" will be center-aligned:

Example

```
.center {text-align:center;}
```

Unit 5 (Introduction to Client Side Scripting)

What is Scripting Language?

1: A scripting language is a type of programming language that is often used for writing scripts or small programs to automate tasks, manipulate data, or control other software applications.
2: Scripting languages are typically interpreted rather than compiled, allowing for rapid development and easy debugging.
3: They are commonly used in web development, system administration, and other areas where quick code execution is important.
4: Examples of scripting languages include Python, JavaScript, Ruby, and PowerShell.

What is meant by script in HTML?

1: It defines how locally executable scripts may be used in a webpage.

2: A particular client- side application, such as a web browser, may support several script languages.

3: Script code may be executed as the document loads or at a later

time.

4: Script code can be written directly in the HTML document inside: SCRIPT elements.

Client side scripting

1: Client-side scripting refers to the execution of scripts on the user's web browser rather than on the web server.

2: This scripting is used to enhance the interactivity and functionality of web pages.

3: Commonly, it involves the use of languages like JavaScript to manipulate the Document Object Model (DOM), which represents the structure of a webpage.

4: Client-side scripts allow developers to create dynamic content, validate user input, handle user interactions, and update the webpage without requiring a full page reload.

5: This approach leads to a more responsive and interactive user experience.

Server side scripting

1: Server-side scripting refers to the execution of scripts on the web server rather than on the user's browser.

2: This scripting is used to generate dynamic content and perform operations on the server before sending the resulting content to the user's browser.

3: Server-side scripts handle tasks like retrieving data from databases, processing user input, generating HTML content, and performing complex calculations.

4: This approach offers better security, as sensitive operations are not exposed to users.

5: However, it can result in slower response times compared to client-side scripting due to the need for server processing before delivering the final content to the user.

Languages used in server side scripting

Python, Java, PHP, Ruby, C#

Languages used in client side scripting

Javascript, HTML, CSS

Unit 6 (Javascript)

Javascript

1: Javascript is object based scripting language based on c++.
2: Javascript was basically designed to add interactivity in HTML pages and is directly embedded into HTML.
3: Javascript is open source.
4: Javascript is interpreted language i.e. it is not precompiled before execution.
5: As javascript is interpreted, it is platform independent.
6: Every object has properties and methods. Property is value(s) associated with an object. Methods are actions associated with an object.
7: Modern javascript security is based upon Java. Scripts downloaded are isolated from the operating system and then executed.

JAVASCRIPT OPERATORS

Category	Operator	Name/Description	Exan
Arithmetic	+	Addition	3+2
	-	Subtraction	3-2
	*	Multiplication	3*2
	/	Division	10/5
	%	Modulus	10%5
	++	Increment and then return value	X=3; +
		Return value and then increment	X=3; X
	--	Decrement and then return value	X=3; -
		Return value and then decrement	X=3; X
Logical	&&	Logical “and” evaluates to true when both operands are true	3>2 && 5>3
	||	Logical “or” evaluates to true when either operand is true	3>1 || 2
	!	Logical “not” evaluates to true if the operand is false	3!=2
Comparison	==	Equal	5==9
	!=	Not equal	6!=4
	<	Less than	3<2
	<=	Less than or equal	5<=2
	>	Greater than	4>3
	>=	Greater than or equal	4>=4
String	+	Concatenation(join two strings together)	“A”+”

JavaScript Comments

1: JavaScript comments can be used to make the code more readable.

2: Comments can be added to explain the JavaScript.

3: Comments can be added at the end of a line.

4: Single line comments start with //.

5: Multi line comments start with /* and end with */.

6: The comment is used to prevent the execution of a single code line or a code block. This can be suitable for debugging.

Javascript Variables (Var statement)

1: Variables in JavaScript serve as "containers" for holding information, which can be values or expressions.

2: They can be given concise names like 'x' or descriptive ones like 'MyName.' **3:** Following specific rules, variable names are case-sensitive and must begin with a letter or underscore.

4: The process of introducing variables in JavaScript is often termed "declaring."

5: Declaration involves using the 'var' keyword, such as 'var x;' which initializes an empty variable.

6: Alternatively, values can be assigned during declaration, like 'var x = 10;' giving 'x' a value of 10.

7: Variables declared within functions are local and limited to that function's scope.

8: Conversely, variables declared outside functions are global, accessible across scripts and functions, but destroyed when the page closes.

---

if statement

This statement is used to execute some

code only if aspecified condition is true.

Syntax:

```
if(condition)
{
```

```
code to be executed if condition is true
}
```

If...else statement

This statement is used to execute some

code if the condition is true and another code if the condition is false.

```
Syntax:
if (condition)
{
code to be executed if condition is true
}
else
{
code to be executed if condition is not true
}
```

If...else if....else statement

This statement is used to select one of many blocks of code to be executed.

```
if (condition1)
{
code to be executed if condition1 is true
}
else if (condition2)
{
code to be executed if condition2 is true
}
else
{
code to be executed if neither condition1
nor condition2 is true
}
```

---

switch statement

This statement is another way to select

one of many blocks of code to be executed.

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

---

for Loop

for - loops through a block of code a specified number of times. It can be used only when it is known in advance how many times we have to run the loop.

Sggggvgggggfgggfgfcgfvgyntax:

```
for (
e=variable+increment)
{
code to be executed
}
```

---

while loop

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

syntax:

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

Do While loop

Do While loop is a variation to the while loop. In this case, block will be executed at least once, as the statements are executed before the condition is tested. Syntax for do while is as follows –

```
do
{
code to be executed
}
while (var<=endvalue);
```

Break and Continue statement in javascript

1: The break statement will break the loop and continue executing the code that follows after the loop (if any).

2: The continue statement will break the current loop and continue with the next value.

Function

1: Function is a segment of program that performs a given task.

2: A function contains code that will be executed by an event or by a call to

the function.

3: You may call a function from anywhere within a page (or even from other

pages if the function is embedded in an external javascript file).

4: Functions can be defined both in the <head> and in the <body> section of

a document.

5: However, to assure that a function is read/loaded by the browser before it is called, it should be put functions in the <head> section.

6: Syntax of the function is as follows –

```
functionfunctionname(var1,var2,...,varX)
{
some code
}
```

Rules for function

1: Function always includes parenthesis after the name of function '()‘

2: Function calls are case-sensitive as javascript is case-sensitive.

3: The return statement is used to specify the value that is returned from the function. So, functions that are going to return a value must use the return statement.

4: If a variable is declared using "var", within a function, the variable can only be accessed within that function.

5: If a variable is declared outside a function, all the functions on your page can access it.

6: The lifetime of these variables starts when they are declared, and ends when the page is closed.

Composite Data types

In JavaScript, composite data types allow us to group together multiple values or entities into a single entity. Here are some common composite data types:

Arrays: Arrays are ordered lists that can hold various data types. They are defined using square brackets and can store multiple values under a single variable name.

Objects: Objects are collections of key-value pairs, where each key is a string and each value can be any data type, including arrays and other objects. Objects are defined using curly braces.

Functions: Functions are reusable blocks of code that can be executed when called. They can take arguments as inputs and return values as outputs.

Classes: JavaScript supports object-oriented programming through classes. A class is a blueprint for creating objects with shared properties and methods.

Maps and Sets: Maps are collections of key-value pairs, similar to objects, but they allow any data type as keys and maintain the insertion order. Sets are collections of unique values.

JSON:JSON (JavaScript Object Notation) is a text-based data format that closely resembles JavaScript object syntax. It's commonly used for data exchange between a server and a web application.

Primitive data types

Primitive data types in JavaScript are basic data types that represent single values. They are immutable, meaning their values cannot be changed directly. Here are the main primitive data types:

String: Represents a sequence of characters and is enclosed in single (' ') or double (" ") quotes.

Number: Represents both integer and floating-point numbers. JavaScript doesn't distinguish between these two types.

Boolean: Represents a true or false value. Used for logical operations and conditions.

Undefined: Represents a variable that has been declared but hasn't been assigned a value yet.

Null: Represents an intentional absence of any value. It's often used to indicate the absence of an object or value.

Date Object

1: The Date object is used to work with dates and times.

2: Date objects are created with new Date().

3: There are four ways of instantiating a date.

Var d = new Date();

var d = new Date(milliseconds);

var d = new Date(dateString);

var d = new Date(year, month, day, hours, minutes, seconds, milliseconds);

Some javascript predefined methods are :

1: getDate() Returns the day of the month (from 1-31)

2: getDay() Returns the day of the week (from 0-6)

3: getFullYear() Returns the year (four digits)

4: getHours() Returns the hour (from 0-23)

Math object –

The Math object allows you to perform mathematical tasks.

The Math object includes several mathematical constants and methods.

1: round() – rounds the number to nearest integer value.

2: random() - returns a random number between 0 and 1.

3: max() - returns the number with the highest value of two

specified numbers.

4: min() - returns the number with the lowest value of two specified numbers.

Array

1: An Array is an ordered collection of data values.

2: In JavaScript, an array is just an object that has an index to refer to its contents.

3: In other words, the fields in the array are numbered, and you can refer to the number position of the field.

4: JavaScript does not have multi dimensional arrays, but we can nest them, which is to say, an array element can contain another array.

5: We access them listing the array numbers starting for the outmost array and working inward.

DOCUMENT AND ASSOCIATED OBJECTS

1: Each HTML document loaded into a browser window becomes a Document object.

2: The Document object provides access to all HTML elements in a page, from within a script.

3: The Document object is also part of the Window object, and can

be accessed through the 'window.document‘ property.

4: Document object is part of document object model.

5: This model has a fixed hierarchy, where the topmost object in the hierarchy is Browser itself.

Document object has the following properties and methods

Properties –

1. cookieReturns all name/value pairs of cookies in the document.

2. documentModeReturns the mode used by the browser to render the document.

3. domain Returns the domain name of the server that loaded the document.

4. lastModifiedReturns the date and time the document was last modified.

5. readyStateReturns the (loading) status of the document.

6. referrer Returns the URL of the document that loaded the current document.

7. title Sets or returns the title of the document.

8. URL Returns the full URL of the document.

Methods –

1. close() Closes the output stream previously opened *withdocument.open/*().
2. getElementById() Accesses the first element with the specified id.
3. getElementsByName() Accesses all elements with a specified name.
4. getElementsByTagName() Accesses all elements with a specified tag name.
5. open() Opens an output stream to collect the output from document.write() or document.writeln()
6. write() Writes HTML expressions or JavaScript code to a document.
7. writeln() Same as write(), but adds a newline character after each statement.

What are events?

1: Events are actions that can be detected by JavaScript.

2: Every element on a webpage has certain events which can trigger a JavaScript.

3: For example, we can use the onClick event of a button element to indicate that a function will run when a user clicks

on the button.

Examples of events:

1. A mouse click

2. A webpage or an image loading

3. Mousing over a hot spot on the webpage

4. Selecting an input field in an HTML form to Submitting an HTML form

5. A keystroke.

4: Events are normally used in combination with functions, and the

function will not be executed before the event occurs!

Following is the list of events used by various javascript objects and when are these events triggered

onfocus	An element gets focus
onkeydown	A keyboard key is pressed
onkeypress	A keyboard key is pressed or held dow
onkeyup	A keyboard key is released
onload	A page or image is finished loading
onmousedown	A mouse button is pressed
onmousemove	The mouse is moved
onmouseout	The mouse is moved off an element
onmouseover	The mouse is moved over an element
onmouseup	A mouse button is released
onmove	The position of top left corner of an ob is moved.
onresize	A window or frame is resized

Event handlers

Event handlers are functions or pieces of code that are designed to respond to specific events that occur within a software application or system. An event can be any notable occurrence, such as user interactions, system notifications, data changes, or external inputs. Event handlers are commonly used in programming to manage these events and execute appropriate actions or logic in response.

Here's how event handlers typically work:

1. **Event Registration:** In many programming languages and frameworks, you can register event handlers for specific events. This is often done by associating a function or method with an event name. For example, you might register a "click" event handler for a button.
2. **Event Triggering:** When the associated event occurs, the programming environment triggers the corresponding event handler function. This might be due to user interactions (like clicking a button), changes in data, or other system events.
3. **Event Handling:** The event handler function is then executed. This function contains the logic that defines how your application should respond to the event. It can involve updating the user interface, modifying data, performing calculations, or any other necessary actions.

Unit 7 (XML)

XML

1: XML (extensible Markup Language) is a language in which other languages are created.
2: In XML, data is "marked up" with tags, similar to HTML tags.
3: The latest version of HTML, called XHTML, is an XML-based language, which follows the syntax rules of XML.
4: XML was designed to describe and store data.
5: This data might be intended to be read by people or by machines.
6: It can be highly structured data such as data typically stored indatabases or spreadsheets, or loosely structured data, such as data stored in letters or manuals.
7: XML tags are not predefined in XML. We have to define our own tags.
8: XML uses a DTD (Document Type Definition) to formally describe the data.

The main difference between XML and HTML

1: XML is not a replacement for HTML.

XML and HTML were designed with different goals:

a. XML was designed to describe data and to focus on what data is.

b. HTML was designed to display data and to focus on how data looks.

2: HTML is about displaying information, XML is about describing information.

XML benefits

Structured Data: XML allows you to create structured data that is easy to understand and organize, making it useful for representing various types of information.

Platform Independence: XML is a platform-independent format, meaning it can be used on different operating systems and programming languages.

Human-Readable: XML is human-readable, making it easier for developers to understand and debug.

Customizable: XML is extensible, meaning you can define your own elements and attributes, tailoring it to your specific needs.

Anatomy of an XML document

1: Declaration: The XML declaration is optional but recommended. It specifies the version of XML being used and optionally the character encoding. It appears at the very beginning of the document and looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
```

2: Root Element: Every XML document has a single root element that encapsulates all other elements. It's enclosed within angle brackets and represents the highest level of the document's hierarchy.

```
<rootElement>
 <!-- Other elements and content go here - >
</rootElement>
```

3: Elements: Elements are the building blocks of an XML document. They consist of opening and closing tags and can contain other elements, text, attributes, or a combination of these.

```
<elementName attribute="value">
    <!-- Content goes here -->
</elementName>
```

4: Attributes: Elements can have attributes, which provide additional information about the element. Attributes are placed within the opening tag and follow the format attribute="value".

```
<person name="John" age="30"/>
```

5: Text Content: Elements can also contain text content. Text content is placed between the opening and closing tags of an element.

```
<description>This is the text content.</description>
```

6: Comments: Comments are used to provide annotations or explanations within the XML document. They are enclosed in <!-- and -->.

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

7: Whitespace: XML ignores whitespace (spaces, tabs, line breaks) outside of the text content. However, whitespace within text content is preserved.

8: Hierarchy: Elements can be nested within each other, forming a hierarchical structure that represents the relationships between different pieces of data.

```
<book>
    <title>XML Basics</title>
    <author>John Doe</author>
</book>
```

XML Syntax Rules

XML has relatively straightforward, but very strict, syntax rules. A document that follows these syntax rules is said to be well-formed :

1. There must be one and only one document element.

2. Every open tag must be closed.

3. If an element is empty, it still must be closed.

O Poorly-formed: <TAG>

o Well-formed: <TAG></TAG>

o Also well-formed: <TAG />

4. Elements must be properly nested:

o Poorly-formed: <A><B></A></B>

o Well-formed: <A><B></B></A>

5. Tag and attribute names are case-sensitive.

6. Attribute values must be enclosed in single or double quotes.

Special Characters

There are five special characters that cannot be included in XML documents. These characters are replaced with

predefined entity references as shown in the table below:

Character	Entity Reference
<	&lt;
>	&gt;
&	&amp;
"	&quot;
'	&apos;

XML DTDs

1: A Document Type Definition (DTD) is a type of schema.

2: The purpose of DTDs is to provide a framework for validating XML documents.

3: By defining a structure that XML documents must conform to, DTDs allow different organizations to create shareable data files.

Well-formed vs. Valid DTD

1: A well-formed XML document is one that follows the syntax rules

described in "XML Syntax Rules".

2: A valid XML document is one that conforms to a specified structure.

3: For an XML document to be validated, it must be checked against a schema (document that defines the structure for a class of XML documents).

4: XML documents that are not intended to conform to a schema can be well-formed, but they cannot be valid.

Child Elements

1: When defining child elements in DTDs, you can specify how many times those elements can appear by adding a modifier after the element name.

2: If no modifier is added, the element must appear once and only once.

Other Elements

1: The other elements are declared in the same way as the document element - with the <!ELEMENT> declaration.

2: The ARTISTS DTD declares three additional elements.

Choice of Elements

1: It is also possible to indicate that one of several elements may appear as a child element. E.g., the declaration below indicates that an IMG element may have a child element NAME or a child element ID, but not both.

```
<!ELEMENT IMG (NAME|ID)>
```

Empty Elements

Empty elements are declared as follows.

```
<!ELEMENT IMG EMPTY>
```

Mixed Content

Sometimes elements can have elements and text mixed.

E.g., the following declaration is for a BODY element that may contain text in addition to any number of LINK and IMG elements.

```
<!ELEMENT BODY (#PCDATA | LINK | IMG)*>
```

Location of Modifier

1: The location of modifiers in a declaration is important.

2: If the modifier is outside of a set of parentheses, it applies to the group;

whereas, if the modifier is immediately next to an element name, it applies only to that element.

Using Parentheses for Complex Declarations

Element declarations can be more complex than the examples above. E.g., you can specify that a PERSON element either contains a single NAME element or a FIRSTNAME and LASTNAME element. To group elements, put them in parentheses as shown below:

```
<!ELEMENT PERSON (NAME | (FIRSTNAME, LASTNAME))>
```

XML SCHEMAS

1: It is an XML-based language used to create XML-based languages and data models.

2: It defines element and attribute names for a class of XML documents.

3: It specifies the structure that those documents must adhere to and the type of content that each element can hold.

Why need XML Schema / Limitations of DTDs

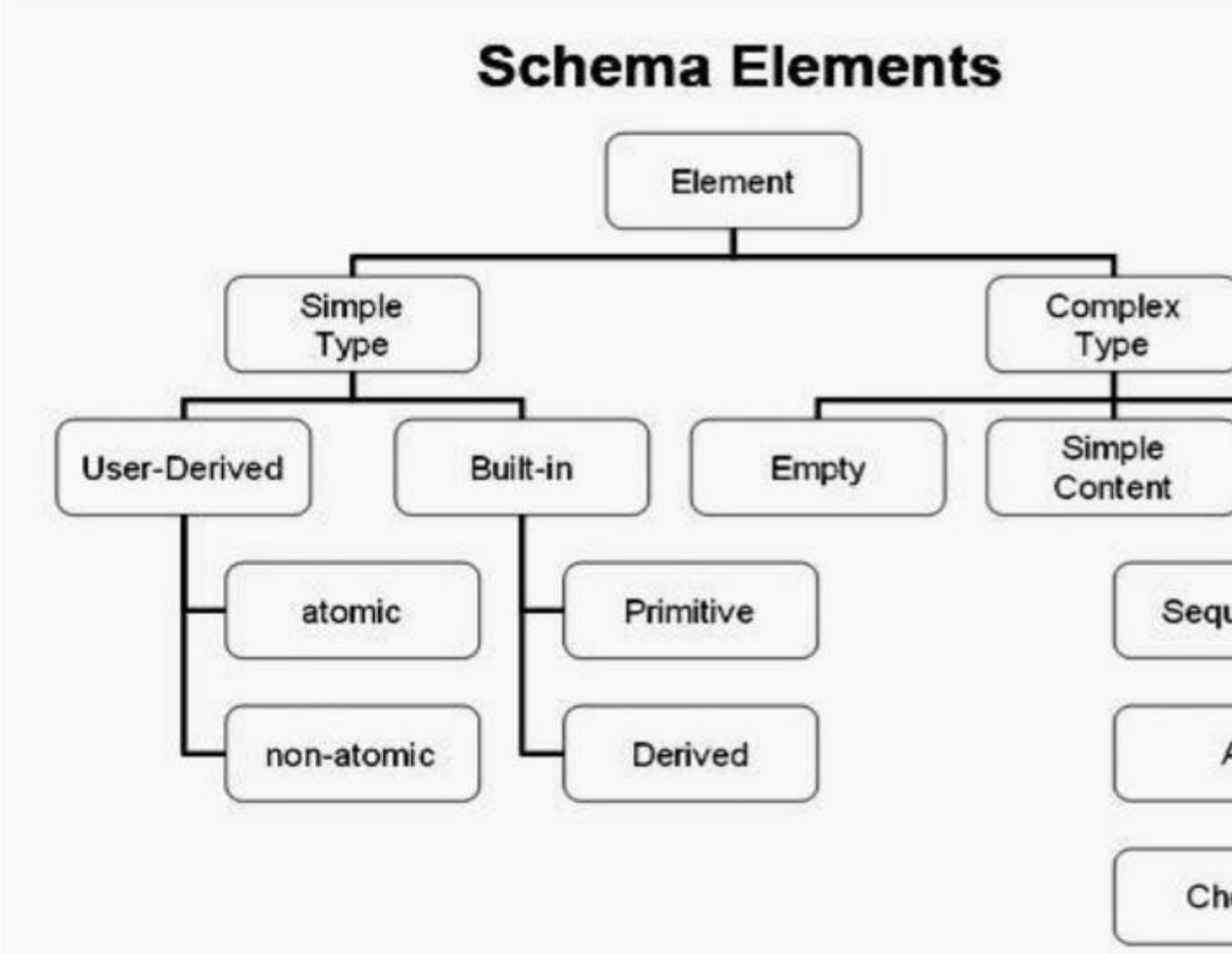
1: DTDs do not have built-in data types.

2: DTDs do not support user-derived data types.

3: DTDs allow only limited control over cardinality (the number of occurrences of an element within its parent).

4: DTDs do not support Namespaces or any simple way of reusing or importing other schemas.

Schema Elements



Simple type elements in Schema

1: These elements can only contain text.

2:They cannot have child elements or attributes.

3: All the built-in types are simple types (E.g., XS:STRING).

4: Schema authors can derive simple types by restricting another simple type.

5: E.g., an email type could be derived by limiting a string to a specific pattern (that includes :'@‘, '. ‘ , '_' etc.)

6: Simple types can be atomic (E.g., strings and integers) or non-atomic (E.g., lists).

Complex Type

1: These elements can contain child elements and attributes as well as text.

2: By default, complex-type elements have child elements.

3: These elements can only contain text. But they are different from simple type elements in that they have attributes.

4: These elements can be empty, but they have may have attributes.

5: These elements may have mixed content - a combination of text and child elements.

XSL

1: XSL (Extensible Stylesheet Language) is a language used to define the presentation and transformation of XML documents.

2: It consists of two main components: XSLT (XSL Transformations) for converting XML content into various formats, and XSL-FO (XSL Formatting Objects) for specifying the layout and formatting of the transformed content. **3:** XSLT employs templates and rules to define how data should be manipulated and presented, while

XSL-FO handles the arrangement of elements on the output medium, such as paper or screen.

4: XSL enables developers to separate content from presentation, making it a powerful tool for managing and customizing XML-based data representation.

XSLT

1: XSLT (Extensible Stylesheet Language Transformations) is a language used for transforming XML documents into different formats or structures.

2: It allows you to apply templates and rules to XML data, enabling you to extract, manipulate, and rearrange information within an XML document.

3: XSLT uses a declarative approach, where you define how the transformation should occur rather than writing procedural code.

4: It's commonly used for tasks like converting XML to HTML, generating reports, and extracting specific data from XML files.

5: XSLT's power lies in its ability to separate content from presentation, making it a key technology for XML-based data processing.

Unit 8 (Website Design Concepts)

What Is Good Web Design?

Good web design involves a combination of aesthetics, usability, and functionality to create a positive and engaging user experience. Here are some key principles of good web design:

Visual Appeal: The design should be visually pleasing, with a balanced use of color, typography, and images that align with the brand identity and overall theme.

User-Centered Design Prioritize the needs and preferences of the target audience, ensuring easy navigation, clear content hierarchy, and intuitive interactions.

Responsive Design: Create a design that works well on various devices and screen sizes, ensuring a consistent experience for users accessing the site from desktops, tablets, and smartphones.

Fast Loading Speed: Optimize images, use efficient coding practices, and minimize the use of heavy elements to ensure quick loading times, preventing user frustration.

Hierarchy and Readability: Arrange content with a clear hierarchy, using headings, subheadings, and appropriate spacing to make the text easy to read and understand.
.
Cross-Browser Compatibility: Test the website across different browsers to ensure it functions correctly and looks consistent regardless of the browser being used.

Performance Optimization: Optimize code, minimize HTTP requests, and leverage browser caching to improve overall performance.

Mobile-Friendly Design: Given the prevalence of mobile browsing, design with mobile users in mind, using responsive design techniques or mobile-specific layouts.

---

10 Rules of Web Design

Creating effective websites involves following these 10 key rules:

Simplicity: Avoid clutter on pages to enhance user experience and ease of use.

Design: A visually appealing website creates a positive first impression.

Intuitive Navigation: Organize content logically to help visitors find what they need easily.

Consistency: Maintain a consistent design throughout the site for seamless navigation.

Color Harmony: Choose colors carefully, ensuring readability and a pleasing aesthetic.

Responsiveness: Design for various devices using responsive techniques like CSS media queries.

Cross-Browser Compatibility: Test the site across browsers to ensure uniform rendering.

Error Checking: Regularly inspect for typos, broken links, and image issues.

Custom Code: Writing your code offers better control and understanding when issues arise.

Quality Content: Combine excellent design with unique, engaging content to create a worthwhile website experience.

Types of websites

There are various types of websites designed to cater to different purposes and audiences. Here are some common types:

1.E-commerce Websites: Platforms for online buying and selling of products and services.

2.Portfolio Websites: Showcases an individual's or a company's work, often used by artists, photographers, designers, and freelancers.

3.Blogging Websites: Focuses on written content, personal opinions, stories, and articles.

4.Corporate Websites: Represents businesses and provides information about their products, services, and company details.

5.News/ Websites: Provides timely news articles and updates on various topics.

6.Educational Websites: Offers educational content, courses, tutorials, and resources.

7.Entertainment Websites: Hosts multimedia content like videos, games, and music for entertainment purposes.

8.Social/ Media Websites: Connects users with friends, family, and others, allowing them to share content and interact online.

9.Community/ Forums: Online platforms for discussions and sharing information on specific topics.

10. Government Websites: Provides official information and services offered by government agencies.