INTERNET TECHNOLOGIES

UNIT – 3

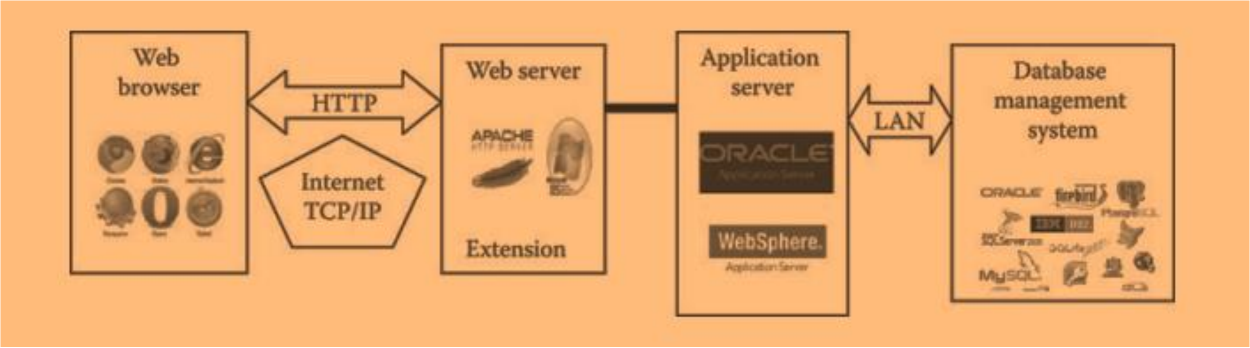
WEB DEVELOPMENT BASICS

Web development is the building and maintenance of websites, it is the work that happens behind the scenes to make a website look great, work fast and perform well with a seamless user experience. Web developers, do this by using a variety of coding languages.

Elements of Web Development: Web development itself is complex, as it involves much more in terms of concepts, tools, and technologies. Though the set of tools and technologies is constantly expanding, there are essential tools and technologies that every web developer should be familiar with:

Browsers: As the interpreters of the Web, browsers request information and on receiving a response, display the web page in a human-readable format.

Example: Google Chrome (currently, the most popular browser), Safari (Apple's web browser)



Web app client-server architecture

- ➢ Firefox (an open-source browser supported by the Mozilla Foundation)
- ➢ Internet Explorer (Microsoft's browser)
- ➢ HTML: The Hypertext Markup Language provides structure to web pages and web sites so the web browser knows what to display.
- ➢ **CSS:** Cascading Style Sheets enable web designers to control colours, fonts, animations, and transitions on the web. CSS makes the web look good.
- ➢ **Programming languages:** Programming languages allows to communicate with computers and tell them what to do. Among the many available, there is no concept of one being better than the other. It's the programmer's capabilities and application requirements that typically drive this choice. The prominent ones are:
  - JavaScript: Supported by all web browsers.
  - Python: Used by the Django framework.
  - Ruby: Used by the Ruby on Rails framework.
  - PHP: Used by WordPress, Joomla Content Management Systems.

  - Objective-C: Led by Apple, this is the programming language behind iOS.
  - Swift: Apple’s newest programming language.
  - Java: Used by Android (Google) and many desktop application.

- **Frameworks:** Frameworks are built to make building and working with programming languages easier. Frameworks typically take all the difficult, repetitive tasks in setting up a new web application and either do them for you or make them easier to complete. Examples include:
  - Meteor: A full-stack (front- and back-end) JavaScript framework.
  - Node.js: A server-side JavaScript framework.
  - Ruby on rails: A full-stack framework built using ruby.
  - Django: A full-stack framework built using python.
  - Ionic: A mobile framework.
  - Bootstrap: A UI (user interface) framework for building with HTML,
    CSS, and JavaScript.
  - Content management systems (CMS): WordPress, Joomla, and Drupal are popular solutions that support the creation and modification of digital content.
  - NET: A full-stack framework built by Microsoft
  - Angular.js: A front-end JavaScript framework

- **Libraries:** Libraries are groups of code snippets that enable a large amount of functionality without having to write it all. Libraries typically ensure code is efficient and works well across browsers and devices. Examples include jQuery and Underscore.

- **Databases:** Databases are where you store data. A database is like a bunch of filing cabinets, with folders filled with files. They come mainly in two types: SQL and NoSQL.
  1. SQL provides more structure, which helps with making sure all the data is correct and validated.
  2. NoSQL provides a lot of flexibility for building and maintaining applications. Examples include:
     - MongoDB: This open-source NoSQL database is currently the only database supported by Meteor.
     - Redis: The most popular key-value store, this database is lighting fast for retrieving data but does not allow for much depth in the data storage.
     - PostgreSQL: A popular open-source SQL database.
     - MySQL: Another popular open-source SQL database, MySQL is used in WordPress sites.
     - Oracle: An enterprise SQL database.
     - SQL Server: A SQL server manager created by Microsoft.
     - Neo4j: A graph-based NoSQL variant.

- **Client (or client-side): A client is one user of an application.** For accessing www.facebook.com on a client (computer, tablet, or mobile device), there are typically multiple clients interacting with the same application stored on a server.
- **Server (or server-side): The server is where the web application code is typically stored.** Requests are made to the server from clients, and the server gathers the appropriate information and responds to those requests. Some of the common servers include: web server, application server, FTP server, chat server, online game server, IRC server, and mail server.

  Examples of popular web servers include Apache and IIS.
- **Front-end:** The front-end is comprised of HTML, CSS, and JavaScript. This is how and where the web site is displayed to users.
- **Back-end:** The back-end is comprised of servers and databases. This is where functions, methods, and data manipulation happen that you don't want the client to see.
- **Protocols:** Protocols are standardized instructions for how to pass information back and forth between computers and devices.
- **HTTP:** Hypertext Transfer Protocol is how the data from a web site gets to your browser. Whenever you type a web site address (such as http://www.google.com), this protocol requests the web site from Google's server and then receives a response with the HTML, CSS, and JavaScript of the web site.
- **DDP:** Distributed data protocol is a new protocol created in connection with Meteor. The DDP protocol uses web sockets to create a consistent connection between the client and the server. This constant connection lets web sites and data on those sites update in real-time without having to refresh the browser.
- **REST:** Representational state transfer is a protocol mainly used for APIs. It has standard methods like GET, POST, and PUT that let information be exchanged between applications.
- **API:** An application programming interface is created by the developer of an application to allow other developers to use some of the application's functionality without sharing code. Developers expose "end points," which are like inputs and outputs for the application. API access can be controlled with API keys. Examples of good APIs are those created by Facebook, Twitter, and Google for their web services.
- **Data formats:** Data formats are the structure of how data is stored.

Common examples include:

- JSON: JavaScript Object Notation is quickly becoming the most popular data format.
- XML: The main data format in the earlier days of the Web, eXtensible Markup Language was predominantly used by Microsoft systems.
- CSV: Comma-Separated Values is data formatted by commas. Excel data is typically formatted this way.

- **GitHub:** Referred to as a developer's collaborative platform, **GitHub is now the largest online storage space of collaborative works that exist in the world.** One of the main misconceptions about GitHub is that it is a development tool. In reality, GitHub itself is simply a social network like Facebook or Instagram where you build a profile, upload projects to share, and connect with other users by following their accounts. Git is the version control software that runs at the heart of GitHub. GitHub is a graphical interface, which manages changes to a project without overwriting any part of that project. Thus, GitHub makes Git easier to use in two ways. First, if you download the GitHub software to your computer, it provides a visual interface to help you manage your version-controlled projects locally. Secondly, creating an account on GitHub.com brings a version controlled projects to the Web and ties in social network features for good measure.
- **Web hosting:** Hosting your own web site doesn't have to cost a monthly fee or require a lot of technical knowledge to setup. If you just need to host a small web site that only has few visitors, you can turn your Windows PC into a WAMP (Windows, Apache, MySQL, and PHP) server—a local development server. Downloading WAMP simply downloads a program that installs three different things: Apache, MySQL, and PHP. WAMPs are convenient because they allow you to download and install all of the packages needed for hosting dynamic web content. (If you're running Linux instead of Windows, you can install a LAMP.) It's also possible to host a web site on Windows using IIS so you don't have to install any third-party software.

OTHER ELEMENTS OF WEB DEVELOPMENT

It might look easy to design websites but in reality, it's not easy as it looks. While designing websites, one has to keep in mind both the appearance and functionality of websites. A good web design will only impress the audience as the impression will either make remain on the website and get to know about the business or will leave the website and look for another one.

There are basically two elements of web design i.e., **visual elements and functional elements.**

1. Visual Elements: Visual elements simply mean an aspect or representation of something that we see. There are various visual elements and some of them are given below:

- **Layout:** Layout represents the overall structure of the website that includes graphics, ads, texts, headers, menus, content, etc. The website layout is very important as it helps to guide users and tell them where they want to look. A good layout will also make it easy for one to have access to information and give a clear path for navigation. There are different layouts available nowadays and one can choose any one of them as there are no such rules when choosing a layout.

- **Shapes:** Shapes simply mark the boundaries of content. It also plays important role in branding. Each geometrical or graphical shape has its own meaning that influences our mind. Shapes simply represent a concept, create movement, affect the composition, help to create complex drawings and paintings, provide depth and texture, etc. It also helps to increase the number of visitors on the website as appearance influence peoples that in turn improve your business.
- **Colors:** Choosing appropriate colors for web design is very essential and important. It can be considered a powerful element in developing a website successfully. It is a way to connect to users or visitors and increase brand recognition. Effective usage of color attracts people and increases the number of visitors. One can choose different colors as per their choice.
- **Images and Icons:** Images and icons are the best way to improve the user experience of your website as well as attract the attention of users. It is the best and most compatible way to present information and explaining concepts. It makes the website look professional and more interesting. It also supports text content and gives a quick summary of the text.

2. Functional Elements: Functional elements simply mean aspect or representation of functionalities of websites. It simply represents what the website can do and how it works. There are various functional elements and some of them are given below:

- **Navigation:** Navigation simply means the process of monitoring and controlling the movement around websites or screens, etc. It simply directs visitors or users to various web pages and will allow searching websites for longer. It is one of the best ways to determine whether the website is working properly or not.
- **User Interaction:** The user interface is very important as it will either make and increase visitors to your website or break the user base. It simply attracts users and simplifies their activities. It is basically a point where users interact with websites such as scrolling, clicking, typing, etc.
- **Animation:** Animations play a very important role to attract more users and enhance the user experience. Animation simply demonstrates a product or images and are mostly made with CGI (Computer Generated Imagery). It allows one to do more with less, interacts with people in a better way, and communicates clearly and effectively.

CLIENT-SIDE and SERVER-SIDE SCRIPTING

Let us first understand the concepts of markup, scripting, and programming languages:

- **Markup language:** Markup languages are the building blocks of the Web and were initially a set of text and some formatting instructions when web sites were just static pages. Originating from typesetting processes used in early printing presses, these languages have long been used to annotate the text of a site, dictating both the architecture of a site and the display of text. Markup languages were designed for the processing, definition, and presentation of text. The language specifies code for formatting both the layout and style within a text file. The code used to specify the formatting is referred to as tags. HTML is an example of a widely known and used markup language. With an elaborate history, markup languages have evolved and are in no way obsolete. They are very much a part of the current generation of dynamic web and are still used to control the presentation of data.
- **Programming Language:** A programming language is a notation for writing programs, which are specifications of a computation or algorithm. They are used to transform the data or we can say are used to instruct the computer to perform logic. These are compiled, that is, they take text files, run them through a compiler, and the compiler creates binary instruction files (binaries). Examples include C, C++, Java, Objective-C, and Swift.
- **Scripting language:** Scripting languages are programming languages, but they fit into a category called interpreted languages (i.e., Python, Ruby, PHP). They use an interpreter or some running application to take programming commands and turn them into instructions to be executed. Basically, scripting languages are programming languages. The theoretical difference between the two is that scripting languages do not require the compilation step and instead are interpreted. This is a historical distinction that is almost obsolete and in the today's environment, they are used interchangeably.

  For a web app to work, two programs run at the same time:

  The code that lives on the server and responds to HTTP requests, referred to as the server-side code.

  The code that lives in the browser and responds to user input, referred to as the client-side code.

1. Client-side scripting: Web browsers execute client-side scripting. It is used when browsers have all code. Source code is used to transfer from webserver to user's computer over the internet and run directly on browsers. It is also used for validations and functionality for user events.

It allows for more interactivity. It usually performs several actions without going to the user. It cannot be basically used to connect to databases on a web server. These scripts cannot access the file system that resides in the web browser. Pages are altered on basis of the user's choice. It can also be used to create "cookies" that store data on the user's computer.

2. Server-side scripting: Web servers are used to execute server-side scripting. They are basically used to create dynamic pages. It can also access the file system residing at the webserver. A server-side environment that runs on a scripting language is a web server.

Scripts can be written in any of a number of server-side scripting languages available. It is used to retrieve and generate content for dynamic pages. When you need to store and retrieve information a database will be used to contain data. It can use huge resources of the server. It reduces client-side computation overhead. The server sends pages to the request of the user/client.

Features or Differences between client-side scripting and server-side scripting:

Client-side scripting	Server-side scripting
Source code is visible to the user.	Source code is not visible to the user because its output of server-side is an HTML page.
Its main function is to provide the requested output to the end user.	Its primary function is to manipulate and provide access to the respective database as per the request.
It usually depends on the browser and its version.	In this any server-side technology can be used and it does not depend on the client.
It runs on the user's computer.	It runs on the webserver.
There are many advantages linked with this like faster response times, a more interactive application.	The primary advantage is its ability to highly customize, response requirements, access rights based on user.
It does not provide security for data.	It provides more security for data.
It is a technique used in web development in which scripts run on the client's browser.	It is a technique that uses scripts on the webserver to produce a response that is customized for each client's request.
HTML, CSS, and JavaScript are used.	PHP, Python, Java, Ruby are used.
No need of interaction with the server.	It is all about interacting with the servers.
It reduces load on processing unit of the server.	It surge the processing load on the server.
Examples : JavaScript, VBScript, HTML (Structure), CSS (Design) AJAX, jQuery	**Examples :** PHP, ASP.NET, JavaScript (Node.js), Java and JSP, Python (Django) Ruby (Rails)

Advantages and disadvantages of client-side scripts

Advantages: Client-side scripts offer numerous advantages

1. Allow for more interactivity by immediately responding to users' actions.

2. Execute quickly because they do not require a trip to the server.

3. May improve the usability of Web sites for users whose browsers support scripts.

4. Can give developers more control over the look and behaviour of their Web widgets.

5. Can be substituted with alternatives (for example, HTML) if users' browsers do not support scripts

6. Are reusable and obtainable from many free resources.

Disadvantages: Client-side scripts also create additional worries which include:

1. Not all browsers support scripts, therefore, users might experience errors if no alternatives have been provided.

2. Different browsers and browser versions support scripts differently, thus more quality assurance testing is required.

3. More development time and effort might be required (if the scripts are not already available through other resources).

4. Developers have more control over the look and behaviour of their Web widgets; however, usability problems can arise if a Web widget looks like a standard control but behaves differently or vice-versa.

Advantages and disadvantages of server-side scripts

Advantages:

1. User can create one template for the entire website

2. The site can use a content management system which makes editing simpler.

3. Generally quicker to load than client-side scripting

4. User is able to include external files to save coding.

5. Scripts are hidden from view so it is more secure. Users only see the HTML output.

6. User does not need to download plugins like Java or Flash.

Disadvantages:

1. Many scripts and content management systems tools require databases in order to store dynamic data.

2. It requires the scripting software to be installed on the server.

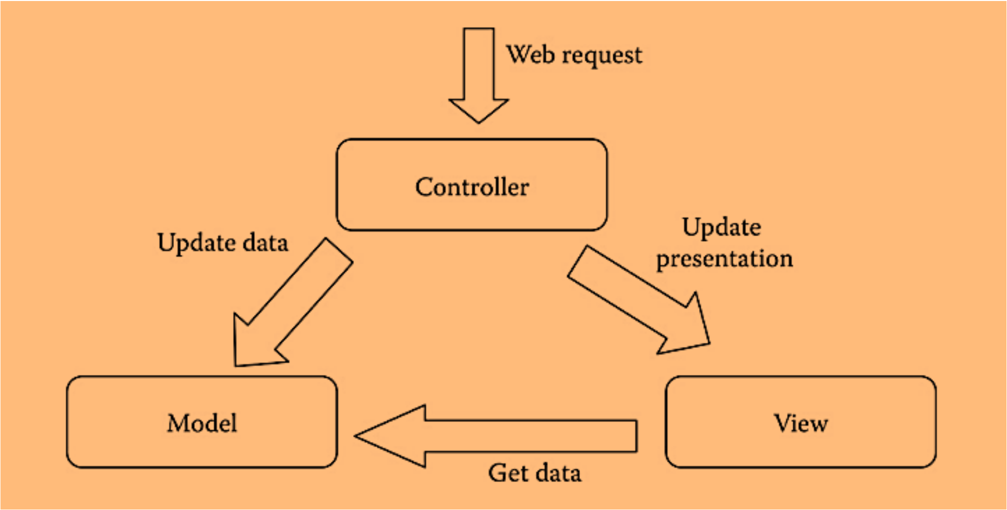
3. The nature of dynamic scripts creates new security concerns, in some cases making it easier for hackers to gain access to servers exploiting code flaws.

MODEL – VIEW – CONTROLLER ARCHITECTURE FOR WEB APPLICATION DEVELOPMENT

Definition: The Model-View-Controller (MVC) is an architectural pattern that separates an application into three main logical components:

- The Model
- The View
- The Controller

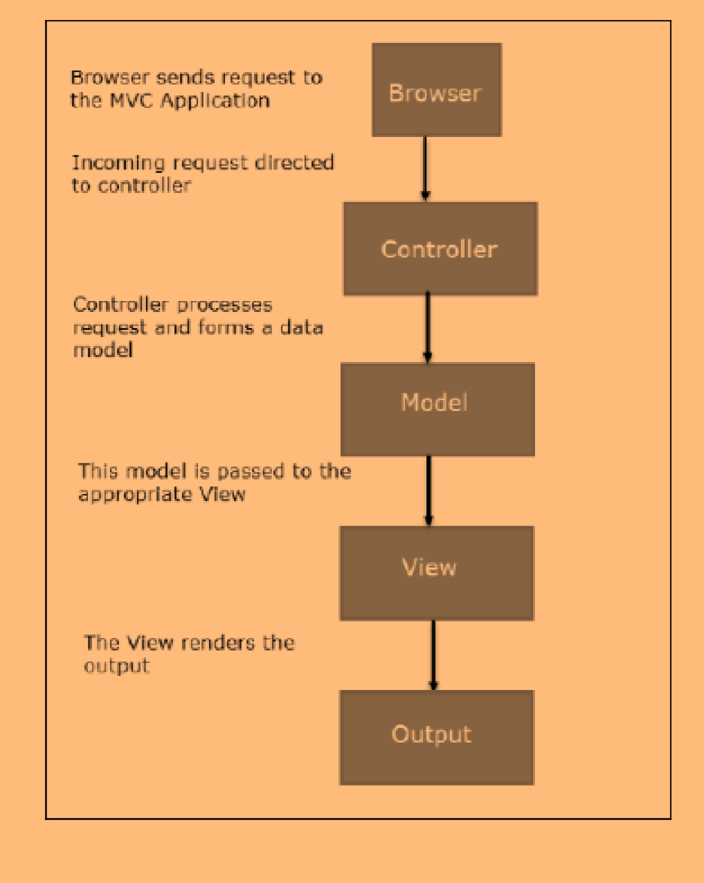
Each of these components is built to handle specific development aspects of an application.



- **Model:** The Model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data. For example, a Customer object will retrieve the customer information from the database, manipulate it and update it data back to the database or use it to render data.
- **View:** The View component is used for all the UI logic of the application. For example, the Customer view will include all the UI components such as text boxes, dropdowns, etc. that the final user interacts with.
- **Controller:** Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output. For example, the Customer controller will handle all the interactions and inputs from the Customer View and update the database using the Customer Model. The same controller will be used to view the Customer data.

Now let us take a look at how the execution of an MVC application takes place when there is a certain request from the client. The following diagram illustrates the flow.

MVC Flow Diagram:


Browser sends request to the MVC Application
Browser
Incoming request directed to controller
Controller
Controller processes request and forms a data model
Model
This model is passed to the appropriate View
View
The View renders the output
Output

Flow Steps:

Step 1: The client browser sends request to the MVC Application.

Step 2: Global.ascx receives this request and performs routing based on the URL of the incoming request using the Route Table, Route Data, URL Routing Module and MVC Route Handler objects.

Step 3: This routing operation calls the appropriate controller and executes it using the Controller Factory object and MVC Handler object's Execute method.

Step 4: The Controller processes the data using Model and invokes the appropriate method using Controller Action Invoker object

Step 5: The processed Model is then passed to the View, which in turn renders the final output.

MVC has been adopted and established as a great architecture for application development no matter what language you are using for development. Essentially, it allows for the programmer to isolate these very separate pieces of code into their own domain, which makes code maintenance and debugging much simpler than if all these items were chunked into one massive piece. MVC supports rapid and parallel development with proven cost and time benefits. The advantages and disadvantages of using MVC framework are as follows:

Advantages:

- **Simultaneous development:** Multiple developers can work simultaneously on model, controller, and views.
- **High cohesion:** MVC enables logical grouping of related actions on a controller. The views for a specific model are also grouped together.
- **Low coupling:** The very nature of the MVC framework is such that there is low coupling among models, views, or controllers.
- **Ease of modification:** Because of the separation of responsibilities, future development or modification is easier.
- **Multiple views for a model:** Models can have multiple views.

Disadvantages:

- **Code navigability:** The framework navigation can be complex because it introduces new layers of abstraction and requires users to adapt to the decomposition criteria of MVC.
- **Multi-artifact consistency:** Decomposing a feature into three artifacts causes scattering. Thus, requiring developers to maintain the consistency of multiple representations at once.
- **Pronounced learning curve:** Knowledge on multiple technologies becomes the norm. Developers using MVC need to be skilled in multiple technologies.

CLIENT-SIDE TECHNOLOGIES

Definition: In web development, 'client side' refers to everything in a web application that is displayed or takes place on the client. This includes what the user sees, such as text, images, and the rest of the UI, along with any actions that an application performs within the user's browser.

HYPERTEXT MARKUP LANGUAGE: Hypertext Markup Language (HTML) is a simple text formatting language for annotating a document in a way that is syntactically distinguishable from the text, thus creating hypertext documents. A hypertext document is viewed in a web Browser, such as Internet Explorer or Netscape Navigator.

The **browser** is a program that understands the document written in HTML and displays it by interpreting the document contents. A unique feature of this versatile language is that it allows the creation of links, also known as hyperlinks. These links are references to data that the user follows by clicking.

A **hyperlink** may point to a whole document or to a specific element within a document. Thus, “hypertext” means machine readable text, “markup” means to structure it in a specific format, and “hyperlinks” are links from a hypertext document to another location, activated by clicking on highlighted text, an icon, or a graphic image.

HTML Versions: Since the time HTML was invented there are lots of HTML versions in market, the brief introduction about the HTML version is given below:

HTML 1.0: The first version of HTML was 1.0, which was the barebones version of HTML language, and it was released in1991.

HTML 2.0: This was the next version which was released in 1995, and it was standard language version for website design. HTML 2.0 was able to support extra features such as form-based file upload, form elements such as text box, option button, etc.

HTML 3.2: HTML 3.2 version was published by W3C in early 1997. This version was capable of creating tables and providing support for extra options for form elements. It can also support a web page with complex mathematical equations. It became an official standard for any browser till January 1997. Today it is practically supported by most of the browsers.

HTML 4.01: HTML 4.01 version was released on December 1999, and it is a very stable version of HTML language. This version is the current official standard, and it provides added support for stylesheets (CSS) and scripting ability for various multimedia elements.

HTML5 : HTML5 is the newest version of HyperText Markup language. The first draft of this version was announced in January 2008. There are two major organizations one is W3C (World Wide Web Consortium), and another one is WHATWG(Web Hypertext Application Technology Working Group) which are involved in the development of HTML 5 version, and still, it is under development.

Features of HTML:

1) It is a very easy and simple language. It can be easily understood and modified.

2) It is very easy to make an effective presentation with HTML because it has a lot of formatting tags.

3) It is a markup language, so it provides a flexible way to design web pages along with the text.

4) It facilitates programmers to add a link on the web pages (by html anchor tag), so it enhances the interest of browsing of the user.

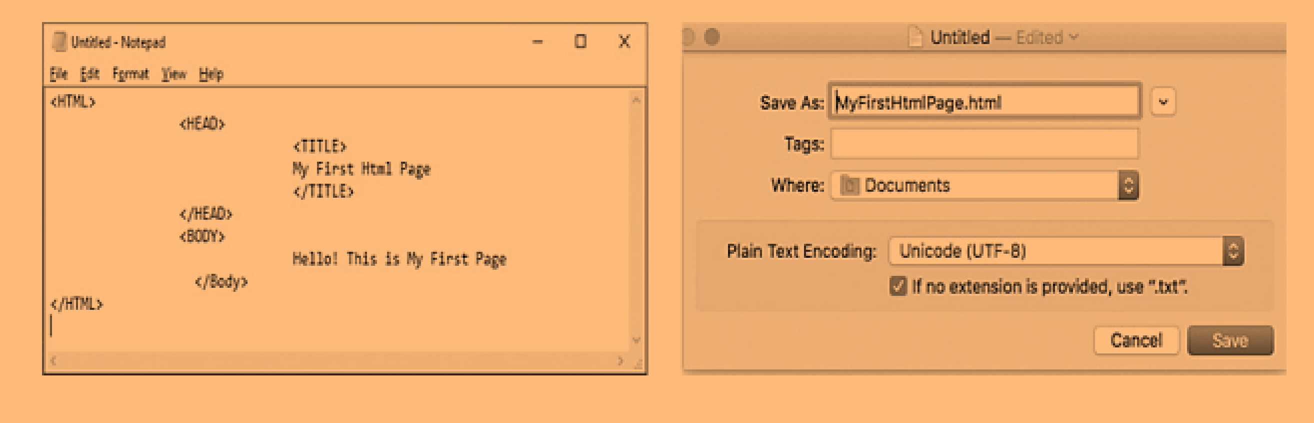
5) It is platform-independent because it can be displayed on any platform like Windows, Linux, and Macintosh, etc.

6) It facilitates the programmer to add Graphics, Videos, and Sound to the web pages which makes it more attractive and interactive.

7) HTML is a case-insensitive language, which means we can use tags either in lower-case or upper-case.

Now, to create an HTML document, open the Notepad application by accessing the Start Menu on the Taskbar: Start->Programs->Accessories->Notepad

Add the HTML code in Notepad and save the file, naming it MyFirstHtmlPage.html as shown below:



```
<HTML>
            <HEAD>
                       <TITLE>
                       My First Html Page
                       </TITLE>
            </HEAD>
            <BODY>
                       Hello! This is My First Page
              </Body>
</HTML>
```

Save As: MyFirstHtmlPage.html

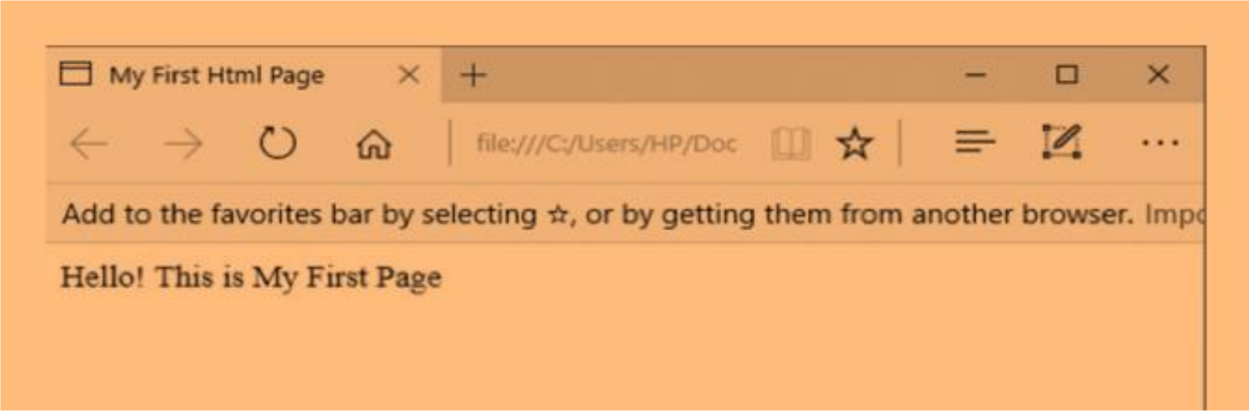
Tags:

Where: Documents

Plain Text Encoding: Unicode (UTF-8)

☑ If no extension is provided, use ".txt".

To view the HTML page in your browser, open the My Documents folder on your computer and double-click on the file MyFirstHtmlPage.html which will automatically open in the default browser program and display the result as shown below:



Add to the favorites bar by selecting ☆, or by getting them from another browser. Impo

Hello! This is My First Page

HTML TAGS: HTML tags are the building blocks of HTML documents. They are used to define the structure and content of a web page. HTML tags are made up of two parts: **an opening tag and a closing tag**. The opening tag contains the name of the tag, and the closing tag contains a forward slash followed by the name of the tag (/).

HTML tags are a powerful tool that can be used to create complex and interactive web pages. By understanding how HTML tags work, you can create websites that are both visually appealing and functional.

Tags are control text used to denote various elements in an HTML document. A tag in HTML comprises a left angular bracket (<), followed by a tag name, and closed by a right angular bracket (>).

For example:

<html>, <body>, and <title> etc. The tags described here are examples of start tags.

Every start tag must have an end tag, which is similar to the start tag but with a forward slash sign before the tag name: </html>, </body>, and </title>.

The following code defines a paragraph tag:

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

The opening tag is <p>, and the closing tag is </p>. The text between the opening and closing tags is the content of the paragraph.

There are many different types of HTML tags, each with its own purpose. Some common HTML tags include:

<html>: The root element of an HTML document.

<head>: The head element contains information about the document, such as the title and the author.

<title>: The title element defines the title of the document.

<body>: The body element contains the content of the document.

<h1> to <h6>: The heading elements define headings of different levels.

<p>: The paragraph element defines a paragraph.

<br>: The line break element inserts a line break.

<hr>: The horizontal rule element inserts a horizontal rule.

HTML tags can be nested, which means that one tag can be inside another tag. For example, the following code defines a paragraph that contains a heading:

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

The heading tag is nested inside the paragraph tag. This means that the heading will be displayed as part of the paragraph.

HTML tags can also have attributes, which are additional pieces of information about the tag. Attributes are specified after the tag name, and they are enclosed in quotation marks. For example, the following code defines a paragraph tag with an id attribute:

<p id="myParagraph">This is a paragraph.</p>

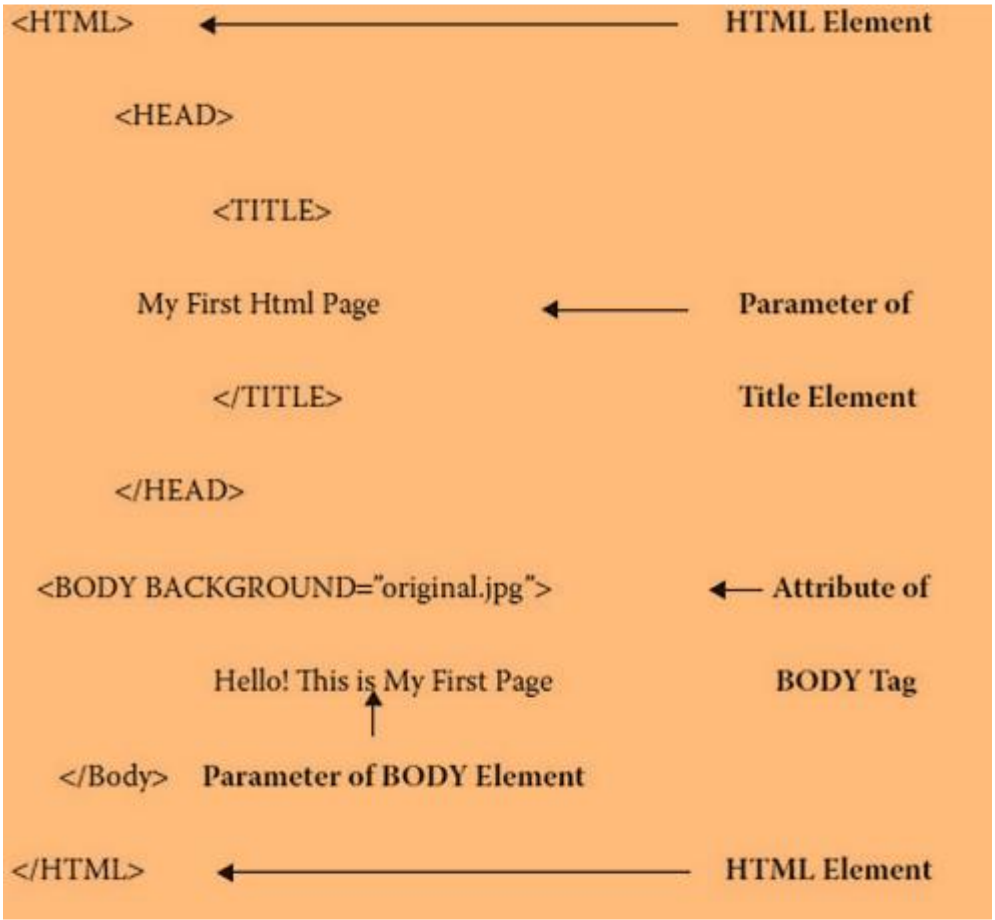
The id attribute is used to identify the paragraph. This can be useful for styling the paragraph or for linking to it.

Basic structure of an HTML page:

Empty elements: Empty elements have only a start tag and no end tag. Hence, an empty element has no parameters, but can take attributes, which are given within the angular brackets of the start tag.

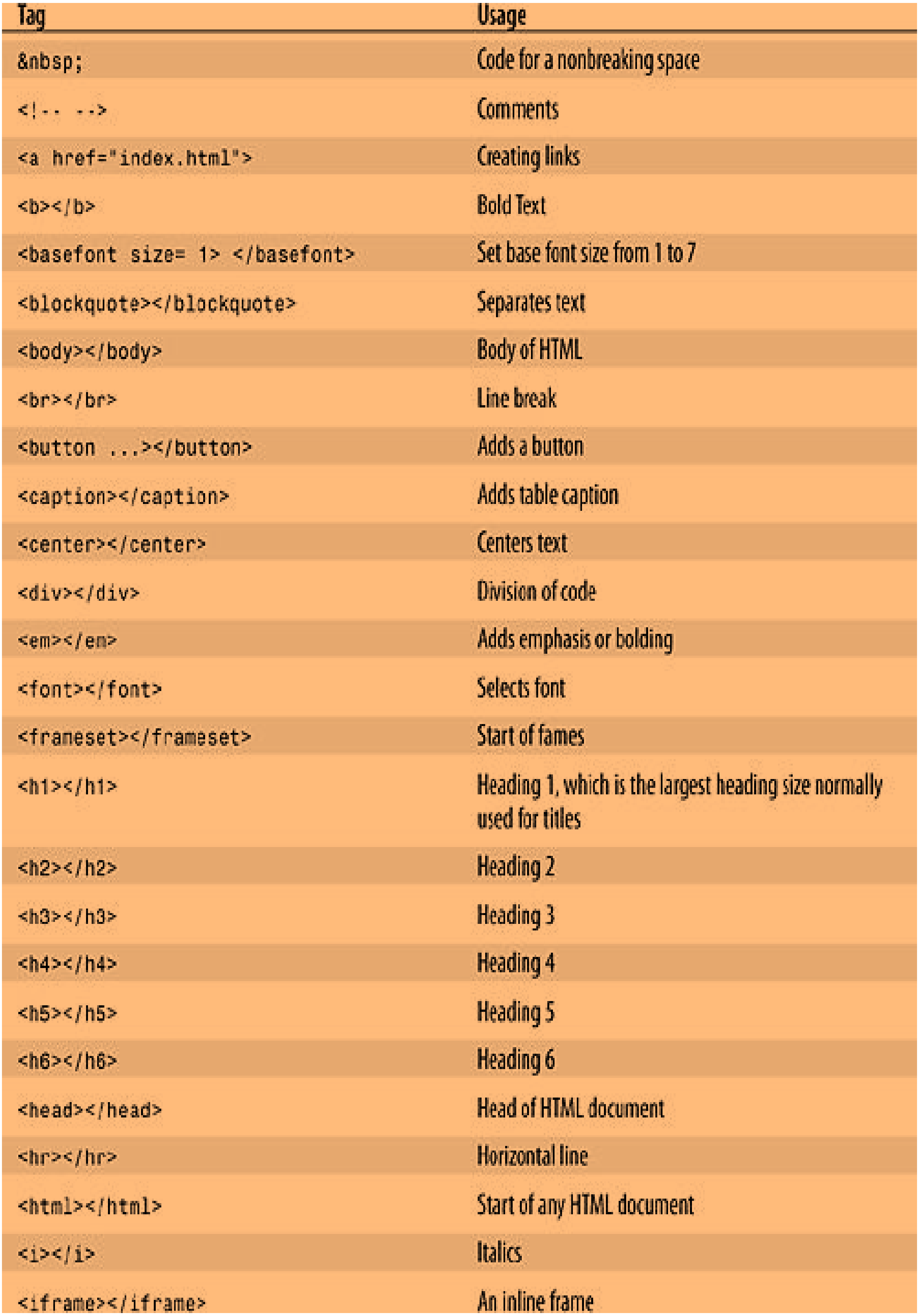
HTML attributes: Attributes are used to define the characteristics of an HTML element and they are placed inside the element's opening tag. All attributes are made up of two parts: a name (the property that you want to set) and a value (the value of the named property). The value is always put in quotes. Attribute names and values are not case sensitive, but lowercase is recommended. For example, in the figure below, the <body> tag has an attribute named background whose value has been set to an image named "original.jpg."

The basic structure of an HTML document is given by the following elements, known as document structure elements.



```
<HTML>
      <HEAD>
            <TITLE>
         My First Html Page
            </TITLE>
      </HEAD>
  <BODY BACKGROUND="original.jpg">
            Hello! This is My First Page
    </Body>
</HTML>
```

The most frequently used HTML tags are described in the Table below.



Tag	Usage
`&nbsp;`	Code for a nonbreaking space
`<!-- -->`	Comments
`<a href="index.html">`	Creating links
`<b></b>`	Bold Text
`<basefont size= 1> </basefont>`	Set base font size from 1 to 7
`<blockquote></blockquote>`	Separates text
`<body></body>`	Body of HTML
` </br>`	Line break
`<button ...></button>`	Adds a button
`<caption></caption>`	Adds table caption
`<center></center>`	Centers text
`<div></div>`	Division of code
`<em></em>`	Adds emphasis or bolding
`<font></font>`	Selects font
`<frameset></frameset>`	Start of fames
`<h1></h1>`	Heading 1, which is the largest heading size normally used for titles
`<h2></h2>`	Heading 2
`<h3></h3>`	Heading 3
`<h4></h4>`	Heading 4
`<h5></h5>`	Heading 5
`<h6></h6>`	Heading 6
`<head></head>`	Head of HTML document
`<hr></hr>`	Horizontal line
`<html></html>`	Start of any HTML document
`<i></i>`	Italics
`<iframe></iframe>`	An inline frame

Tag	Usage
<img src ="image location></img>	Image source
<input name></input>	Input form
<li></li>	List item
<meta name= ></meta>	Meta information
<ol></ol>	Numbered list
<option value="list"></option>	Option form
<p></p>	New paragraph
<param></param>	Sets a parameter on an element
<q></q>	Quotation
<select name></select>	Selection Form
<strike></strike>	Strikethrough text
<style></style>	Style sheet definition
	Subscript text
	Superscript text
<table></table>	Table definition
<td></td>	Data cell definition
<th></th>	Header cell for a table
<title></title>	Document title
<tr></tr>	Table row
<ul></ul>	Bulleted list

Attributes of the <body> element include the following:

- **Background:** Used to specify the URL (uniform resource locator) of an image that will be used as a background for the document and the image tiles on the entire background.

For example: <body background=“original.jpg”>---------</body>.

The image to be copied should have an extension of either .JPG or .GIF. To display a .BMP file, it must first be converted into .JPG or .GIF format.

- **Bgcolor:** This sets the color of the background. There are three major colors RED, BLUE, and GREEN. If combination is required then the value for bgcolor can be given as #RRGGBB.
- **Text:** Sets the color of the normal text in the document. Color values can be given in the same way as that of bgcolor attribute.

HTML Programs:

1. **Write a HTML program to display six headings or heading tags.**

```
<html>
<head>
<title> Introduction to HTML </title>
</head>
<body>
<body bgcolor="blue">
<h1> SINDHI COLLEGE </h1>
<h2> SINDHI COLLEGE </h2>
<h3> SINDHI COLLEGE </h3>
<h4> SINDHI COLLEGE </h4>
<h5> SINDHI COLLEGE </h5>
<h6> SINDHI COLLEGE </h6>
</body>
</html>
```

2. **Write a HTML program to display bold, italic and underline tags.**

```
<html>

<head>
```

```
<title>

Program to display bold, italic and underline tags

</title>

</head>

<body>

<body bgcolor="red">

<br>

<b>  BCA  </b>    <br>

<i>  B.Sc  </i>       <br>

<u> Sindhi College  </u>          <br>

</body>

</html>
```

3. Write a HTML program to insert a horizontal line.

```
<html>
<body>
<body bgcolor="pink">
<title>  Horizontal Line </title>
<p>   The  hr defines a horizontal line  </p>
<hr>
<p>   This is a paragraph   </p>
<hr>
</body>
</html>
```

4. Write a HTML program to center the content.

```
<html>
<head>
<title> Centring Content  </title>
</head>
```

```
<body>
<body bgcolor="yellow">
<p> This text is not in the center </p>
<center>
<p>This text is in the center </p>
</center>
</body>
</html>
```

5. **Write a HTML program to implement ordered list.**

```
<html>
<head>
        <title>ordered list</title>
</head>
<body>
        <h1>Example of ordered list in default</h1>
        <ol >
                <li>Sindhi  College</li>
                <li>Courses</li>
                <li>BCA</li>
                <li>B.Sc</li>
                <li>B.Com</li>
                <li> BBA </li>
        </ol>
</body>
</html>
```

6. **Write a HTML program to implement inline style attribute.**

```
<html >
<head>
        <title>inline style attribute</title>
</head>
<body>
        <ol>
        <li value="11">English</li>
        <li>K/H/S</li>
        <li>IT</li>
        <li>SE</li>
        <li>ADA</li>
    </ol>
</body>
</html>
```

7. **Write a HTML program to implement types of ordered list.**

```
<html>
<head>
    <title>ordered list</title>
</head>
<body>
    <h1>Example of ordered list whose type = "A"</h1>
    <ol type="A">
        <li>Sachin</li>
        <li>Manoj</li>
    </ol>
    <h1>Example of reverse ordered list</h1>
    <ol reversed>
        <li>Parth</li>
        <li>sujay</li>
    </ol>
    <h1>Example of ordered list start from 10</h1>
    <ol start = "10">
        <li>Pushpa</li>
        <li>Purvi</li>
    </ol>
    </body>
</html>
```

Note:- Different ways to number the ordered lists using the type attribute:

type=”1″- The list items will be numbered with numbers (default)

type=”A”- The list items will be numbered with uppercase letters

type=”a”- The list items will be numbered with lowercase letters

type=”I”- List items will be numbered with uppercase Roman numbers

type=”i”- The list items will be numbered with lowercase Roman numbers.

8. **Write a HTML program to implement unordered list.**

```
<html>
<head>
        <title>unordered list</title>
</head>
<body>
        <h2>Example of unordered list in circle</h2>
        <ul style="list-style-type:circle;">
                <li>sachin</li>
                <li>manoj</li>
        </ul>
        <h2>Example of unordered list in disk</h2>
        <ul style="list-style-type:disk;">
                <li>Priya</li>
                <li>Mohit</li>
        </ul>
        <h2>Example of unordered list in square</h2>
        <ul style="list-style-type:square;">
                <li>Pinky</li>
                <li>Punam</li>
        </ul>
        <h2>Example of unordered list in none</h2>
        <ul style="list-style-type:none;">
                <li>Mukti</li>
                <li>Dhama</li>
        </ul>   </body>  </html>
```