TWKSAA SKILLS CENTER

Foundation Day
30-09-2023

त्वक्सा कौशल केंद्र

TWKSAA HTML BOOK

Er. Rajesh Prasad

- आप एक सागर हो बहते नदी का जल नहीं आप एक बदलाव हो भटकाव की कोई राह नहीं
- उस रास्ते पर चलो जिस रास्ते पर भीड़ कम हो (हर काम हो कुछ अलग)
- देश की मिट्टी से करो आप इतना प्यार जहाँ जाओ वहाँ मिले खूब इज्जत और सम्मान
- छह दिन कीजिए अपना काम एक दिन कीजिए त्वक्सा को दान
- त्वक्सा एक चिंगारी हैं हर जगह जलना हम सब की जिमेवारी हैं
- **Motive: - New (RID PMS & TLR)**

"त्वक्सा HTML के इस पुस्तक में आप HTML के संबंध में सभी बुनियादी अवधारणाएँ सीखेंगे। मुझे आशा है कि इस पुस्तक को पढ़ने के बाद आपके ज्ञान में वृद्धि होगी और आपको कंप्यूटर विज्ञान के बारे में और अधिक जानने में रुचि होगी"

"In this TWKSAA HTML book you will learn all basic concept regarding HTML. I hope after reading this book your knowledge will be improve and you will get more interest to know more thing about computer Science".

"Skill कौशल एक व्यक्ति के पास उनके ज्ञान, अनुभव, तत्वशास्त्रीय योग्यता, और प्रैक्टिकल अभियांत्रिकी के साथ संचित नौकरी, व्यापार, या अन्य चुनौतीपूर्ण परिस्थितियों में सक्रिय रूप से काम करने की क्षमता को कहते हैं। यह व्यक्ति के द्वारा सीखी जाने वाली कौशलों की प्रतिभा, क्षमता और निपुणता को संक्षेप में व्यक्त करता है"।

TWKSAA RID MISSION

(Research)	(Innovation)	(Discovery)
अनुसंधान करने के महत्वपूर्ण कारण:	**नवीनीकरण करने के महत्वपूर्ण कारण:**	**खोज करने के महत्वपूर्ण कारण:**
1. नई ज्ञान की प्राप्ति	1. प्रगति के लिए	1. नए ज्ञान की प्राप्ति
2. समस्याओं का समाधान	2. परिवर्तन के लिए	2. ज्ञान के विकास में योगदान
3. तकनीकी और व्यापार में उन्नति	3. उत्पादन में सुधार	3. अविष्कारों की खोज
4. विकास को बढ़ावा देना	4. प्रतिस्पर्धा में अग्रणी होने के लिए	4. समस्याओं का समाधान
5. सामाजिक प्रगति	5. समाज को लाभ	5. समाज के उन्नति का माध्यम
6. देश विज्ञान और प्रौद्योगिकी का विकास	6. देश विज्ञान और प्रौद्योगिकी के विकास।	6. देश विज्ञान और तकनीक के विकास

"TWKSAA Skills Center is Learning Earning and Development Based Skill Center."

त्वक्सा कौशल केंद्र सीखने कमाई और विकास आधारित कौशल केंद्र है।

TWKSAA SKILLS CENTER

TWKSAA SKILLS CENTER

Definition: - www is a global collection of documents and other resources linked by hyperlink and URLs.it is known as web, it is an information system technology enabling.

History: - computer scientist "Tim Berners Lee" at CERN {(European Organization for nuclear Research) it is a Intergovernmental org. established in 1954)} invented in1989.1st proposal was written & working system implemented by end of 1990 including www Browser & http server.

Function: - 1).HTML 2). Linking 3). www prefix 4). Scheme specifiers 5). Web Page 6). Website 7). Browser 8). Search Engine 9). Server 10). Cookie 11). Deep web 12). Caching 13). Security 14). Privacy 15). Standards

HTML: - Hypertext Markup Language it used for Creating Web page & Web Application.

Linking: - it is interconnecting the web page via Hyperlinks.

www prefix: -it is like .com, .org, .net etc. **Scheme specifiers: -** http:// or https://

Browser: - it is a software responsible for open the website

Web Page: - A webpage is an HTML document on the WWW. **Website: -** it is a collection of web page.

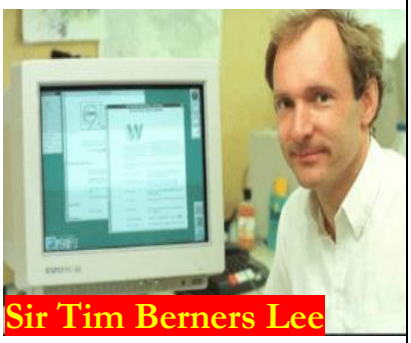
Search Engine: - it is a software program/system Software Desigen to carry out the web search.

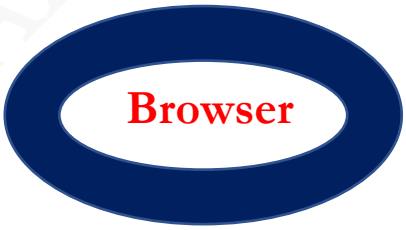
Server: - it is a software or hardware device that accept & respond to request made over a network.

Cookie: - it is a small piece of data sent from the website and stored on the user's computer by the web browser while user is browsing. It is stateful

Deep web: - it is an invisible web or hidden web are parts of www whose contents are not indexed by standard web search engine. Computer scientist "Michael K. Bergman" is credited with deep web in 2001

Caching: - A web cache is a server computer located on the public internet. It is stores recently accessed web page to improve response time for user's


Sir Tim Berners Lee



Definition: - Browser is an application software or a software Program.

Use: - Browser is used for accessing websites fetch content from the www or from local storage and display on the user's Device

History: - www was the 1st Browser created in 1990 by sir Tim Berner Lee Mosaic-1993 Netscape-1994 Internet Explorer-1995 Opera-1995 Mozilla Firefox-2004 Safari-2003 Chrome-20008 Edge-

Features: - Automatically log user's Browsing history, set Book Marks, Customize Browser with Extensions, User password, Sync Service, Web Accessibility, open Multiple Pages, Back & forward Bottoms, Refresh, Reload
Stop, Home bottom, Address Bar to IP URLs and Security etc.

TWKSAA SKILLS CENTER

Definition: - A search engine is a software system designed to carry out web searches. Or it is set of Program.

use: - it is used to search www in a systematic way for particular information specified in a textual web search query.

History: - 10 sept 1990 is D.O.B of 1st Search Engine "Archie" by Alan Emtage (note: - note index Concept)
1st Popular Search Engine was Yahoo! (Founded by Yang and David Filo in 1994) Google Search Engine (Founded by Lary Page & Sergey Brin in 1998) used Page Rank Algorithm, Indexing & Hyperlinks

Example: - Google, Bing, Yahoo, Baidu, DuckDuckGo, Yandex, Ask.com, AOL Search, Ecosia, Qwant etc.

Types: - 1. Conventional 2. Text-Based 3. Voice-Based 4. Multimedia Search 5. Q/A 6. Clustering 7. Research System

Conventional (Library CatLog): - Search by keyword title, Author etc.

Text-based & Voice-based: - Google, Bing & Yahoo! Search by keyword

Multimedia Search: - (QBIC, Web seek, safe) Search by visual Appearance (Shapes, colours)

Q/A: - Stack Exchange, NSIR search in (Restricted) Natural Language.

Clustering: - Vivisimq, clustery **Research System: -** Lemur, Nutch

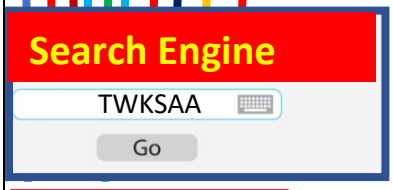
How Search Engine work

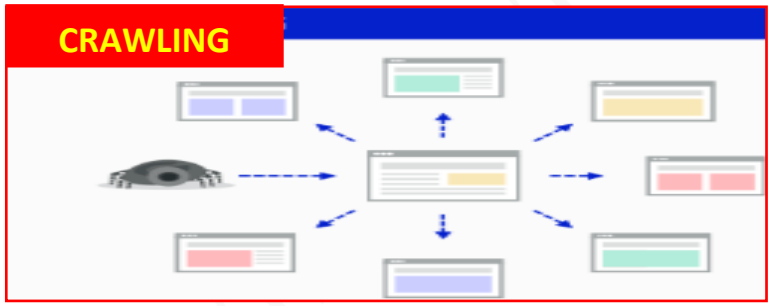
Crawling: - also known as Spider or Spider Bot it is internet bot that systematically browses the www and that is typically operated by search engine for the purpose of web indexing.

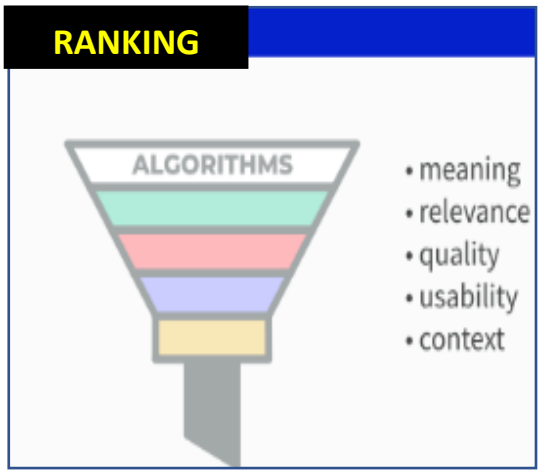
Indexing: - Collecting, Parsing and Storing of Data

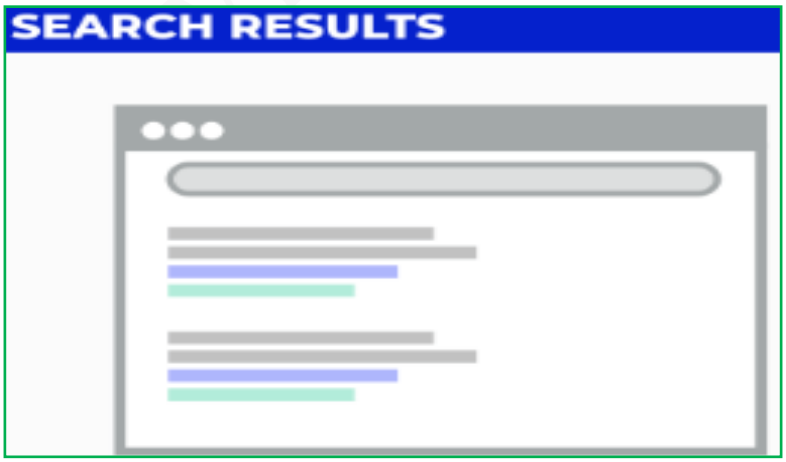
Ranking: - position a website or webpage holds within a specific search engine results page.

Search Results: - it is a query that a user enters into a "web search engine "to satisfy the information needs









TWKSAA SKILLS CENTER

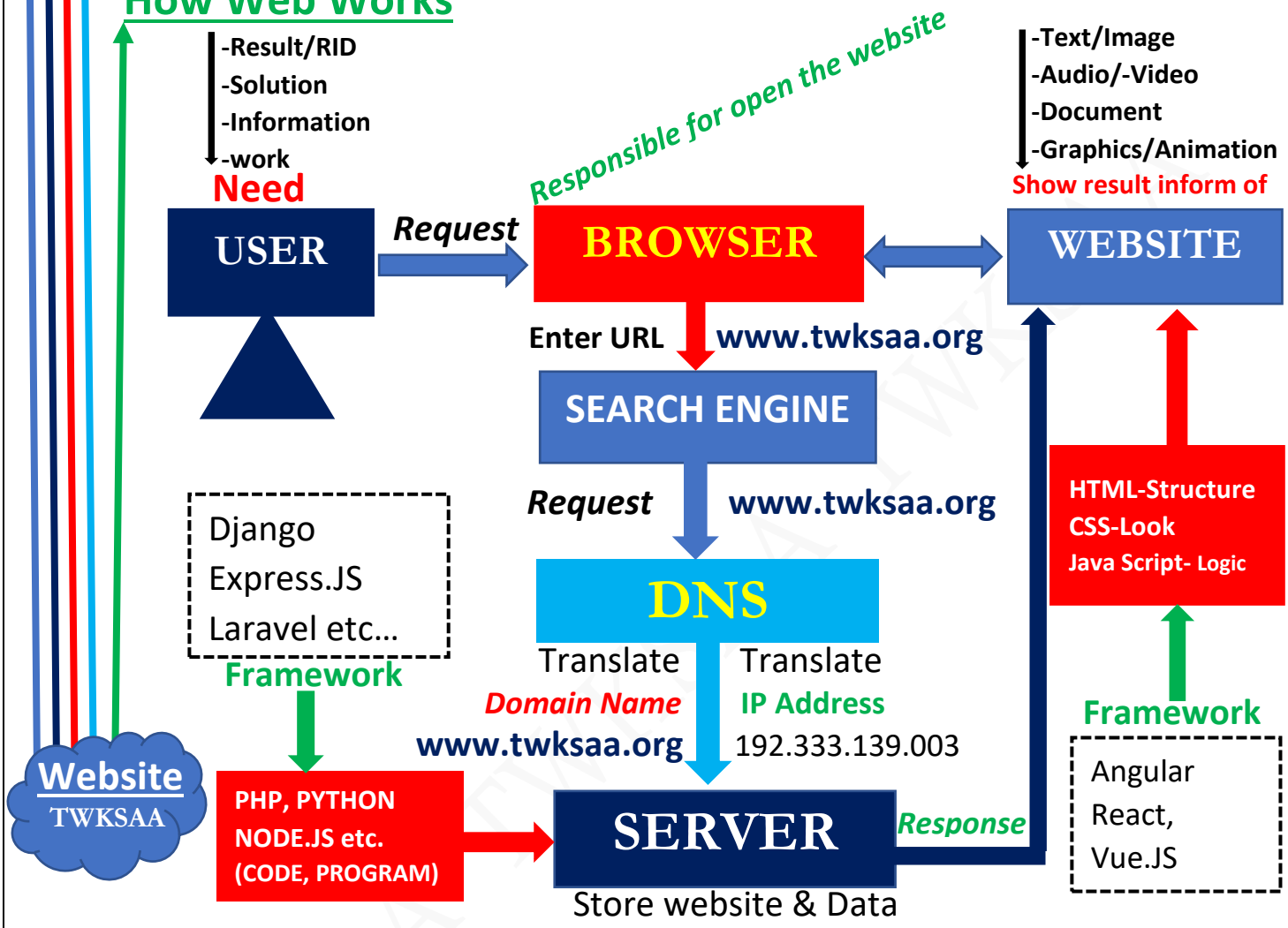
Definition: - website is collection of web page and related content that is identified by a common "Domain name"

Types: - 1). Static Website 2). Dynamic Website

Static Website: - consists of a series of HTML files, each one representing a physical page of a website.

Dynamic Website: -change or customizes itself frequently and automatically.

How Web Works



Web: The web is a global system of interconnected computer networks that use the Internet protocol suite to access and share information. It allows users to access and share information over the Internet. Or Web is virtual directory on web server. Or Web [Portion of Internet]

Site:

- Site [Location] A site refers to a location or a collection of web pages hosted on a web server and accessible through a specific domain or URL.
- A site refers to a specific location on the internet identified by a unique domain name and accessible via a web browser.

Page:

- A page refers to a single, individual document or resource on the web.
- It is a single document or resource that is part of a website and can be accessed through a specific URL?

Web Page:

- A web page is a single hypertext document available on World Wide Web (WWW).
- Hyper Text document that contains information beyond what is displaying.
- The term "Hyper" is derived from a Greek term, which means "beyond"

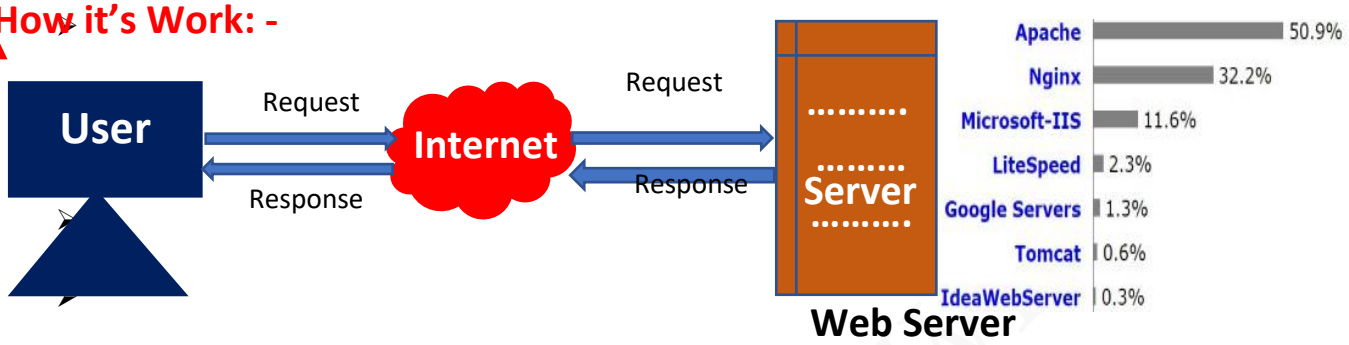
WEB SERVER

Definition: - web server is computer software and hardware that accepts requests via HTTPS (Network protocol created to distribute web content). A web server is a dedicated computer responsible for running websites

use: - it is used to process and manage HTTP/HTTPS requests and responses from the client system. A web server Store and protect website data.

Example: - Apache (Http server project), Microsoft IIS, Nginx, Apache Tomcat etc.

How it's Work: -



Work: - 1. Receive Client Request/Response (Read & Verify, URL-Normalization, URL Mapping, URL Path Redirections) **2. Executes or refuse HTTP Request Method** (URL Authorization, URL Redirection, Directory Index File Regular Files) **3. Response/Replies** (HTTP Response, Logs)

Features: - 1. Static Content serving 2. HTTP/HTTPS 3. Logging 4. Dynamic Content Serving 5. Virtual Hosting 6. Authorization 7. Content Cache 8. Large file Support 9. Bandwidth throttling 10. Rewrite Engine 11. Custom Error Page 12. Security

"WEB SERVER"

- **Web page:** Web page is a single document on the web.
- **Website:** web site is a collection of related web pages.
- **Web server:** web server is hosts and delivers web pages to users.

web server software companies and release dates:

- **Apache HTTP Server (Apache):** Released in 1995
- **Nginx:** First released in 2004
- **Microsoft Internet Information Services (IIS):** Initial release in 1995
- **LiteSpeed Web Server:** First release in 2003
- **Google Web Server (GWS):** Private server developed by Google
- **Caddy:** Initial release in 2015
- **IBM HTTP Server (IHS):** First released in the late 1990s
- **LiteWebServer:** Initial release in 2016
- **Cherokee:** First released in 2005
- **Hiawatha:** Initial release in 2002

WEB PAGE

- **Definition:**
  - A web page is a single document or resource that contains content in the form of text, images, multimedia, or interactive elements, designed to be displayed within a web browser.
  - web page is a document which is commonly written in HTML and translated by a web browser.
- **HTML(Hyper Text Markup Langauge):**
  - Web pages are primarily created using HTML which provides the structure and content of page.
- **URL:**
  - Each web page has a unique address called a URL (Uniform Resource Locator), which allows users to access it using a web browser.
- **Interactivity:**
  - Web pages can include interactive elements using technologies like JavaScript, allowing users to interact with the content and perform various actions.
- **Navigation:**
  - Hyperlinks are used to connect web pages together, allowing users to navigate from one page to another within a website or across different websites.
- **Metadata:**
  - Web pages can include metadata, such as title, description, and keywords.
- **Rendering:**
  - When a user requests a web page, the web server delivers the HTML code to the user's browser, which then interprets and renders the content for display.
- **Design and Styling:**
  - CSS (Cascading Style Sheets) is used to style the web page.

❖TYPES OF WEB PAGE:

- There are two types of web pages.
  1. Static page
  2. Dynamic Page

1. Static Page:

- Static refers to continuous memory.
- The memory allocated for first request will continue for others.
- Static page contains information that will be same across any number of requests.
- Statics page respond with the same content across any number of requests.

- **Static pages will have extension**
  - **.htm**
  - **.html**

2. Dynamic Page:

- Dynamic refers to discreet memory
- The memory is newly allocated for every request.
- Dynamic page contains information that is customized according to the client request.

Static pages will have extension

- **.aspx, .php**
- **.asp, .jsp**

Note: Every website that you designed starts with a default page called “index.html”.

HTML

HYPER TEXT MARKUP LANGUAGE

❖ What is HTML?

- HTML stands for Hyper Text Markup Language
- HTML is the standard markup language for creating Web pages
- HTML describes the structure of a Web page
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.
- Technically, HTML is a Markup language rather than a programming language.
- HTML is Not Case Sensitive.

- **Hyper Text:** Hyper Text means "Text within Text." A text has a link within it, is a hypertext.
- **Markup language:** A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic.
- **Developer:**
  - HTML was developed by **Sir Tim Berners-Lee,** a British computer scientist, in 1990 while working at CERN (the European Organization for Nuclear Research). He is often credited as the inventor of the **World Wide Web.** Father of HTML is known as Sir Tim Berners-Lee.
  - It is an open standard maintained by the World Wide Web Consortium (W3C).
  - The latest version of HTML is HTML5.

❖ HTML text editors:

- Notepad(windows)
- Notepad++ (Windows)
- Visual Studio Code (Cross-platform)
- Sublime Text (Cross-platform)
- Atom (Cross-platform)
- Brackets (Cross-platform)
- Vim (Cross-platform)
- Emacs (Cross-platform)
- TextMate (Mac)
- Komodo Edit (Cross-platform)
- Bluefish (Cross-platform)
- Coda (Mac)
- UltraEdit (Windows, Mac, Linux)
- Dreamweaver (Windows, Mac)
- CoffeeCup HTML Editor (Windows, Mac)
- Pinegrow (Cross-platform)

Features of HTML:

1. **Markup Language:**
- HTML is a markup language that uses tags and elements to define the structure and content of a web page.

2. **Document Structure:**
- HTML provides a hierarchical structure for organizing content, using elements like `<html>`, `<head>`, `<body>`, `<header>`, `<footer>`, etc.

3. **Hyperlinks:**
- HTML supports hyperlinks (`<a>` element) that allow users to navigate between web pages or external resources.

4. **Text Formatting:**
- HTML provides tags for text formatting, such as headings (`<h1>` to `<h6>`), paragraphs (`<p>`), bold (`<b>` or `<strong>`), italics (`<i>` or `<em>`), etc.

5. **Lists:**
- HTML supports ordered lists (`<ol>`), unordered lists (`<ul>`), and definition lists (`<dl>`).

Images and Multimedia: HTML allows embedding images (`<img>`), audio (`<audio>`), video (`<video>`), and other multimedia elements.

6. **Forms:**
- HTML provides form elements (`<form>`, `<input>`, `<select>`, `<textarea>`, etc.) for creating interactive input fields and collecting user data.

7. **Semantic Elements:**
- HTML5 introduces semantic elements (`<header>`, `<nav>`, `<main>`, `<section>`, `<article>`, `<aside>`, `<footer>`, etc.) that define the meaning and structure of content.

8. **Responsive Design:**
- HTML supports responsive design principles, allowing websites to adapt to different screen sizes and devices.

9. **Metadata:**
- HTML includes meta tags (`<meta>`) for providing metadata, such as character encoding, author, description, and viewport settings.

10. **Accessibility:**
- HTML supports accessibility features, like providing alternative text for images (`alt` attribute) and creating semantic structures for screen readers.

11. **Cross-Browser Compatibility:**
- Allowing web pages to be viewed consistently across different web browsers.

12. **Embedded Scripting:**
- HTML can include scripts (e.g., JavaScript) to add interactivity and dynamic behaviour to web pages.

13. **Versioning:**
- HTML versions evolve over time, with HTML5.

Structure of HTML page:

- Every HTML Page comprises of 2 sections at high level.
    1. Document Declaration
    2. Document Scope. Scope 'Means' region of code

1. Document Declaration.

- It specifies the version of HTML used for web page.
- If document declaration is not defined then it is HTML 4.
- To indicate that the page is designed in HTML 5 we need document declaration in the first line of page.

<!DOCTYPE html>

Where: **!** Comment-not a tag

- Document declaration contains meta data, information about HTML.
- **Like**
    a) Its version
    b) Its culture
    c) License etc.

2. Document Scope:

- It specifies the scope of HTML document in page.
- It is defined by using **<html>** tag

Syntax:

```
<!DOCTYPE html>
<html>
</html>
```

- It is mandatory to define the culture type used in document so that Browser engine can understand the format of content in page.
- The language culture in page is defined by using "lang" attribute.

Example:

```
<html lang= "en-in">
 <html>
```

Where **lang** -is an attribute of <html> tag

Section in Document Scope:

- It comprises of content that is intended to load into Browser memory. So that it can accessed and used by Brower or by page whenever required.
- It is defined by using

```
<head>
</head>
```

❖ **Head section of web page contains following elements:**

1) <meta>
2) <title>
3) <link>
4) <style>
5) <script>

1). <meta>:

- Meta refers to "**meta-Data**"
- Meta data means information about your page given to Web Spiders and Web Crawlers used in SEO (Search Engine Optimization)
- Meta is one of the options used for SEO.
- Meta is also used for responsive design.

Syntax:

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
```

<meta charset="UTF-8">

- <meta charset="UTF-8"> tag is used to specify the character encoding of the document.
- In this case, it indicates that the document is encoded using UTF-8

UTF-8:

- UTF-8 was developed by Ken Thompson and Rob Pike, computer scientists at Bell Labs, in 1992.
- UTF-8 (Unicode Transformation Format - 8-bit) is a character encoding standard that is widely used in computing, particularly for representing and processing text.
- Character set utf-8, utf-16, utf-32 [English- 8bit, Chinese, jap, French, Korean -16 bit]
- Arabic, Regional-32-bit, **Hindi utf-8, Unicode Example: अ -> E0 A4 85**
- The popularity of UTF-8 stems from its efficiency and compatibility with ASCII.

Character encoding:

- Character encoding is a method used in computing to represent characters, symbols, and textual information in a digital form. It is essential for encoding and decoding text in various programming languages and communication protocols.

ASCII:

- ASCII (American Standard Code for Information Interchange) is a character encoding standard that was first developed in the early 1960s.ASCII was developed and initially published by the American National Standards Institute (ANSI). ANSI is a private, non-profit organization

Unicode:

- Unicode is a universal character encoding standard that provides a unique numeric code for virtually every character used in human writing systems
- Purpose of Unicode is to eliminate limitations of traditional character encoding systems, such as ASCII, which only supported a limited set of characters used in English language. need arose for a unified way to represent and exchange text in various languages and writing systems

2). <title>:

- It is the title to display for page in the title bar of Browser window.
- It is also used for book marking the page.

Syntax:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>TWKSAA</title>
</head> </html>
```

3). <link>:

- It is used to link any external document to web page.

Syntax:

```
<head>
        <link rel= “stylesheet”   href= “favicon.ico>
</head>
```

Where:

14. **Rel:** it specifies the relation type of external file.
15. **Href:** it specifies the path and name of icon file.

4). <script>: it is used to embed client or server-side script.

5). <style>: it is used to embed styles in web page.

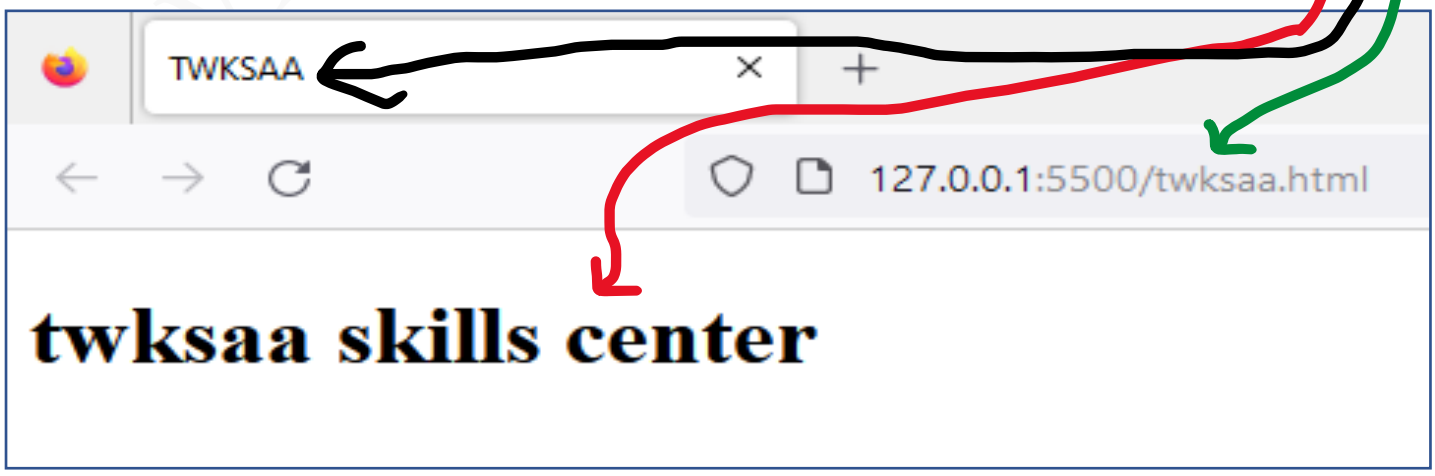
Example:

Twksaa.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>TWKSAA</title>
</head>
<body>
  <h1>TWKSAA Skills Center</h1>
</body>
</html>
```

Description of HTML Example

- **<!DOCTYPE>:** It defines the document type or it instruct the browser about the version of HTML.
- **<html >:** This tag informs the browser that it is an HTML document. Text between html tag describes the web document. It is a container for all other elements of HTML except <!DOCTYPE>
- **<head>:** It should be the first element inside the <html> element, which contains the metadata (information about the document). It must be closed before the body tag opens.
- **<title>:** As its name suggested, it is used to add title of that HTML page which appears at the top of the browser window. It must be placed inside the head tag and should close immediately.
- **<body>:** Text between body tag describes the body content of the page that is visible to the end user. This tag contains the main content of the HTML document.
- **<h1> TWKSAA Skills Center <h1>:** tag describes the first level heading of the webpage.



❖How to View HTML Source from any website?

- **View HTML Source Code:**

➢ Right-click in an HTML page and select "View Page Source" (in Chrome) or "View Source" (in Edge), or similar in other browsers. This will open a window containing the HTML source code of the page.

- **Inspect an HTML Element:**

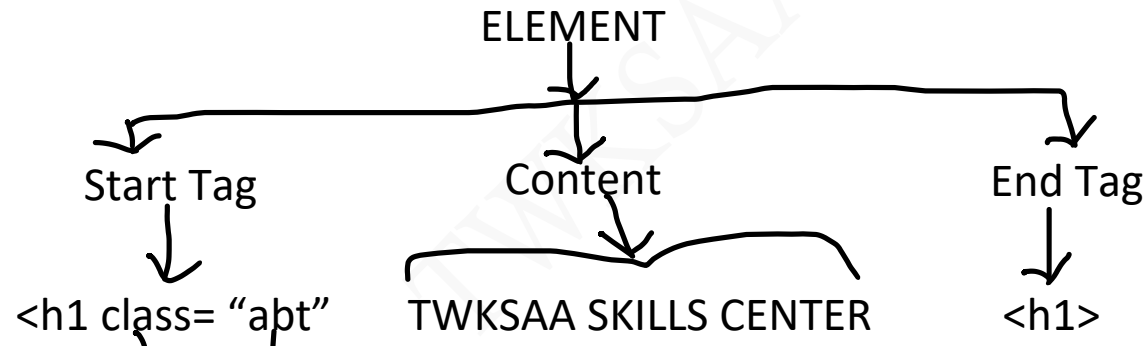
➢ Right-click on an element (or a blank area), and choose "Inspect" or "Inspect Element" to see what elements are made up of (you will see both the HTML and the CSS). You can also edit the HTML or CSS on-the-fly in the Elements or Styles panel that opens.

- **Building blocks of HTML:**

16. **Tags:** it is represented by angle brackets (<>) and come in pairs: an opening tag and a closing tag. Example: <h1>……..</h1>, …….. , ……..
17. **Attribute:** An attribute in HTML provides extra information about the element, and it is applied within the start tag. An HTML attribute contains two fields: name & value.
18. **Elements:** An HTML element is an individual component of an HTML file. In an HTML file, everything written within tags are termed as HTML elements.

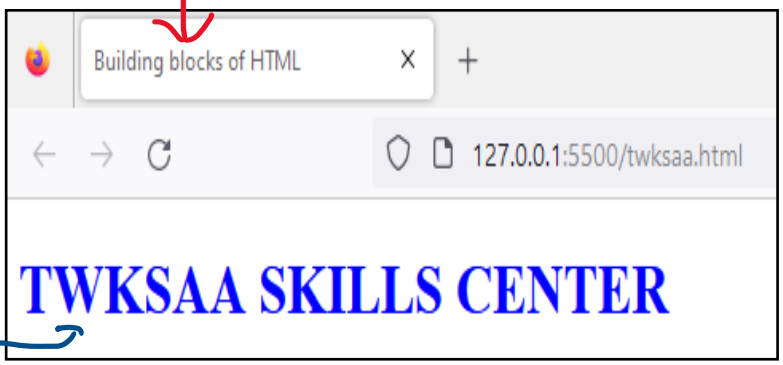
Syntax:

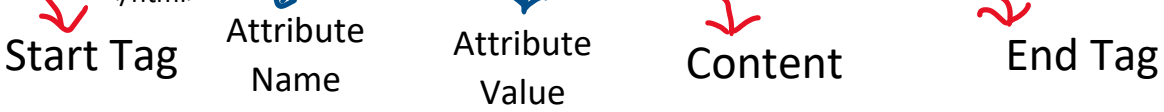
- <tag name attribute_name= " attr_value"> content </ tag name>



Example:

```
<!DOCTYPE html>
<html>
 <head>
        <title>Building blocks of HTML</title>
 </head>
 <body>
    <h1 style="color: blue">TWKSAA SKILLS CENTER</h1>
 </body>
</html>
```





HTML Editors:

- To work with HTML, you can use a variety of text editors, integrated development environments (IDEs), and code editors.
  - **Visual Studio Code (VS Code):** Visual Studio Code is a free, open-source code editor developed by Microsoft. It's highly extensible and supports a wide range of programming languages, including HTML, CSS, and JavaScript. VS Code offers features like syntax highlighting, code completion, and a robust extension marketplace for web development.
  - **Sublime Text:** Sublime Text is a lightweight and fast code editor available for Windows, macOS, and Linux. It's known for its speed and responsiveness. Sublime Text supports HTML and many other programming languages and offers a variety of plugins to extend its functionality.
  - **Atom:** Atom is another free and open-source code editor created by GitHub. It's highly customizable and has a vibrant community of developers who create packages and themes to enhance the editor's features and appearance. Atom is suitable for web development, including HTML, CSS, and JavaScript.
  - **Notepad++:** Notepad++ is a free and open-source text editor for Windows. While it's not as feature-rich as some other editors, it's lightweight and easy to use. It provides syntax highlighting for HTML and other programming languages.
  - **Brackets:** Brackets is an open-source code editor specifically designed for web development. It offers live preview functionality, which allows you to see changes in your HTML, CSS, and JavaScript code in real-time as you edit. Brackets is suitable for HTML development.
  - **Adobe Dreamweaver:** Dreamweaver is a commercial web design and development tool by Adobe. It offers a visual design interface in addition to code editing features. It's often used by web designers and developers who prefer a comprehensive IDE for building websites and web applications.
  - **Emacs:** Emacs is a highly extensible and customizable text editor that has been popular among developers for decades. With the right extensions, it can be turned into a powerful HTML and web development environment.
  - **Vim:** Vim is a highly configurable, text-based code editor with a steep learning curve but powerful features. Many developers prefer Vim for its efficiency and productivity once they become proficient with it. It has various plugins and configurations for web development.
  - **Online Code Editors:** There are several online HTML editors, such as CodePen, JSFiddle, and Repl.it, that allow you to write, test, and share HTML, CSS, and JavaScript code directly in your web browser. These platforms are often used for quick prototyping and collaboration.

❖ How to write HTML code in Visual Studio Code:

Step 1: Install Visual Studio Code

- If you haven't already installed Visual Studio Code, you can download it from the official website (https://code.visualstudio.com/) and follow the installation instructions for your operating system.

Step 2: Open Visual Studio Code

- After installation, open VS Code. You should see a clean, minimalist interface with a sidebar on the left and a code editing area in the center.

Step 3: Create a New HTML File

- To create a new HTML file, go to the "File" menu and select "New File" or use the keyboard shortcut `Ctrl+N` (Windows/Linux) or `Cmd+N` (macOS). A new untitled file will appear in the editing area.

Step 4: Save the HTML File

- It's a good practice to save your HTML file in an organized manner. To save the file, go to the "File" menu and choose "Save" or use the keyboard shortcut `Ctrl+S` (Windows/Linux) or `Cmd+S` (macOS). Specify a name for your HTML file and choose a location on your computer to save it. Make sure to give it the ".html" file extension, such as "index.html."

Step 5: Write HTML Code

- Now, you can start writing your HTML code in the editing area. Here's a simple example of an HTML structure:

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>My First Web Page</title>
</head>
<body>
  <h1>Hello, World!</h1>
  <p>This is a simple HTML page.</p>
</body>
</html>
```

- You can type this code directly into the VS Code editor.

Step 6: Use Auto-Completion and Formatting**

- VS Code provides features like auto-completion and formatting to help you write HTML code more efficiently. For example, when you start typing an HTML tag (e.g., `<h1>`), VS Code will suggest the tag and automatically close it for you when you press `Enter`.

Step 7: Save Your Changes

- As you work on your HTML code, make sure to save your changes regularly by pressing `Ctrl+S` (Windows/Linux) or `Cmd+S` (macOS).

Step 8: Preview Your HTML Page

- You can preview your HTML page directly in VS Code by right-clicking the file in the file explorer and selecting "Open with Live Server." This extension will open your HTML file in a web browser, allowing you to see how it appears. You can install the "Live Server" extension from the VS Code marketplace if you haven't already.

Step 9: Debug and Test

- If you encounter issues or want to test your HTML code, you can use the built-in debugging and testing tools provided by VS Code or install additional extensions for web development.

Step 10: Continue Editing and Developing

- Continue writing, editing, and enhancing your HTML code as needed. VS Code offers a wide range of extensions and features for web development, including support for CSS, JavaScript, and other web technologies.

❖ How to write HTML code in notepad process step by step:

- Writing HTML code in Notepad is a simple process that doesn't require any special software other than the built-in text editor. Here's a step-by-step guide:

Step 1: Open Notepad

- On Windows, you can open Notepad by pressing `Win + R`, typing "notepad" in the "Run" dialog, and pressing Enter. Alternatively, you can search for "Notepad" in the Windows Start menu and open it from there.

Step 2: Start a New Document

- In Notepad, go to "File" in the top-left corner and select "New" to start a new, empty document.

Step 3: Write Your HTML Code

- You can now write your HTML code in the Notepad document. Here's a simple example to get you started:

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>My First Web Page</title>
</head>
<body>
  <h1>Hello, World!</h1>
  <p>This is a simple HTML page.</p>
</body>
</html>
```

- You can type this code directly into the Notepad document.

Step 4: Save the HTML File

- After writing your HTML code, you should save it. Go to "File" and select "Save" or press `Ctrl + S`.
- Choose the location where you want to save your HTML file on your computer.
- In the "Save as type" dropdown menu, select "All Files (*.*)" to ensure that you can specify the ".html" file extension.
- Give your file a name with the ".html" extension, such as "index.html." Make sure the filename doesn't have any spaces or special characters.
- Click the "Save" button to save your HTML file.

Step 5: View Your HTML Page

- You can view your HTML page in a web browser. Locate the HTML file you just created, right-click on it, and choose "Open with" from the context menu. Then, select your preferred web browser (e.g., Chrome, Firefox, Edge).
- Your web browser will open the HTML file, and you'll see your web page rendered as it should appear.

Step 6: Continue Editing and Developing

- If you want to make changes to your HTML code, open the HTML file in Notepad again, make your edits, and save the file. Then, refresh your web browser to see the updated version of your web page.

❖ How to write HTML code in sublime text editor process step by step:

- Writing HTML code in Sublime Text, a popular code editor, is a straightforward process. Here's a step-by-step guide:

Step 1: Install Sublime Text (if not already installed)

- If you haven't already installed Sublime Text, you can download it from the official website (https://www.sublimetext.com/) and follow the installation instructions for your operating system.

Step 2: Open Sublime Text

- After installation, open Sublime Text by locating it in your applications or programs list and clicking to launch it.

Step 3: Create a New HTML File

- To create a new HTML file, go to the "File" menu and select "New File" or use the keyboard shortcut `Ctrl+N` (Windows/Linux) or `Cmd+N` (macOS). A new, untitled file will appear in the editor.

Step 4: Save the HTML File

- Save your HTML file by going to the "File" menu and choosing "Save" or using the keyboard shortcut `Ctrl+S` (Windows/Linux) or `Cmd+S` (macOS). A save dialog will appear.
- Choose the location where you want to save your HTML file on your computer.
- Give your file a name with the ".html" extension, such as "index.html."
- Click the "Save" button to save your HTML file.

Step 5: Write Your HTML Code

- Now, you can start writing your HTML code in the editor. Here's a simple example to get you started:

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>My First Web Page</title>
</head>
<body>
  <h1>Hello, World!</h1>
  <p>This is a simple HTML page.</p>
</body>
</html>
```

- You can type this code directly into the Sublime Text editor.

Step 6: Use Auto-Completion and Formatting (optional)

- Sublime Text offers features like auto-completion and code formatting. For example, when you start typing an HTML tag (e.g., `<h1>`), Sublime Text may suggest the tag and automatically close it for you when you press `Enter`.

Step 7: Save Your Changes

- As you work on your HTML code, make sure to save your changes regularly by pressing `Ctrl+S` (Windows/Linux) or `Cmd+S` (macOS).

Step 8: Preview Your HTML Page

- To preview your HTML page, locate the HTML file you just created, right-click on it, and choose "Open with" from the context menu.

Step 9: Continue Editing and Developing

HTML Tag:

- HTML tags are like keywords which defines that how web browser will format and display the content.
- When a web browser reads an HTML document, browser reads it from top to bottom and left to right.
- HTML tags are used to create HTML documents and render their properties.
- Each HTML tags have different properties.
- All HTML tags must be enclosed within < > these brackets.
- If you have used an open tag <tag>, then you must use a close tag </tag> (except some tags)

Syntax

- <tag> content </tag>

Unclosed HTML Tags:

- Some HTML tags are not closed, for **example** br and hr.
- **
** Tag: br stands for break line, it breaks the line of the code.
- **<hr>** Tag: hr stands for Horizontal Rule. This tag is used to put a line across the webpage.

HTML Meta Tags:

- DOCTYPE, title, link, meta and style

HTML Text Tags:

- <p>, <h1>, <h2>, <h3>, <h4>, <h5>, <h6>, , , <abbr>, <acronym>, <address>, <bdo>, <cite>, <q>, <code>, <ins>, <del>, <dfn>, <kbd>, <pre>, <samp>, <var> and

HTML Link Tags:

- <a> and <base>

HTML Image and Object Tags:

- <img>, <area>, <map>, <param> and <object>

HTML List Tags:

- , <ol>, <li>, <dl>, <dt> and <dd>

HTML Table Tags:

- table, tr, td, th, tbody, thead, tfoot, col, colgroup and caption

HTML Form Tags:

- form, input, textarea, select, option, optgroup, button, label, fieldset and legend

HTML Scripting Tags

- script and noscript

HTML Attributes:

- All HTML elements can have attributes
- Attributes provide additional information about elements
- Attributes are always specified in the start tag
- Attributes usually come in name/value pairs like: name="value"

Example:

- All HTML elements can have attributes
- The **href** attribute of <a> specifies the URL of the page the link goes to
  - **twksaa**
- The **src** attribute of <img> specifies the path to the image to be displayed
  - **<img src="t3.jpg">**
- The **width** and **height** attributes of <img> provide size information for images

- **<img src="t3.jpg" width="500" height="600">**

- The **alt** attribute of <img> provides an alternate text for an image
  - **<img src="t3.jpg" alt="this is logo">**
- The **style** attribute is used to add styles to an element, such as color, font, size, and more
  - ** This is a red paragraph. **
- The **lang** attribute of the <html> tag declares the language of the Web page
  - **<html lang="eng">……. </html**
- The **title** attribute defines some extra information about an element
  - ** This is an NGO. **

HTML Elements:

- The HTML element is everything from the start tag to the end tag:

Syntax:

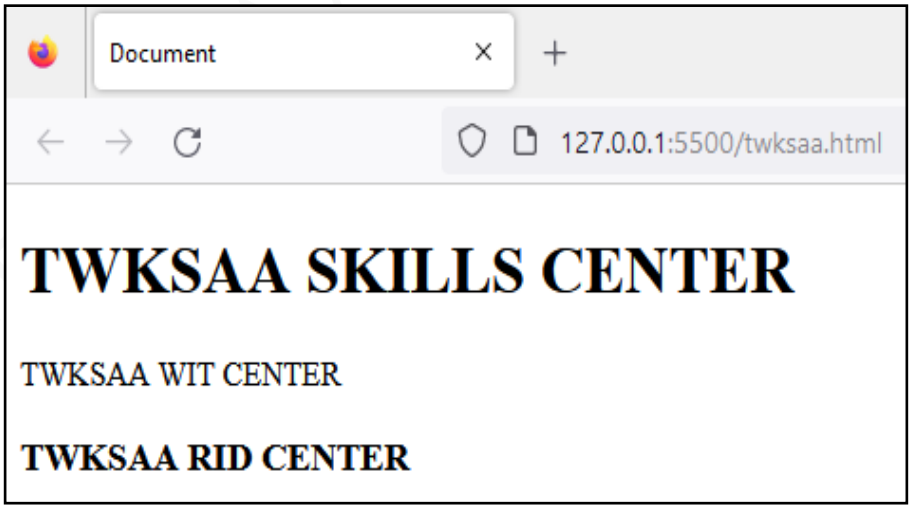
- <tagname>Content goes here...</tagname>

Examples:

```
<h1>TWKSAA SKILLS CENTER</h1>
<p>TWKSAA WIT CENTER</p>
<h3>TWKSAA RID CENTER</h3>
```

twksaa.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Document</title>
</head>
<body>
<h1>TWKSAA SKILLS CENTER</h1>
<p>TWKSAA WIT CENTER</p>
<h3>TWKSAA RID CENTER</h3>
</body>
</html>
```



Empty HTML Elements:

- HTML elements with no content are called empty elements.
- The <br> tag defines a line break, and is an empty element without a closing tag:
- Empty element also called Void elements are <hr> (represents a horizontal line)

Nested HTML Elements: HTML can be nested, which means an element can contain another element.

Block-level and Inline HTML elements:

❖ Block-level element:

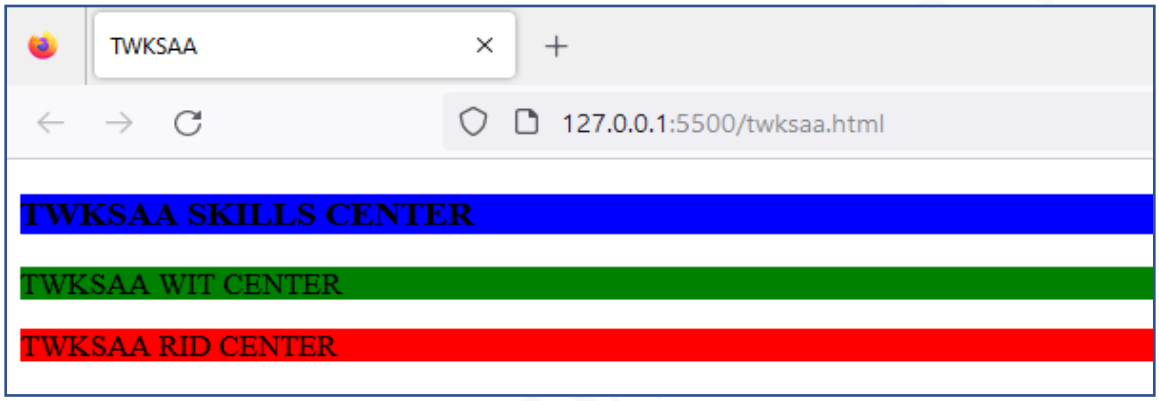
- These are the elements, which structure main part of web page, by dividing a page into coherent blocks.
- A block-level element always starts with new line and takes the full width of web page, from left to right.
- These elements can contain block-level as well as inline elements.

❖ **Example:**

- <address>, <article>, <aside>, <blockquote>, <canvas>, <dd>, , <dl>, <dt>, <fieldset>, <figcaption>, <figure>, <footer>, <form>, <h1>-<h6>, <header>, <hr>, <li>, <main>, <nav>, <noscript>, <ol>, <output>, <p>, <pre>, <section>, <table>, <tfoot>, <ul> and <video>.

❖ **twksaa.html:**

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body>
   <h3 style="background-color: blue">TWKSAA SKILLS CENTER</h3>
   <div style="background-color: green">TWKSAA WIT CENTER</div>
   <p style="background-color: red">TWKSAA RID CENTER</p>
</body></html>
```



❖ **Inline elements:**

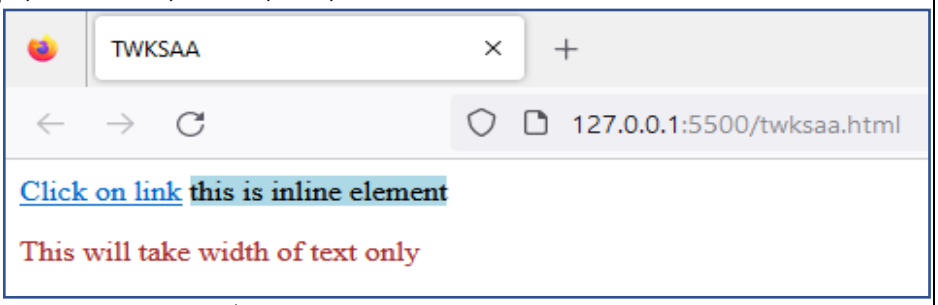
- Inline elements are those elements, which differentiate the part of a given text and provide it a particular function.
- These elements do not start with new line and take width as per requirement.
- The Inline elements are mostly used with other elements.

❖ **Example:**

- <a>, <abbr>, <acronym>, <b>, <bdo>, <big>, <br>, <button>, <cite>, <code>, <dfn>, , <i>, <img>, <input>, <kbd>, <label>, <map>, <object>, <q>, <samp>, <script>, <select>, <small>, , , <sub>, <sup>, <textarea>, <time>, <tt>, <var>.

❖ **twksaa.html:**

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body>
   <a href="https://www.twksaa.org">Click on link</a>
   <span style="background-color: lightblue">this is inline element</span>
   <p style="color: brown;">This will take width of text only</p>
</body></html>
```



HTML Attributes:

- HTML attributes are used to provide additional information or properties to HTML elements.
- it is specified in the opening tag of an HTML element
- Attributes help control the behavior, appearance, and functionality of HTML elements.
- Attributes usually come in name/value pairs like: name="value"

Example:

1. Common Attributes:

- **id:** Specifies a unique identifier for an HTML element.
- **class:** Assigns one or more class names to an element for styling or JavaScript targeting.
- **style:** Defines inline CSS styles for an element.
- **title:** Provides additional information about an element, typically displayed as a tooltip.

2. Form Attributes:

- **name:** Identifies the name of an input field when submitting form data.
- **type:** Specifies the type of input element (e.g., text, password, checkbox).
- **value:** Sets the initial or default value for form elements.
- **placeholder:** Displays a short hint or example text in an input field.
- **required:** Requires that the user fill in the input field before submitting the form.
- **disabled:** Disables the input element, making it uneditable and unclickable.
- **readonly:** Makes the input element read-only (users can't edit it but can see its value).
- **form:** Associates an input element with a specific form by referencing the form's `id`.
- **maxlength:** Specifies the maximum number of characters allowed in a text field.
- **min` and `max:** Define the minimum and maximum values for numeric input fields.
- **pattern:** Sets a regular expression pattern for input validation.
- **autocomplete:** Controls whether browser autofill suggestions are enabled.
- **autofocus:** Automatically focuses the input element when the page loads.

3. Link Attributes (`<a>`):

- **href**: Specifies the URL to which the link should navigate.
- **target:** Defines where to open the linked document

4. Image Attributes (`<img>`):

- **src:** Specifies the image file's source URL.
- **alt:** Provides alternative text for the image for accessibility and when the image can't be displayed.
- **width and height:** Set the dimensions of the image (in pixels).

5. List Attributes (`<ul>`, `<ol>`, `<li>`):

- type` (for `<ol>`): Specifies the type of numbering or bullet style (e.g., "1," "A," "a").
- start` (for `<ol>`): Sets the starting value for an ordered list.

6. Table Attributes (`<table>`, `<th>`, `<td>`):

- **border` (for `<table>`):** Defines the width of the table border.
- **colspan` and `rowspan` (for `<th>` and `<td>`):** Specifies the number of columns or rows a table cell spans.
- **scope` (for `<th>`):** Indicates the scope of a header cell for accessibility.

7. Script Attributes (`<script>`):

- **src:** Specifies the source file (URL) of an external JavaScript file.
- **type:** Defines the scripting language used (e.g., "text/javascript").
- **defer` and `async`:** Control how the script is executed and loaded.

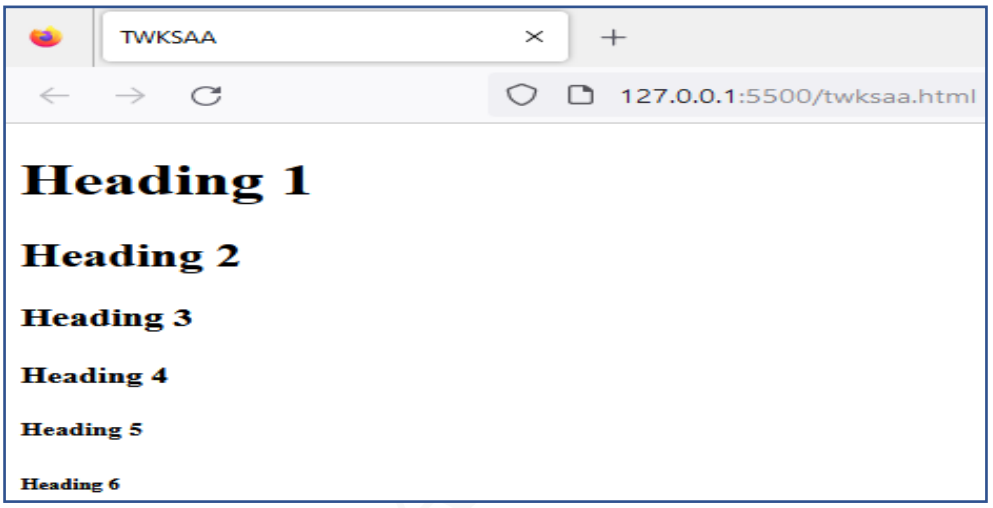
HTML Headings:

- HTML headings are titles or subtitles that you want to display on a webpage.
- HTML headings are defined with the **<h1> to <h6> tags.**
- **<h1>** defines the most important heading. **<h6>** defines the least important heading.

Example:

twksaa.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body>
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
</body></html>
```



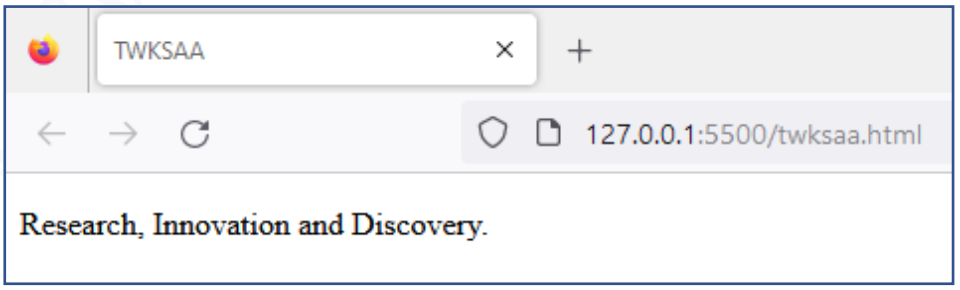
HTML Paragraph:

- HTML paragraph tag is used to define a paragraph in a webpage.
- A paragraph always starts on a new line, and is usually a block of text.
- paragraph

Example:

twksaa.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body>
<p>Research, Innovation and Discovery. </p>
</body>
</html>
```

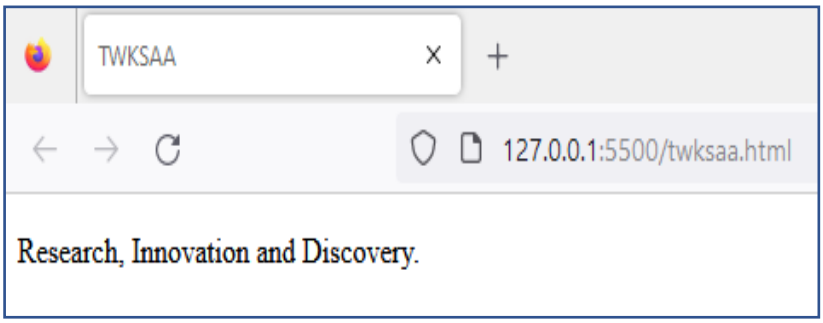


❖ Space inside HTML Paragraph:

- If you put a lot of spaces inside the HTML p tag, browser removes extra spaces and extra line while displaying the page. The browser counts number of spaces and lines as a single one.

Example:

```
<!DOCTYPE html>
<head>
<title>TWKSAA</title>
  </head>
<body>
<p>Research,
Innovation and
      Discovery.</p>
</body>
</html>
```



❖ HTML <pre> Element

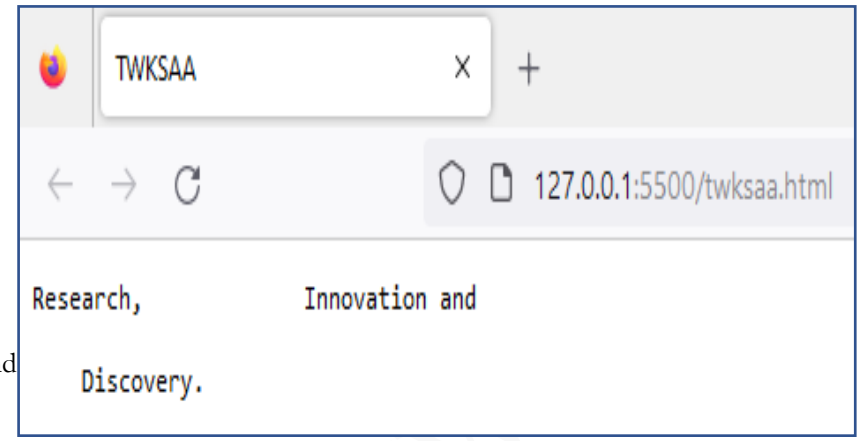
- The HTML <pre> element defines preformatted text.
- The text inside a <pre> element is displayed in a fixed-width font and it preserves both spaces and line breaks:

Example:

twksaa.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body>
<pre>Research,  Innovation and

Discovery.</pre>
</body> </html>
```

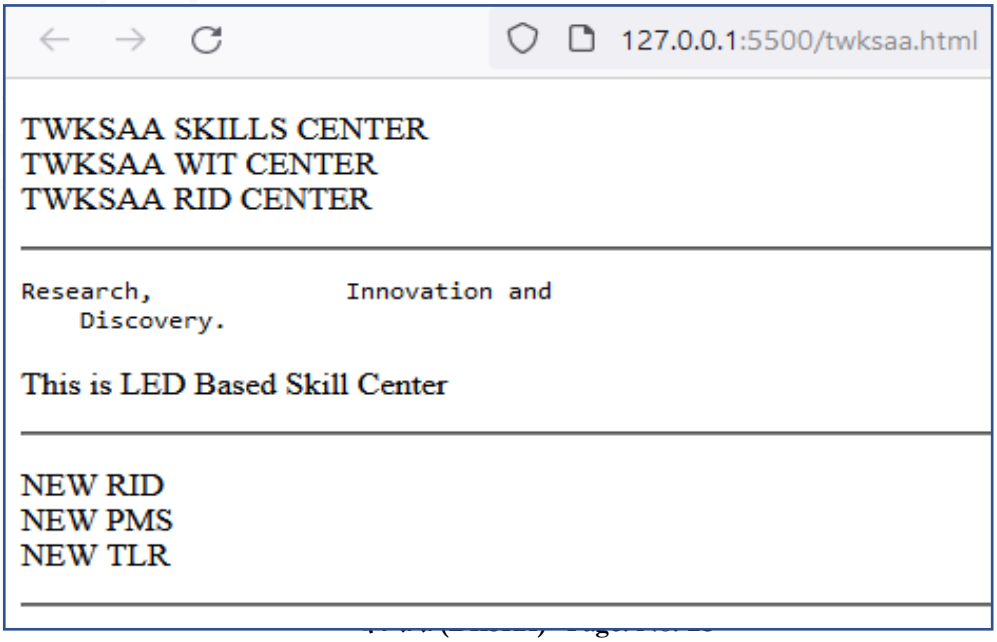


❖ What is Use of <br> and <hr> tag?

- **
** tag is used for line break and it can be used with paragraph elements.
- **<hr>** tag is used to apply a horizontal line between two statements or two paragraphs.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
</head>
<body>
   <p>TWKSAA SKILLS CENTER <br>TWKSAA WIT CENTER <br>TWKSAA RID CENTER </p><hr>
<pre>Research,  Innovation and
   Discovery.</pre>
   <p>This is LED Based Skill Center</p><hr>
   <p>NEW RID <br> NEW PMS <br> NEW TLR </p> <hr>
</body> </html>
```



HTML Text Formatting:

- HTML Formatting is a process of formatting text for better look. HTML provides us ability to format text without using CSS.
- In HTML the formatting tags are divided into two categories:

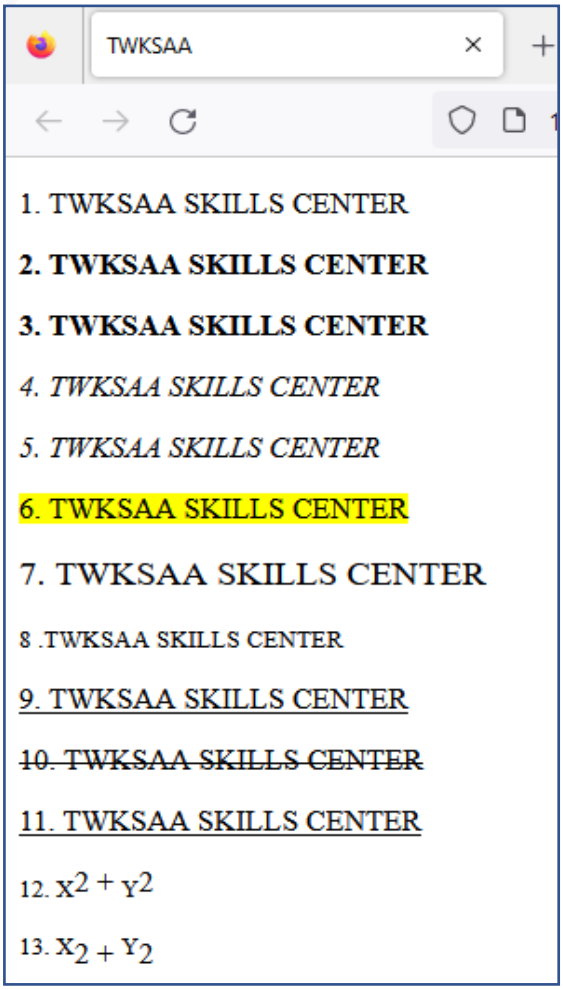
1. **Physical tag:** These tags are used to provide the visual appearance to the text.
2. **Logical tag:** These tags are used to add some logical or semantic value to the text.

Example:

- **<b>** : This is a physical tag, which is used to bold the text written between it.
- **** : This is a logical tag, which tells the browser that the text is important.
- **<i>** : This is a physical tag which is used to make text italic.
- **** : This is a logical tag which is used to display content in italic.
- **<mark>** : This tag is used to highlight text.
- **<big>** : This tag is used to increase the font size by one conventional unit.
- **<small>** : This tag is used to decrease the font size by one unit from base **size.**
- **<u>** : This tag is used to underline text written between it.
- **<del>** : This tag is used to display the deleted content.
- **<ins>** : This tag displays the content which is added
- **<sub>** : It displays the content slightly below the normal line.
- **<sup>** : It displays the content slightly above the normal line.

Example:
twksaa.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body>
   <p>1. TWKSAA SKILLS CENTER </p>
   <p><b>2.</b> TWKSAA SKILLS CENTER <b></b> </p>
   <p><strong>3.</strong> TWKSAA SKILLS CENTER<strong></strong></p>
   <p><i>4.</i> TWKSAA SKILLS CENTER<i></i></p>
   <p><em>5.</em> TWKSAA SKILLS CENTER<em></em></p>
   <p><mark>6.</mark> TWKSAA SKILLS CENTER<mark></mark></p>
   <p><big>7.</big> TWKSAA SKILLS CENTER<big></big></p>
   <p><small>8.</small> TWKSAA SKILLS CENTER<small></small></p>
   <p><u>9.</u> TWKSAA SKILLS CENTER<u></u></p>
   <p><del>10.</del> TWKSAA SKILLS CENTER<del></del></p>
   <p><ins>11.</ins> TWKSAA SKILLS CENTER<ins></ins></p>
   <p><sub>12.</sub> X</sub>2 + <sub>Y</sub>2</p>
   <p><sup>13.</sup> X</sup>2 + <sup>Y</sup>2</p>
</body>
</html>
```



HTML Quotation and Citation Elements:

- HTML provides elements to represent quotations and citations in web documents.
- These elements help structure and attribute quotes and citations properly

❖ TML Quotation and Citation Elements

Tag	Description
<abbr>	Defines an abbreviation or acronym
<address>	Defines contact information for the author/owner of a document
<bdo>	Defines the text direction
<blockquote>	Defines a section that is quoted from another source
<cite>	Defines the title of a work
<q>	Defines a short inline quotation

- **Example:**

1. <blockquote>` Element:

- The `<blockquote>` element is used to represent a section of text that is a quotation from another source. It is typically indented or styled to distinguish it from the surrounding text.

Example:

```
<blockquote>
  <p>This is a quoted text from another source.</p>
</blockquote>
```

- The `<blockquote>` element can also include a `cite` attribute to provide the source URL or reference:

```
<blockquote cite="https://www.example.com">
        <p>This is a quoted text from another source.</p>
 </blockquote>
```

2. <q>` Element:

- The `<q>` element is used for inline quotations within a paragraph or sentence. Browsers typically add quotation marks around the content inside the `<q>` element.

Example:

```
<p>She said, <q>Life is beautiful.</q></p>
```

3. <cite>` Element:

- The `<cite>` element is used to indicate the title of a creative work, such as a book, movie, or song, and can be used within the `<blockquote>` or `<q>` elements to provide the source's title.

Example:

```
<blockquote>
  <p>This is a quoted text from the book <cite>The Great Gatsby</cite>.</p>
</blockquote>
```

4. <abbr>` Element:

- The `<abbr>` element is used to define an abbreviation or acronym. It can include a `title` attribute to provide the full expanded form of the abbreviation, which serves as a citation.

Example:

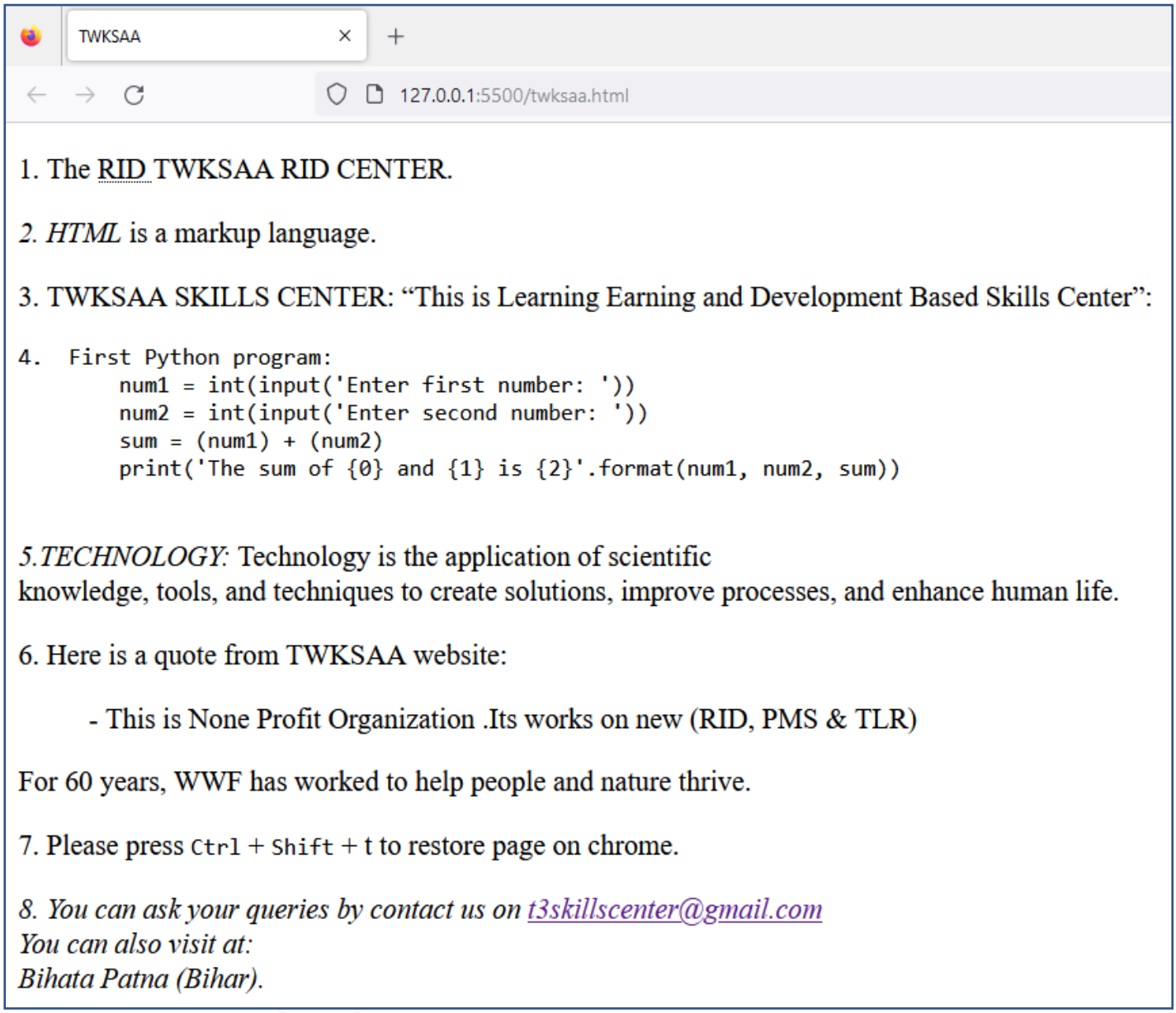
```
<p>The <abbr title="World Health Organization">WHO</abbr> provides health
guidelines.</p>
```

HTML Phrase tag:

- The HTML phrase tags are special purpose tags, which defines the structural meaning of a block of text or semantics of text.
  - Abbreviation tag : **<abbr>**
  - Definition tag: **<dfn>**
  - Quoting tag: **<blockquote>**
  - Short quote tag : **<q>**
  - Code tag: **<code>**
  - Keyboard tag: **<kbd>**
  - Address tag: **<address**
  - Renders in italic: **<cite>**
  - Quoted from another source: **<blockquote>**

Example: twksaa.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head><body>
   <p>1. The <abbr title = "Research Innovation and Development">RID </abbr> TWKSAA RID CENTER.
      <p><dfn>2. HTML </dfn> is a markup language. </p>
      <p>3. TWKSAA SKILLS CENTER: <q>This is Learning Earning and Development Based Skills
Center</q>:</p>
     <pre>4.  First Python program: <code>
      num1 = int(input('Enter first number: '))
      num2 = int(input('Enter second number: '))
      sum = (num1) + (num2)
      print('The sum of {0} and {1} is {2}'.format(num1, num2, sum)) </code>
   </pre>
   <p><cite>5.TECHNOLOGY: </cite> Technology is the application of scientific <br>
       knowledge, tools, and techniques to create solutions, improve processes, and enhance human life.</p>
       <p>6. Here is a quote from TWKSAA website:</p>
       <blockquote cite="http://www.twksaa.org">
          - This is None Profit Organization .Its works on new (RID, PMS & TLR)
       </blockquote>
          For 60 years, WWF has worked to help people and nature thrive.
    <p>7. Please press <kbd>Ctrl</kbd> + <kbd>Shift</kbd> + t<kbd></kbd> to restore page on chrome.</p>
    <address>8.  You can ask your queries by contact us on <a href="">t3skillscenter@gmail.com</a>
      <br> You can also visit at: <br> Bihata Patna (Bihar).
     </address>
</body></html>
```



HTML Comment Tag:

- You can add comments to your HTML source by using the following syntax:
- <! -- Write your comments here -->
- Notice that there is an exclamation point (!) in the start tag, but not in the end tag.
- Comments can be used to hide content.

Note: Comments are not displayed by the browser.

- **Example:**

```
<! -- This is a comment -->
<p>This is a paragraph. </p>
<! -- Remember to add more information here -->
```

- **Example Hide a part of a paragraph:**

```
<p>This <! -- great text --> is a paragraph. </p>
```

- **Example Hide a section of HTML code:**

```
<p>This is a paragraph. </p>
<!--
<p>Look at this cool image:</p>
<img border="100" src="t3.jpg" alt="image">
-->
<p>This is a paragraph too.</p>
```

HTML Styles:

- The HTML style attribute is used to add styles to an element, such as color, font, size etc.

Syntax:

- **<tag name style="property: value;">**

Where: The property is a **CSS property**. The value is a **CSS value**.

1. **Background Color:**

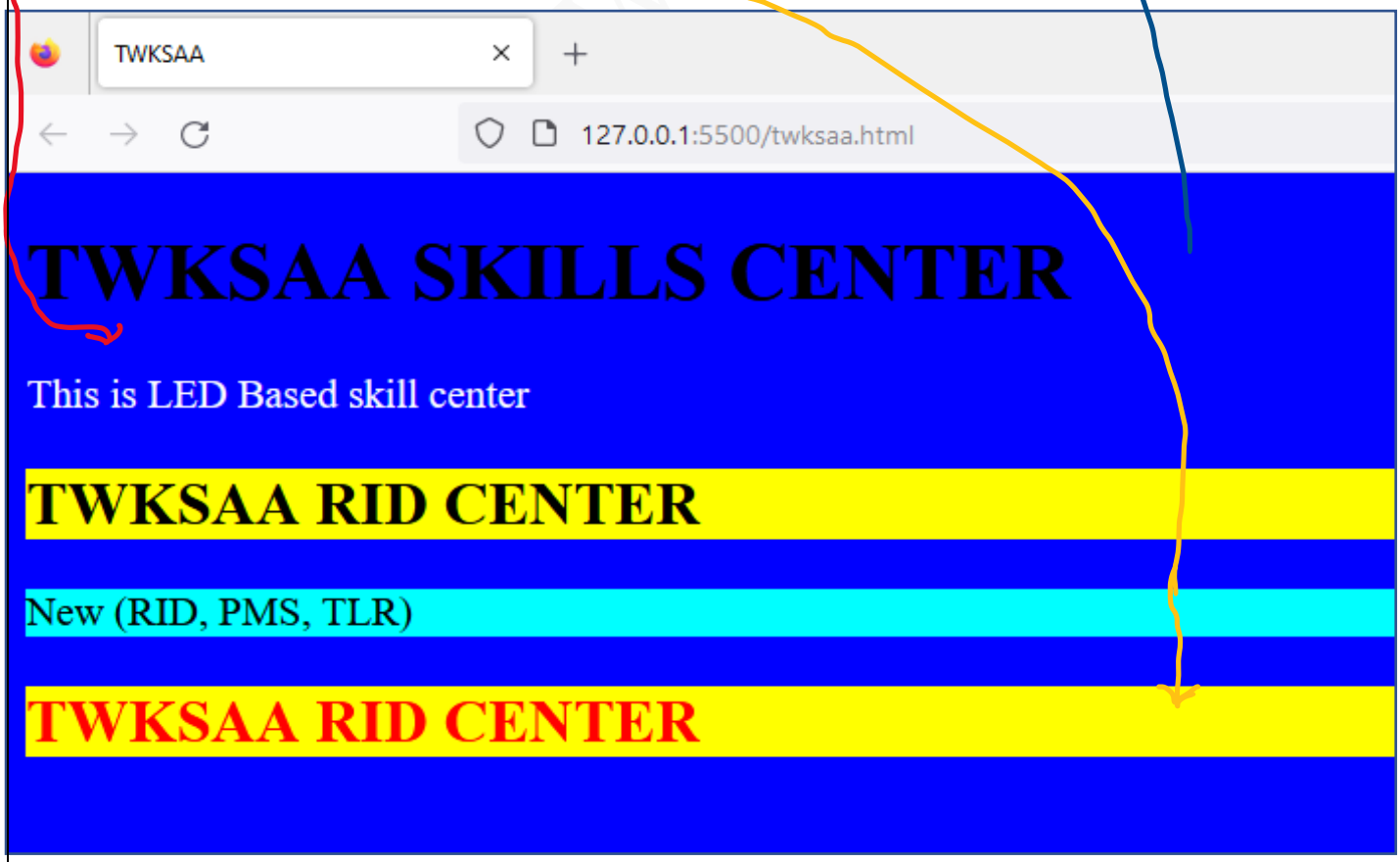
- The CSS background-color property defines the background color for an HTML element.

2. **Text Color:**

- The CSS color property defines the text color for an HTML element:

Example: twksaa.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body style="background-color: blue;">
   <h1>TWKSAA SKILLS CENTER</h1>
   <P style="color: white;">This is LED Based skill center</P>
   <h2 style="background-color: yellow;">TWKSAA RID CENTER</h2>
   <P style="background-color: aqua;">New (RID, PMS, TLR )</P>
   <h2 style="background-color: yellow; color: red;">TWKSAA RID CENTER</h2>
</body></html>
```

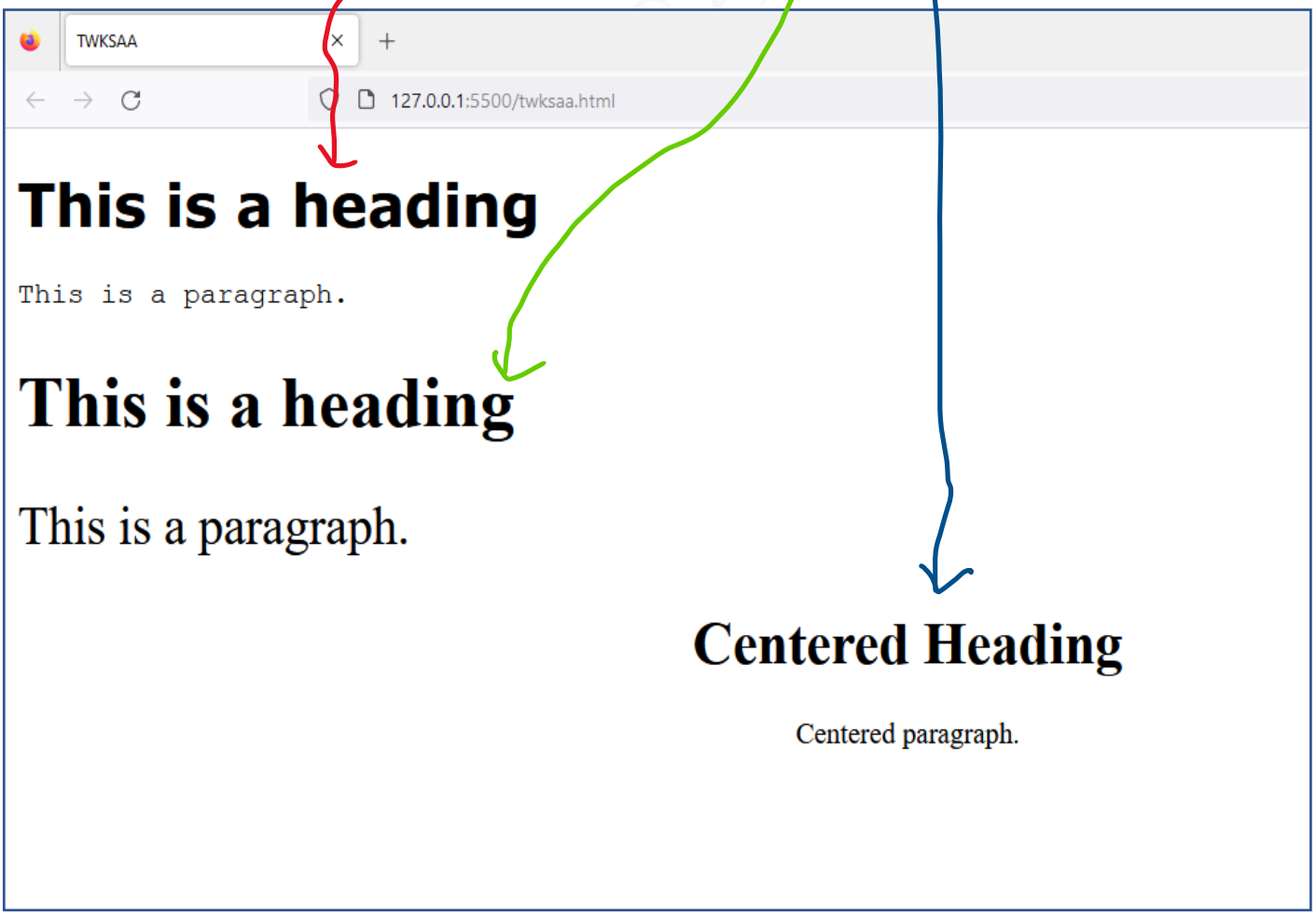


3. **font-family:** for text fonts
4. **font-size:** for text sizes
5. **text-align:** for text alignment

Example:

Twksaa.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body >
   <h1 style="font-family: verdana;">This is a heading</h1>
   <p style="font-family: courier;">This is a paragraph. </p>
   <h1 style="font-size:250%;">This is a heading</h1>
   <p style="font-size:180%;">This is a paragraph.</p>
   <h1 style="text-align: center;">Centered Heading</h1>
   <p style="text-align: center;">Centered paragraph. </p>
</body>
</html>
```



HTML Colors:

- HTML colors are specified with predefined color names, or with RGB, HEX, HSL, RGBA, or HSLA values
- Color Names Supported by All Browsers
- All modern browsers support the following 140 color names

1. AliceBlue
   - RGB: 240, 248, 255
   - HEX: #F0F8FF
   - HSL: 208, 100%, 97%
   - RGBA: 240, 248, 255, 1
   - HSLA: 208, 100%, 97%, 1

2. AntiqueWhite
   - RGB: 250, 235, 215
   - HEX: #FAEBD7
   - HSL: 34, 78%, 91%
   - RGBA: 250, 235, 215, 1
   - HSLA: 34, 78%, 91%, 1

3. Aqua
   - RGB: 0, 255, 255
   - HEX: #00FFFF
   - HSL: 180, 100%, 50%
   - RGBA: 0, 255, 255, 1
   - HSLA: 180, 100%, 50%, 1

4. Aquamarine
   - RGB: 127, 255, 212
   - HEX: #7FFFD4
   - HSL: 160, 100%, 75%
   - RGBA: 127, 255, 212, 1
   - HSLA: 160, 100%, 75%, 1

5. Azure
   - RGB: 240, 255, 255
   - HEX: #F0FFFF
   - HSL: 180, 100%, 97%
   - RGBA: 240, 255, 255, 1
   - HSLA: 180, 100%, 97%, 1

6. Beige
   - RGB: 245, 245, 220
   - HEX: #F5F5DC
   - HSL: 60, 56%, 91%
   - RGBA: 245, 245, 220, 1
   - HSLA: 60, 56%, 91%, 1

7. Bisque
   - RGB: 255, 228, 196
   - HEX: #FFE4C4
   - HSL: 33, 100%, 88%
   - RGBA: 255, 228, 196, 1
   - HSLA: 33, 100%, 88%, 1

8. Black
   - RGB: 0, 0, 0
   - HEX: #000000
   - HSL: 0, 0%, 0%
   - RGBA: 0, 0, 0, 1
   - HSLA: 0, 0%, 0%, 1

9. BlanchedAlmond
   - RGB: 255, 235, 205
   - HEX: #FFEBCD
   - HSL: 36, 100%, 90%
   - RGBA: 255, 235, 205, 1
   - HSLA: 36, 100%, 90%, 1

10. Blue
    - RGB: 0, 0, 255
    - HEX: #0000FF
    - HSL: 240, 100%, 50%
    - RGBA: 0, 0, 255, 1
    - HSLA: 240, 100%, 50%, 1

11. BlueViolet
    - RGB: 138, 43, 226
    - HEX: #8A2BE2
    - HSL: 271, 76%, 53%
    - RGBA: 138, 43, 226, 1
    - HSLA: 271, 76%, 53%, 1

12. Brown
    - RGB: 165, 42, 42
    - HEX: #A52A2A
    - HSL: 0, 59%, 41%
    - RGBA: 165, 42, 42, 1
    - HSLA: 0, 59%, 41%, 1

13. BurlyWood
    - RGB: 222, 184, 135
    - HEX: #DEB887
    - HSL: 34, 57%, 70%
    - RGBA: 222, 184, 135, 1
    - HSLA: 34, 57%, 70%, 1

14. CadetBlue
    - RGB: 95, 158, 160
    - HEX: #5F9EA0
    - HSL: 182, 25%, 50%
    - RGBA: 95, 158, 160, 1
    - HSLA: 182, 25%, 50%, 1

15. Chartreuse
    - RGB: 127, 255, 0
    - HEX: #7FFF00
    - HSL: 90, 100%, 50%
    - RGBA: 127, 255, 0, 1
    - HSLA: 90, 100%, 50%, 1

16. Chocolate
    - RGB: 210, 105, 30
    - HEX: #D2691E
    - HSL: 25, 75%, 47%
    - RGBA: 210, 105, 30, 1
    - HSLA: 25, 75%, 47%, 1

17. Coral
    - RGB: 255, 127, 80
    - HEX: #FF7F50
    - HSL: 16, 100%, 66%
    - RGBA: 255, 127, 80, 1
    - HSLA: 16, 100%, 66%, 1

18. CornflowerBlue
    - RGB: 100, 149, 237
    - HEX: #6495ED
    - HSL: 219, 79%, 66%
    - RGBA: 100, 149, 237, 1
    - HSLA: 219, 79%, 66%, 1

19. Cornsilk
    - RGB: 255, 248, 220
    - HEX: #FFF8DC
    - HSL: 48, 100%, 93%
    - RGBA: 255, 248, 220, 1
    - HSLA: 48, 100%, 93%, 1

20. Crimson
    - RGB: 220, 20, 60
    - HEX: #DC143C
    - HSL: 348, 83%, 47%
    - RGBA: 220, 20, 60, 1
    - HSLA: 348, 83%,47%, 1

21. Cyan
    - RGB: 0, 255, 255
    - HEX: #00FFFF
    - HSL: 180, 100%, 50%
    - RGBA: 0, 255, 255, 1
    - HSLA: 180, 100%, 50%

22. DarkBlue
    - RGB: 0, 0, 139
    - HEX: #00008B
    - HSL: 240, 100%, 27%
    - RGBA: 0, 0, 139, 1
    - HSLA: 240, 100%, 27%, 1

23. DarkCyan
    - RGB: 0, 139, 139
    - HEX: #008B8B
    - HSL: 180, 100%, 27%
    - RGBA: 0, 139, 139, 1
    - HSLA: 180, 100%, 27%, 1

24. DarkGoldenRod
    - RGB: 184, 134, 11
    - HEX: #B8860B
    - HSL: 43, 89%, 38%
    - RGBA: 184, 134, 11, 1
    - HSLA: 43, 89%, 38%, 1

25. DarkGray
    - RGB: 169, 169, 169
    - HEX: #A9A9A9
    - HSL: 0, 0%, 66%
    - RGBA: 169, 169, 169, 1
    - HSLA: 0, 0%, 66%, 1

26. DarkGreen
    - RGB: 0, 100, 0
    - HEX: #006400
    - HSL: 120, 100%, 20%
    - RGBA: 0, 100, 0, 1
    - HSLA: 120, 100%, 20%, 1

27. DarkKhaki
    - RGB: 189, 183, 107
    - HEX: #BDB76B
    - HSL: 56, 38%, 58%
    - RGBA: 189, 183, 107, 1
    - HSLA: 56, 38%, 58%, 1

28. DarkMagenta
    - RGB: 139, 0, 139
    - HEX: #8B008B
    - HSL: 300, 100%, 27%
    - RGBA: 139, 0, 139, 1
    - HSLA: 300, 100%, 27%, 1

29. DarkOliveGreen
- RGB: 85, 107, 47
- HEX: #556B2F
- HSL: 82, 39%, 30%
- RGBA: 85, 107, 47, 1
- HSLA: 82, 39%, 30%, 1

30. DarkOrange
- RGB: 255, 140, 0
- HEX: #FF8C00
- HSL: 33, 100%, 50%
- RGBA: 255, 140, 0, 1
- HSLA: 33, 100%, 50%, 1

31. DarkOrchid
- RGB: 153, 50, 204
- HEX: #9932CC
- HSL: 280, 61%, 50%
- RGBA: 153, 50, 204, 1
- HSLA: 280, 61%, 50%, 1

32. DarkRed
- RGB: 139, 0, 0
- HEX: #8B0000
- HSL: 0, 100%, 27%
- RGBA: 139, 0, 0, 1
- HSLA: 0, 100%, 27%, 1

33. DarkSalmon
- RGB: 233, 150, 122
- HEX: #E9967A
- HSL: 15, 72%, 70%
- RGBA: 233, 150, 122, 1
- HSLA: 15, 72%, 70%, 1

34. DarkSeaGreen
- RGB: 143, 188, 143
- HEX: #8FBC8F
- HSL: 120, 25%, 65%
- RGBA: 143, 188, 143, 1
- HSLA: 120, 25%, 65%, 1

35. DarkSlateBlue
- RGB: 72, 61, 139
- HEX: #483D8B
- HSL: 248, 39%, 39%
- RGBA: 72, 61, 139, 1
- HSLA: 248, 39%, 39%, 1

36. DarkSlateGray
- RGB: 47, 79, 79
- HEX: #2F4F4F
- HSL: 180, 25%, 25%
- RGBA: 47, 79, 79, 1
- HSLA: 180, 25%, 25%, 1

37. DarkTurquoise
- RGB: 0, 206, 209
- HEX: #00CED1
- HSL: 181, 100%, 41%
- RGBA: 0, 206, 209, 1
- HSLA: 181, 100%, 41%, 1

38. DarkViolet
- RGB: 148, 0, 211
- HEX: #9400D3
- HSL: 282, 100%, 41%
- RGBA: 148, 0, 211, 1
- HSLA: 282, 100%, 41%, 1

39. DeepPink
- RGB: 255, 20, 147
- HEX: #FF1493
- HSL: 328, 100%, 54%
- RGBA: 255, 20, 147, 1
- HSLA: 328, 100%, 54%,

40. DeepSkyBlue
- RGB: 0, 191, 255
- HEX: #00BFFF
- HSL: 195, 100%, 50%
- RGBA: 0, 191, 255, 1
- HSLA: 195, 100%, 50%, 1

41. DimGray
- RGB: 105, 105, 105
- HEX: #696969
- HSL: 0, 0%, 41%
- RGBA: 105, 105, 105, 1
- HSLA: 0, 0%, 41%, 1

42. DodgerBlue
- RGB: 30, 144, 255
- HEX: #1E90FF
- HSL: 210, 100%, 56%
- RGBA: 30, 144, 255, 1
- HSLA: 210, 100%, 56%, 1

43. FireBrick
- RGB: 178, 34, 34
- HEX: #B22222
- HSL: 0, 68%, 42%
- RGBA: 178, 34, 34, 1
- HSLA: 0, 68%, 42%, 1

44. FloralWhite
- RGB: 255, 250, 240
- HEX: #FFFAF0
- HSL: 40, 100%, 97%
- RGBA: 255, 250, 240, 1
- HSLA: 40, 100%, 97%, 1

45. ForestGreen
- RGB: 34, 139, 34
- HEX: #228B22
- HSL: 120, 61%, 34%
- RGBA: 34, 139, 34, 1
- HSLA: 120, 61%, 34%, 1

46. Fuchsia
- RGB: 255, 0, 255
- HEX: #FF00FF
- HSL: 300, 100%, 50%
- RGBA: 255, 0, 255, 1
- HSLA: 300, 100%, 50%, 1

47. Gainsboro
- RGB: 220, 220, 220
- HEX: #DCDCDC
- HSL: 0, 0%, 86%
- RGBA: 220, 220, 220, 1
- HSLA: 0, 0%, 86%, 1

48. GhostWhite
- RGB: 248, 248, 255
- HEX: #F8F8FF
- HSL: 240, 100%, 99%
- RGBA: 248, 248, 255, 1
- HSLA: 240, 100%, 99%, 1

49. Gold
- RGB: 255, 215, 0
- HEX: #FFD700
- HSL: 51, 100%, 50%
- RGBA: 255, 215, 0, 1
- HSLA: 51, 100%, 50%, 1

50. GoldenRod
- RGB: 218, 165, 32
- HEX: #DAA520
- HSL: 43, 74%, 49%
- RGBA: 218, 165, 32, 1
- HSLA: 43, 74%, 49%, 1

51. Gray
- RGB: 128, 128, 128
- HEX: #808080
- HSL: 0, 0%, 50%
- RGBA: 128, 128, 128, 1
- HSLA: 0, 0%, 50%, 1

52. Green
- RGB: 0, 128, 0
- HEX: #008000
- HSL: 120, 100%, 25%
- RGBA: 0, 128, 0, 1
- HSLA: 120, 100%, 25%, 1

53. GreenYellow
- RGB: 173, 255, 47
- HEX: #ADFF2F
- HSL: 84, 100%, 59%
- RGBA: 173, 255, 47, 1
- HSLA: 84, 100%, 59%, 1

4. HoneyDew
- RGB: 240, 255, 240
- HEX: #F0FFF0
- HSL: 120, 100%, 97%
- RGBA: 240, 255, 240, 1
- HSLA: 120, 100%, 97%, 1

55. HotPink
- RGB: 255, 105, 180
- HEX: #FF69B4
- HSL: 330, 100%, 71%
- RGBA: 255, 105, 180, 1
- HSLA: 330, 100%, 71%, 1

56. IndianRed
- RGB: 205, 92, 92
- HEX: #CD5C5C
- HSL: 0, 53%, 58%
- RGBA: 205, 92, 92, 1
- HSLA: 0, 53%, 58%, 1

57. Indigo
- RGB: 75, 0, 130
- HEX: #4B0082
- HSL: 275, 100%, 25%
- RGBA: 75, 0, 130, 1
- HSLA: 275, 100%, 25%, 1

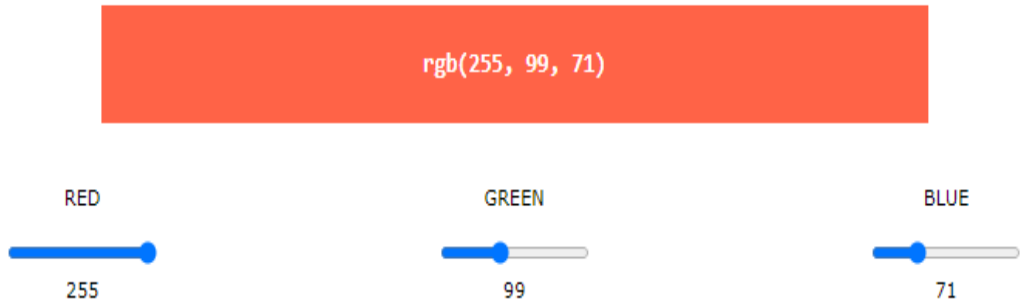
58. Ivory
- RGB: 255, 255, 240
- HEX: #FFFFF0
- HSL: 60, 100%, 97%
- RGBA: 255, 255, 240, 1
- HSLA: 60, 100%, 97%, 1

59. Khaki
- RGB: 240, 230, 140
- HEX: #F0E68C
- HSL: 54, 77%, 75%
- RGBA: 240, 230, 140, 1
- HSLA: 54, 77%, 75%, 1

60. Lavender
- RGB: 230, 230, 250
- HEX: #E6E6FA
- HSL: 240, 67%, 94%
- RGBA: 230, 230, 250, 1
- HSLA: 240, 67%, 94%, 1

❖ HTML RGB and RGBA Colors:

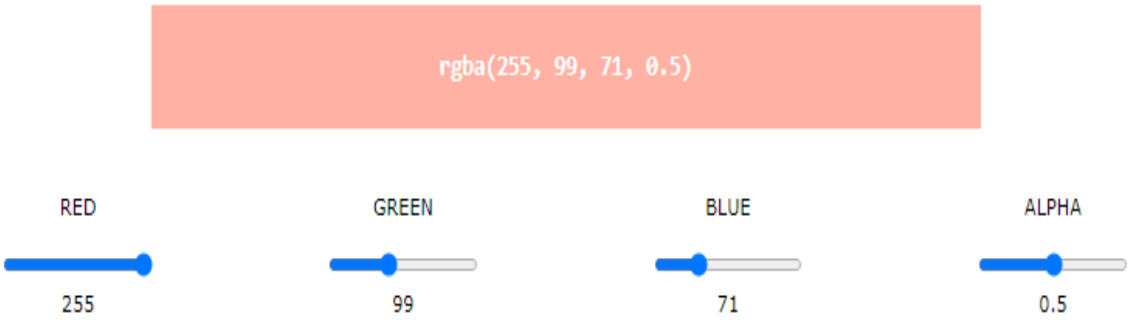
- An RGB color value represents RED, GREEN, and BLUE light sources
- An RGBA color value is an extension of RGB with an Alpha channel (opacity).
- **RGB Color Values**
- In HTML, a color can be specified as an RGB value, using this formula:
- **rgb (red, green, blue)**



Example

rgb(255, 0, 0)
rgb(0, 0, 255)
rgb(60, 179, 113)
rgb(238, 130, 238)
rgb(255, 165, 0)
rgb(106, 90, 205)

- **RGBA Color Values:**
- RGBA color values are an extension of RGB color values with an Alpha channel - which specifies the opacity for a color.
- An RGBA color value is specified with:
  - **rgba (red, green, blue, alpha)**
- Alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all):



Example

rgba(255, 99, 71, 0)

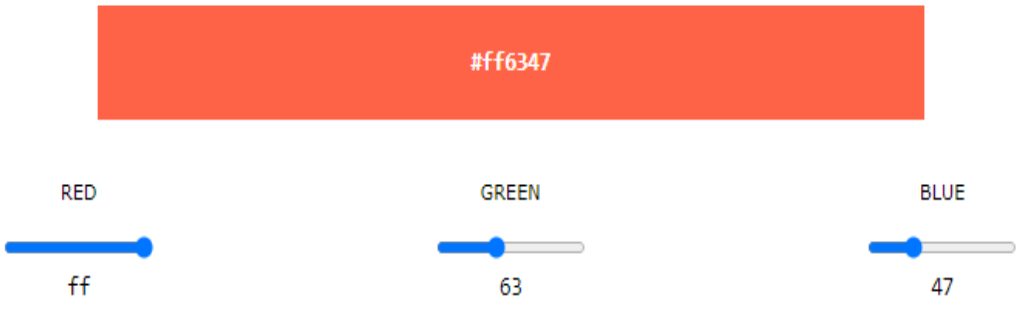
rgba(255, 99, 71, 0.2)

rgba(255, 99, 71, 0.4)

rgba(255, 99, 71, 0.6)

rgba(255, 99, 71, 0.8)

rgba(255, 99, 71, 1)



Example

#ff0000

#0000ff

#3cb371

#ee82ee

#ffa500

#6a5acd

Shades of Gray

Shades of gray are often defined using equal values for all three parameters:

Example

#404040 #686868

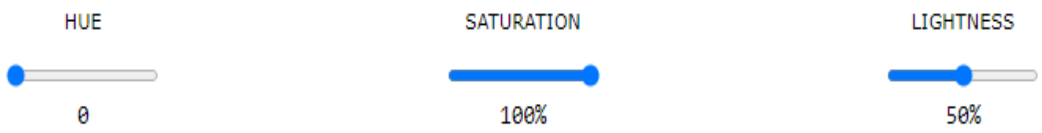
#a0a0a0 #bebebe

#dcdcdc #f8f8f8

❖ HTML HSL and HSLA Colors:

- HSL stands for hue, saturation, and lightness.
- HSLA color values are an extension of HSL with an Alpha channel (opacity).
  - **HSL Color Values**
- In HTML, a color can be specified using hue, saturation, and lightness (HSL) in the form:
  - **hsl (hue, saturation, lightness)**

hsl(0, 100%, 50%)



Example

hsl(0, 100%, 50%) hsl(240, 100%, 50%)

hsl(147, 50%, 47%) hsl(300, 76%, 72%)

hsl(39, 100%, 50%) hsl(248, 53%, 58%)

Saturation

Saturation can be described as the intensity of a color.

100% is pure color, no shades of gray.

50% is 50% gray, but you can still see the color.

0% is completely gray; you can no longer see the color.

Example

hsl(0, 100%, 50%)

hsl(0, 80%, 50%)

hsl(0, 60%, 50%)

hsl(0, 40%, 50%)

hsl(0, 20%, 50%)

hsl(0, 0%, 50%)

Lightness

The lightness of a color can be described as how much light you want to give the color, where 0% means no light (black), 50% means 50% light (neither dark nor light), and 100% means full lightness (white).

Example

hsl(0, 100%, 0%)

hsl(0, 100%, 25%)

hsl(0, 100%, 50%)

hsl(0, 100%, 75%)

hsl(0, 100%, 90%)

hsl(0, 100%, 100%)

Shades of Gray

Shades of gray are often defined by setting the hue and saturation to 0, and adjusting the lightness from 0% to 100% to get darker/lighter shades:

Example

hsl(0, 0%, 20%)

hsl(0, 0%, 30%)

hsl(0, 0%, 40%)

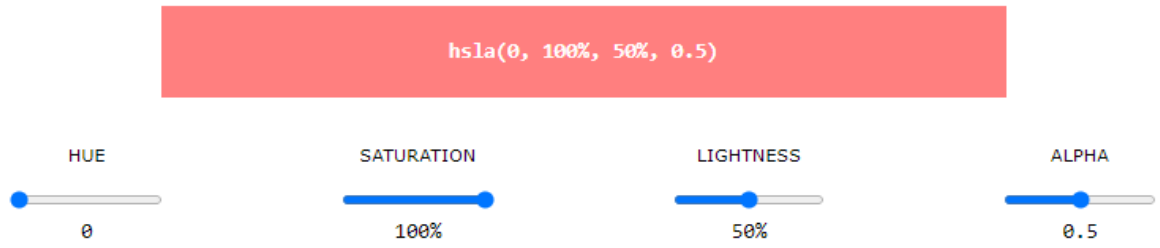
hsl(0, 0%, 60%)

hsl(0, 0%, 70%)

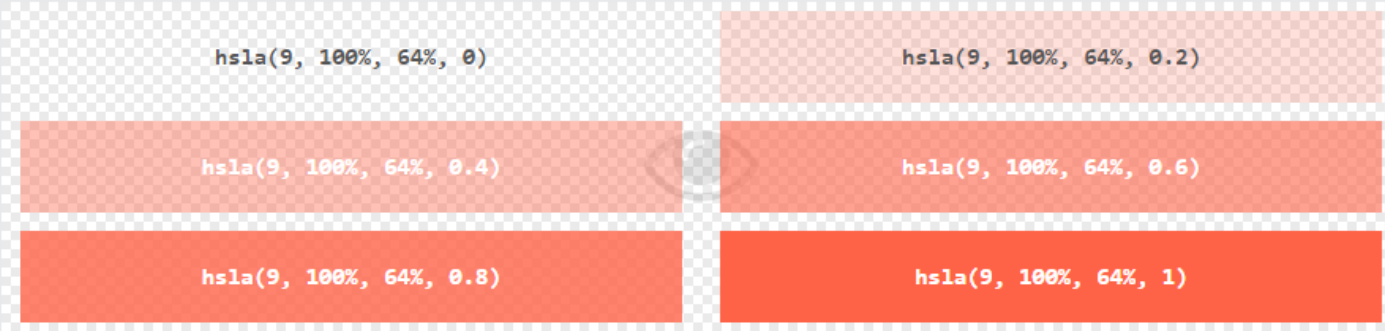
hsl(0, 0%, 90%)

❖ HSLA Color Values:

- HSLA color values are an extension of HSL color values, with an Alpha channel - which specifies the opacity for a color.
  - **hsla(hue, saturation, lightness, alpha)**
- alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all):



Example



HTML ANCHOR TAG:

- The HTML anchor tag defines a hyperlink that links one page to another page. It can create hyperlink to other web page as well as files, location, or any URL. The "href" attribute is the most important attribute of the HTML a tag. and which links to destination page or URL.

Where:

- href attribute is used to define the address of the file to be linked.

Syntax:

- ** write about link**

Example:

- **Click for third Page**
- Specify a location for Link using **target** attribute:

Note:

- The target attribute can only use with href attribute in anchor tag.
- If we will not use target attribute then link will open in same page.

Example:

- ** this-link **
- ** this-link **
- ** this-link **
- ** this-link **
- ** this-link **

1. **target="_blank":**

- when the link is clicked, the linked content should open in a new browser window or tab.

2. **target="_self**:

- when the link is clicked, the linked content should open in the same browsing context or frame as the current page.

3. **target="_parent":**

- when the link is clicked, the linked content should open in the parent browsing context or frame of the current frame, if the current frame has a parent. If the current frame does not have a parent, the link behaves as if target="_self" was used.

4. **target="_top":**

- when the link is clicked, the linked content should open in the top-level browsing context, breaking out of all frames and iframes.

❖ **Appearance of HTML anchor tag:**

- An unvisited link is displayed underlined and blue.
- A visited link displayed underlined and purple.
- An active link is underlined and red.

❖ **HTML Links - Create Bookmarks:**

- HTML links can be used to create bookmarks, so that readers can jump to specific parts of a web page.
- Bookmarks can be useful if a web page is very long.
- To create a bookmark - first create the bookmark, then add a link to it.
- When the link is clicked, the page will scroll down or up to the location with the bookmark.

Example:

- use the id attribute to create a bookmark:
- **<h2 id="C3">Chapter 3</h2>**
- Then, add a link to the bookmark ("Jump to Chapter 3"), from within the same page:

Example:

- Jump to Chapter 4
- You can also add a link to a bookmark on another page:
- Jump to Chapter 4

5. Use the **id attribute (id="value")** to define bookmarks in a page.
6. Use the **href attribute (href="#value")** to link to the bookmark.

Example-1:

```
<!DOCTYPE html>
<html>
<head>
  <title>Anchor Tag Example</title>
</head>
<body>
  <h1>Welcome to My Website</h1>
  <p>Click the link below to visit our fictional website:</p>
  <a href="https://www.TWKSAA.Org" target="_blank">Visit twksaa.org</a>
</body></html>
```

Example-2: Creating bookmark

```
<!DOCTYPE html>
<html>
<head>
  <title>Bookmarks Example</title>
</head>  <body>
  <h1>Table of Contents</h1> <ul>
    <li><a href="#page1">Page 1</a></li>
    <li><a href="#page2">Page 2</a></li>
    <li><a href="#page3">Page 3</a></li>
    <li><a href="#page4">Page 4</a></li>
    <li><a href="#page5">Page 5</a></li>
    <li><a href="#page6">Page 6</a></li>  </ul>
  <h2 id="page1">Page 1</h2>
  <p>This is the content of Page 1.</p>
  <h2 id="page2">Page 2</h2>
  <p>This is the content of Page 2.</p>
  <h2 id="page3">Page 3</h2>
  <p>This is the content of Page 3.</p>
  <h2 id="page4">Page 4</h2>
  <p>This is the content of Page 4.</p>
  <h2 id="page5">Page 5</h2>
  <p>This is the content of Page 5.</p>
  <h2 id="page6">Page 6</h2>
  <p>This is the content of Page 6.</p>
</body></html
```

HTML IMAGE:

- img tag is used to display image on the web page.
- he <img> tag is empty, it contains attributes only, and does not have a closing tag.

Syntax:

- **<img src="URL" alt="alternate text">**

❖ Attributes of HTML img tag:

1) src: -describes the source or path of the image.

2) alt: -The alt attribute defines an alternate text for the image, if it can't be displayed.

3) width: - used to specify the width to display the image.

4) height: - height of the image.

5). Style: - Apply Style

Example: -

<img src="T3.jpg" height="100" width="300" alt="car image">

<img src="T3.jpg" alt="logo" style="width:500px; height:600px;">

Images on Another Server/Website:

- To point to an image on another server, you must specify an full URL in the src attribute.

Example:

- <img src="https://www.twksaa.org/images/new_rid.jpg" alt="New_img "

❖ Common Image Formats:

- Here are the most common image file types, which are supported in all browsers.

Abbreviation	File Format	File Extension
• **APNG**	: Animated Portable Network Graphics	.apng
• **GIF**	: Graphics Interchange Format	.gif
• **ICO**	: Microsoft Icon	.ico, .cur
• **JPEG**	: Joint Photographic Expert Group image	.jpg, .jpeg, .jfif, .pjpeg, .pjp
• **PNG**	: Portable Network Graphics	.png
• **SVG**	: Scalable Vector Graphics	.svg

<img>

- ✓ Use the HTML **<img>** element to define an image
- ✓ Use the HTML **src** attribute to define the URL of the image
- ✓ Use the HTML **alt** attribute to define an alternate text for an image, if it cannot be displayed
- ✓ Use the HTML **width** and **height** attributes or the CSS width and height properties to define the size of the image
- ✓ Use the CSS **float** property to let the image float to the left or to the right

❖ HTML Background Images:

- A background image can be specified for almost any HTML element.
- To add a background image on an HTML element, use the HTML style attribute and the CSS background-image property.

Example:

- Add a background image on a HTML element:
- **<p style="background-image: url ('img_file_name.jpg');">**
- You can also specify the background image in the <style> element, in the <head> section.

HTML TABLE TAG:

HTML table tag is used to display data in tabular form (row * column).

<table> : It defines a table.
<tr> **:** It defines a row in a table.
<th> **:** It defines a header cell in a table.
<td> **:** It defines a cell in a table.
<caption> : It defines the table caption.
<col group> : It specifies a group of one or more columns in a table for formatting.
<tbody> : It is used to group the body content in a table.
<thead> : It is used to group the header content in a table.
<tfooter> : It is used to group the footer content in a table.
<colspan> **:** To make a cell span over multiple columns, use the colspan attribute:
<rowspan> **:** To make a cell span over multiple rows, use the rowspan attribute**:**
<col> : it is used with <col group> element to specify column properties for each column.

❖ Table Cells:

- Each table cell is defined by a <td> and a </td> tag.
- td stands for table data.
- Everything between <td> and </td> are the content of the table cell.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body >
   <table>
      <tr>
       <td>Email</td>
       <td>Name</td>
       <td>Mobile_No</td>
      </tr>
   </table>
</body>
</html>
```

Email_id Name Mobile_No

❖ Table Rows:

- Each table row starts with a <tr> and ends with a </tr> tag.
- tr stands for table row.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
```

Email_id Name Mobile
Ram ram@gmail.com 9854412468

```
</head>
<body >
  <table>
    <tr>
      <td>Email_id</td>
      <td>Name</td>
      <td>Mobile</td>
    </tr>
    <tr>
      <td>Ram</td>
      <td>ram@gmail.com</td>
      <td>9854412468</td>
    </tr>
  </table>
</body>
</html>
```

❖ Table Headers:

- if you want your cells to be table header cells. In those cases use the <th> tag instead of the <td> tag:
- th stands for table header.

Example:

twksaa.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>TWKSAA</title>
</head>
<body >
  <table>
    <tr>
      <th>S.NO</th>
      <th>Name</th>
      <th>Marks</th>
    </tr>
    <tr>
      <td>1</td>
      <td>Ravi</td>
      <td>96</td>
    </tr>
    <tr>
      <td>2</td>
      <td>Ramu</td>
      <td>98</td>
    </tr>
    <tr>
      <td>3</td>
      <td>Radha</td>
      <td>99</td>
    </tr>
  </table>
</body>
</html>
```

S.NO	Name	Marks
1	Ravi	96
2	Ramu	98
3	Radha	99

❖ HTML Table Borders:

Example: twksaa.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body >
   <table border="1"> <! - - 2,3,4  - ->
      <tr>
         <th>S.NO</th>
         <th>Name</th>
         <th>Marks</th>
      </tr>
      <tr>
         <td>1</td>
         <td>Ravi</td>
         <td>96</td>
      </tr>
      <tr>
         <td>2</td>
         <td>Ramu</td>
         <td>98</td>
      </tr>
      <tr>
         <td>3</td>
         <td>Radha</td>
         <td>99</td>
      </tr>
   </table>
</body>
</html>
```

S.NO	Name	Marks
1	Ravi	96
2	Ramu	98
3	Radha	99

❖ HTML Table Sizes:

- HTML tables can have different sizes for each column, row or the entire table.
- Use the style attribute with the width or height properties to specify the size of a table, row or column.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>TWKSAA</title>
</head>
<body >
  <table border="1", style="width: 50%;">
    <tr style="width: 80%;">
      <th style="width: 10%;">S.NO</th>
      <th>Name</th>
      <th>Marks</th>
    </tr>
    <tr>
      <td style="width: 40%;">1</td>
      <td>Ravi</td>
      <td>96</td>
    </tr>
    <tr style>
      <td>2</td>
      <td>Ramu</td>
      <td>98</td>
    </tr>
    <tr>
      <td>3</td>
      <td>Radha</td>
      <td style="height: 20vh;">99</td>
    </tr>
  </table>
</body>
</html>
```

S.NO	Name	Marks
1	Ravi	96
2	Ramu	98
3	Radha	99

HTML Table – rowspan:

- Rowspan is used to specified how many rows current cell need to occupied.

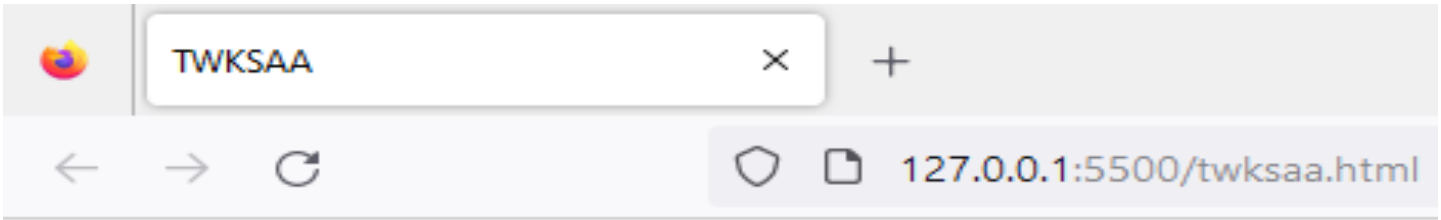
HTML Table -colspan:

- colspan is used to specified how many numbers of Coolum current cell need to be occupied.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>TWKSAA</title>
</head>
<body >
  <table border="1">
    <tr>
      <th colspan="2">Name</th>
      <th colspan="3">Name</th>
    </tr>
    <tr>
      <th>First Name</th>
      <th>Last Name</th>
      <th>s1</th>
      <th>s2</th>
      <th>s3</th>
    </tr>
    <tr>
      <td>RAVI</td>
      <td>KUMAR</td>
      <td>3</td>
      <td>6</td>
      <td>9</td>
    </tr>
    <tr>
      <td rowspan="2">SUMAN</td>
      <td rowspan="2">RAJ</td>
      <td>33</td>
      <td>63</td>
      <td>93</td>
    </tr>
    <tr>
      <td>333</td>
      <td>633</td>
      <td>933</td>
    </tr>
    <tr>
      <td rowspan="2">SANGAM</td>
      <td rowspan="2">KUMAR</td>
      <td>66</td>
```

```
        <td>636</td>
        <td>96</td>
      </tr>
      <tr>
        <td>666</td>
        <td>666</td>
        <td>966</td>
      </tr>
      <tr>
        <td rowspan="2" colspan="3">Summary</td>
        <td>Avg</td>
        <td>22</td>
      </tr>
    </table>
  </body>
</html>
```

Name		Name		
First Name	Last Name	s1	s2	s3
RAVI	KUMAR	3	6	9
SUMAN	RAJ	33	63	93
		333	633	933
SANGAM	KUMAR	66	636	96
		666	666	966
Summary			Avg	22

❖ Nested Table:

➢ How to Created Nested Table:

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body >
   <table border="1">
      <tr>
         <th>TWKSAA</th>
         <th>
            <table border="1">
               <tr>
                  <td>SHKILLS CENTER</td>
                  <td>WIT CENTER</td>
                  <td>RID CENTER</td>
               </tr>
               <tr>
                  <td>Patna</td>
                  <td>Gaya</td>
                  <td>SASARAM</td>
               </tr>
            </table>
            <tr>
               <td>Chennai</td>
               <td>Delhi</td>
               <td>Mumbai</td>
            </tr>
         </th>
      </tr>
<tr>
   <td>1</td>
   <td>2</td>
   <td>3</td>
 </tr>
</table>
</body>
</html>
```

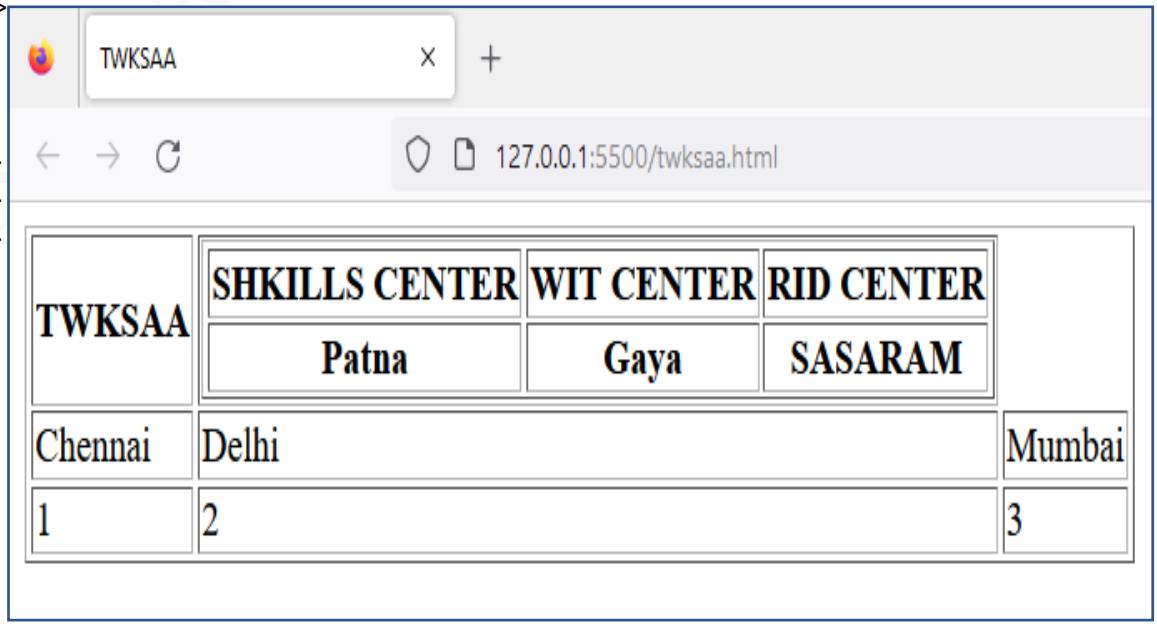


Table-1

HTML Table Example

Student Name		Labs			Avg
First	Last	Lab1	Lab2	Lab3	
Mohan	m	100	100	100	100.0
Rakesh	Raj	100	100	100	100.0
Anj	Raj	100	100	100	100.0

Table-2

Heading	Students	details		
	Id	Name	Department	Roll Number
Student List	1	raj	computer Science	12345
	2	aaj	computer Science	2345
	3	raju	computer Science	345
	4	raja	computer Science	3345

Nested Tables

Table-3

Header Column 1	Header Column 2	Header Column 3	Header Column 4
Row 2 -Item 1	Row 2 -Item 2	Row 2 : Nested Table 1 Row 1 Header item Row 1 Header item Row 1 Header item	Row 2 -Item 4 A second line
Row3: Nested Table Row 1 Header item Row 1 Header item	Row 3 -Item 2		Row 3 -Item 3
Row 4 -Item 1	Row 4 -Item 2	Row 4 -Item 3	
Row 5 -Last Row Of Outer table			

Table-1

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body>
   <h1>HTML Table Example</h1>
   <table border="1">
      <tr>
 <th colspan="2">Student Name</th>
      <th colspan="3">Labs</th>
      <th rowspan="2">Avg</th>
      </tr>
      <tr>
         <th>First </th>
         <th>Last </th>
         <th>Lab1 </th>
         <th>Lab2 </th>
         <th>Lab3 </th>
      </tr>
      <tr>
         <td>Mohan</td>
         <td>m</td>
         <td>100</td>
         <td>100</td>
         <td>100</td>
         <td>100.0</td>
      </tr><tr>
         <td>Rakesj </td>
         <td>Raj</td>
         <td>100</td>
         <td>100</td>
         <td>100</td>
         <td>100.0</td>
      </tr>
      <tr>
         <td>Anj</td>
         <td>Raj</td>
         <td>100</td>
         <td>100</td>
         <td>100</td>
         <td>100.0</td>
      </tr>
      <tr>
         <td rowspan="2">Summary</td>
         <td>Avg</td>
         <td>100.0</td>
         <td>100.0</td>
         <td>100.0</td>
         <td>100.0</td>
      </tr><tr>
         <td>Min</td>
         <td>100.0</td>
         <td>100.0</td>
         <td>100.0</td>
         <td>100.0</td>
      </tr>
   </table>
</body>
</html>
```

Table-2

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>Document</title>
</head>
<body>
   <table border="1">
<tr>
   <th rowspan="2">Heading</th>
   <th>Students</th>
   <th colspan="3">details</th>
</tr>
<tr>
   <th>Id</th>
   <th>Name</th>
   <th>Department</th>
   <th>Roll Number</th>
</tr>
<tr>
<td rowspan="4">Student List</td>
   <td>1</td>
   <td>raj</td>
   <td>computer Science</td>
   <td>12345</td>
</tr>
<tr>
   <td>2</td>
   <td>aaj</td>
   <td>computer Science</td>
   <td>2345</td>
</tr>
<tr><td>3</td>
   <td>raju</td>
   <td>computer Science</td>
   <td>345</td>
</tr>
<tr>
   <td>4</td>
   <td>raja</td>
   <td>computer Science</td>
   <td>3345</td>
</tr>
   </table>
</body>
</html>
```

Table-3

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>Document</title>
</head>
<body>
   <h1>Nested Tables</h1>
   <table border="1">
      <tr>
      <th>Header Column 1</th>
      <th>Header Column 2</th>
      <th>Header Column 3</th>
      <th>Header Column 4</th>
   </tr> <tr>
      <td>Row 2 -Item 1</td>
      <td>Row 2 -Item 2</td>
<td rowspan="2"><p>Row 2 : Nested Table 1</p>
         <table border="1">
            <tr>
               <td>Row 1 Header</td>
               <td>item</td>
               <tr>
               <td>Row 1 Header</td>
               <td>item</td>
            </tr><tr>
               <td>Row 1 Header</td>
               <td>item</td>
            </tr></tr>
         </table>
      </td>
      <td>
      <p>Row 2 -Item 4</p>
      <p> A second line</p>
      </td>
   </tr><tr>
      <td><p>Row3: Nested Table</p>
         <table border="1">
            <tr>
               <td>Row 1 Header</td>
               <td>item</td>
               <tr>
               <td>Row 1 Header</td>
               <td>item</td>
            </tr></tr>
            </table>
      </td>
      <td> Row 3 -Item 2</td>
      <td rowspan="2">Row 3 -Item 3</td>
   </tr>
   <tr>
      <td>Row 4 -Item 1</td>
      <td>Row 4 -Item 2</td>
      <td>Row 4 -Item 3</td>
   </tr>  <tr>
 <td colspan="4">Row 5 -Last Row Of Outer
table</td>
   </tr>
   </table>
</body></html>
```

HTML LISTS:

- HTML Lists are used to specify lists of information. All lists may contain one or more list elements. There are three different types of HTML lists:
  1) Ordered List or Numbered List **(ol)**
  2) Unordered List or Bulleted List **(ul)**
  3) Description List or Definition List **(dl)**

❖ HTML Ordered List or Numbered List:

- ordered HTML lists, all the list items are marked with numbers by default. It is known as numbered list also. The ordered list starts with <ol> tag and the list items start with <li> tag.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>twksaa</title
</head>
<body>
   <ol>
      <li>HTML</li>
      <li>CSS</li>
      <li>JAVASCRIPT</li>
   </ol>
</body>
</html>
```

1. HTML
2. CSS
3. JAVASCRIPT

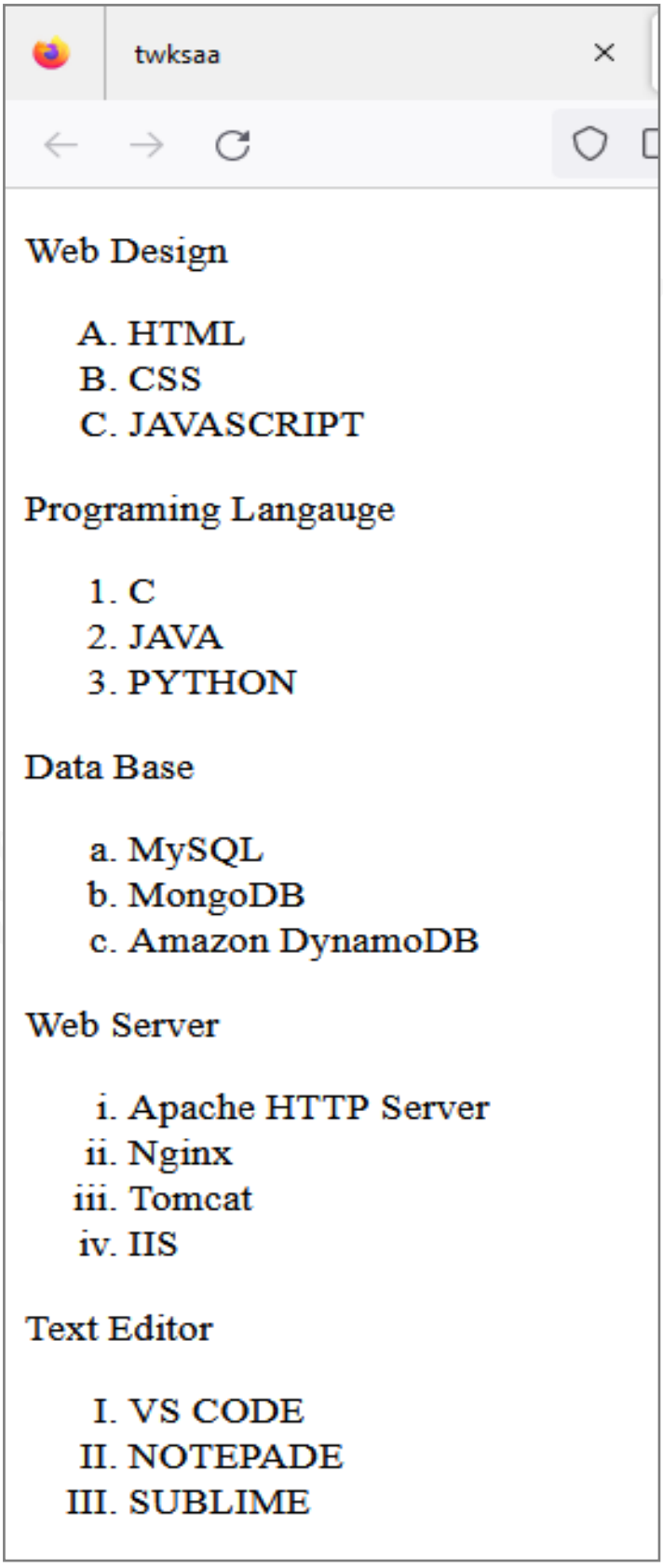
Type: Possible value

type= “1, A, a, i, I”

start= “according your need start the value”

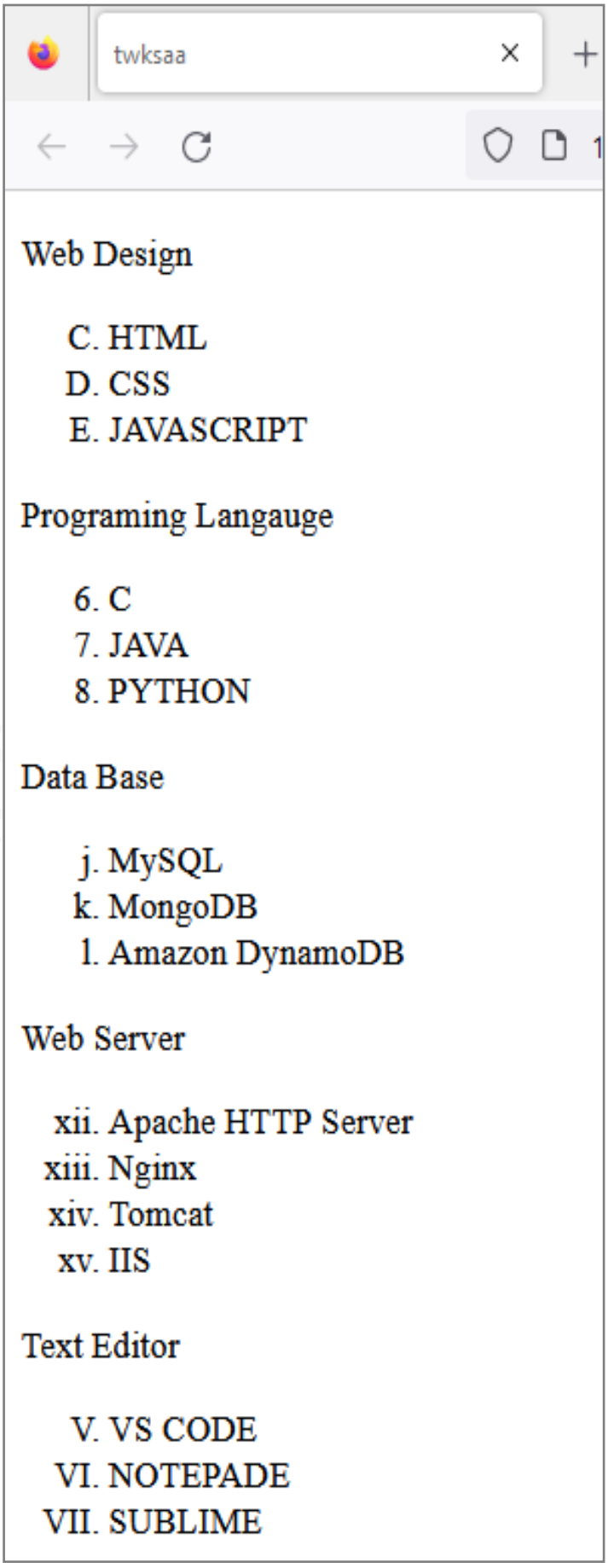
Example-1:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>twksaa</title>
</head>
<body>
   <p>Web Design</p>
   <ol type="A">
      <li>HTML</li>
      <li>CSS</li>
      <li>JAVASCRIPT</li>
   </ol>
   <p>Programming Language</p>
   <ol type="1">
      <li>C</li>
      <li>JAVA</li>
      <li>PYTHON</li>
   </ol>
   <P>Data Base</P>
   <ol type="a">
      <li>MySQL</li>
      <li>MongoDB</li>
      <li>Amazon DynamoDB</li>
   </ol>
   <p>Web Server</p>
   <ol type="i">
      <li>Apache HTTP Server</li>
      <li>Nginx</li>
      <li>Tomcat</li>
      <li>IIS</li>
   </ol>
   <P>Text Editor</P>
   <ol type="I">
      <li>VS CODE</li>
      <li>NOTEPAD</li>
      <li>SUBLIME</li>
   </ol>
</body>
</html>
```



Example-1:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>twksaa</title>
</head>
<body>
   <p>Web Design</p>
   <ol type="A" start="3">
      <li>HTML</li>
      <li>CSS</li>
      <li>JAVASCRIPT</li>
   </ol>
   <p>Programming Language</p>
   <ol type="1" start="6">
      <li>C</li>
      <li>JAVA</li>
      <li>PYTHON</li>
   </ol>
   <P>Data Base</P>
   <ol type="a" start="10">
      <li>MySQL</li>
      <li>MongoDB</li>
      <li>Amazon DynamoDB</li>
   </ol>
   <p>Web Server</p>
   <ol type="i" start="12">
      <li>Apache HTTP Server</li>
      <li>Nginx</li>
      <li>Tomcat</li>
      <li>IIS</li>
   </ol>
   <P>Text Editor</P>
   <ol type="I" start=" 5">
      <li>VS CODE</li>
      <li>NOTEPAD</li>
      <li>SUBLIME</li>
   </ol>
</body>
</html>
```



❖ Unordered List:

- HTML Unordered List or Bulleted List displays elements in bulleted format. We can use unordered list where we do not need to display items in any particular order. The HTML ul tag is used for the unordered list.
- There can be 4 types:

Type	Description
1. **disc**	: This is the default style. list items are marked with disc.
2. **circle**	: The list items are marked with circles.
3. **square**	: The list items are marked with squares.
4. **none**	: the list items are not marked.

Example-1:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>twksaa</title>
</head>
<body>
   <p>Web Design</p>
   <ul>
      <li>HTML</li>
      <li>CSS</li>
      <li>JAVASCRIPT</li>
   </ul>
   <p>Programming Language</p>
   <ul>
      <li>C</li>
      <li>JAVA</li>
      <li>PYTHON</li>
   </ul>
   <P>Data Base</P>
   <ul>
      <li>MySQL</li>
      <li>MongoDB</li>
      <li>Amazon DynamoDB</li>
   </ul>
   <p>Web Server</p>
   <ul>
      <li>Apache HTTP Server</li>
      <li>Nginx</li>
      <li>Tomcat</li>
      <li>IIS</li>
   </ul>
</body>
</html>
```

Example-2:

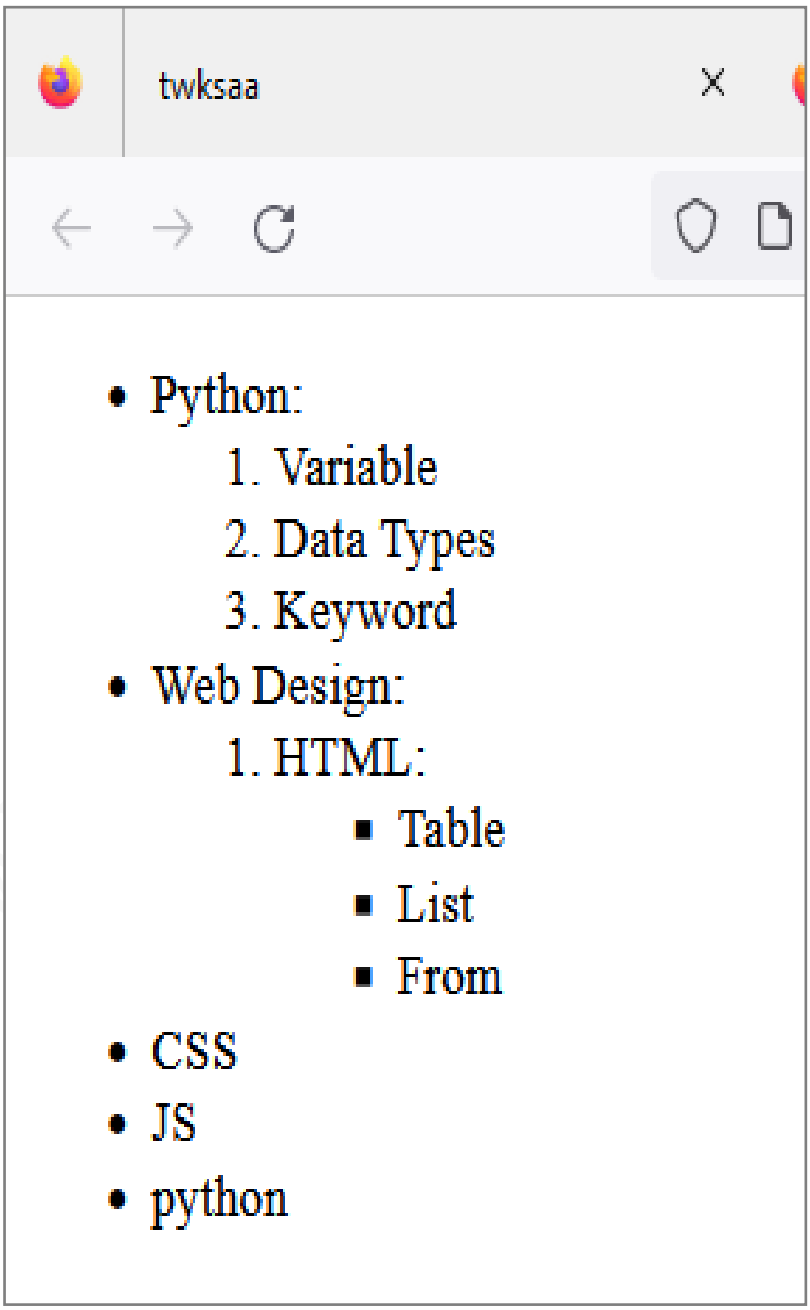
```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>twksaa</title>
</head>
<body>
   <p>Web Design</p>
   <ul type="circle">
      <li>HTML</li>
      <li>CSS</li>
      <li>JAVASCRIPT</li>
   </ul>
   <p>Programming Language</p>
   <ul type="square" >
      <li>C</li>
      <li>JAVA</li>
      <li>PYTHON</li>
   </ul>
   <P>Data Base</P>
   <ul type="disc">
      <li>MySQL</li>
      <li>MongoDB</li>
      <li>Amazon DynamoDB</li>
   </ul>
   <p>Web Server</p>
   <ul type="none">
      <li>Apache HTTP Server</li>
      <li>Nginx</li>
      <li>Tomcat</li>
      <li>IIS</li>
   </ul>
</body>
</html>
```

❖ **Nested list:**

➢ Means list inside list:

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>Document</title>
</head>
<body>
   <ul>
      <li>Python:
         <ol>
            <li>Variable</li>
            <li>Data Types</li>
            <li>Keyword</li>
         </ol>
      </li>
      <li>Web Design:
         <ol>
            <li>HTML:
               <ul>
                  <li>Table</li>
                  <li>List</li>
                  <li>From</li>
               </ul>
            </li>
         </ol>
         <li>CSS</li>
         <li>JS</li>
         <li>python</li>
      </li>
   </ul>
</body>
</html>
```



❖ HTML Description List:

- HTML Description List or Definition List displays elements in definition form like in dictionary. The <dl>, <dt> and <dd> tags are used to define description list.
- The 3 HTML description list tags are given below:

- **<dl>** tag defines the description list.
- **<dt>** tag defines data term.
- **<dd>** tag defines data definition (description).

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Document</title>
</head>
<body>
  <dl>
    <dt>HTML:</dt>
    <dd>is a markup language</dd>
    <dt>Java:</dt>
    <dd>is a programming language and platform</dd>
   <dt>JavaScript:</dt>
   <dd>is a scripting language</dd>
    <dt>SQL:</dt>
    <dd>is a query language</dd>
  </dl>
</body>
</html>
```

HTML:

is a markup language

Java:

is a programming language and platform

JavaScript:

is a scripting language

SQL:

is a query language

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>TWKSAA</title>
</head>
<body>
  <h3>Preceding Text</h3>
  <ol type="I">
    <li>List Item 1
      <ol type="a">
        <li>Nested Item 1.1</li>
        <li>Nested Item 1.2</li>
      </ol>
      </li>
      <li>List Item2
        <ol type="1">
        <li>Nested Item 2.1</li>
        <li>Nested Item 2.2
          <ul type="circle">
          <li>Nested Item 2.2.1</li>
          <li>Nested Item 2.2.2
            <ul type="square">
              <li>Nested Item 2.2.2.1</li>
              <li>Nested Item 2.2.2.2</li>
            </ul>
          </li>
          </ul>
          <li>Nested Item 2.2.3</li>
        </li>
        </ol>
      </li>
      <li>List Item 3
        <ul type="disc">
        <li>Nested Item 3.1</li>
         <li>Nested Item 3.2</li>
         <li>Nested Item 3.3</li>
        </ul>
      </li>
  </ol>
</body>
</html>
```

Preceding Text

I. List Item 1
   a. Nested Item 1.1
   b. Nested Item 1.2

II. List Item2
   1. Nested Item 2.1
   2. Nested Item 2.2
      - Nested Item 2.2.1
      - Nested Item 2.2.2
        - Nested Item 2.2.2.1
        - Nested Item 2.2.2.2
   3. Nested Item 2.2.3

III. List Item 3
   - Nested Item 3.1
   - Nested Item 3.2
   - Nested Item 3.3

Example:

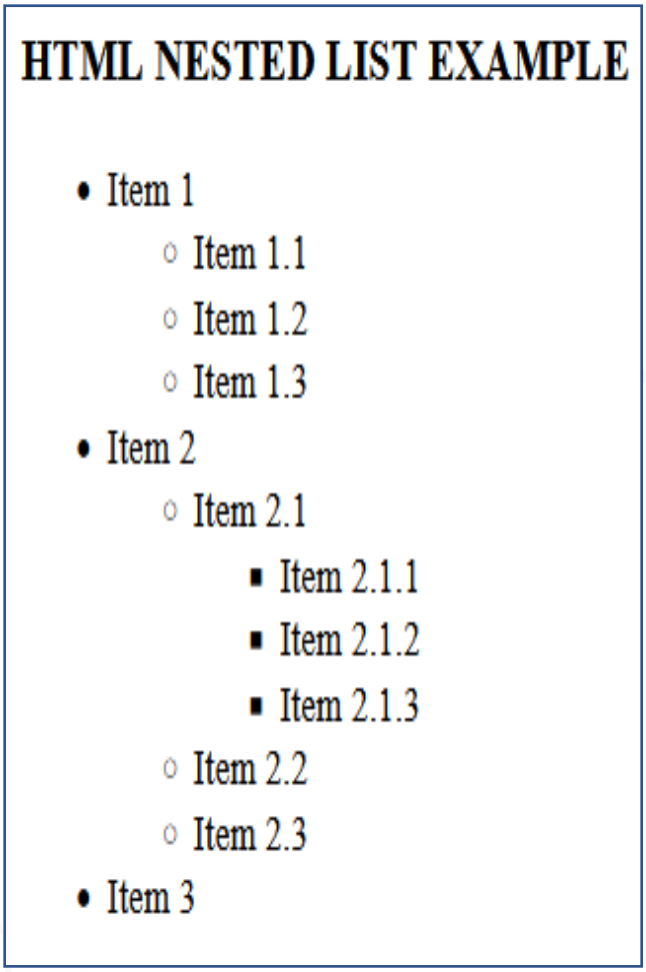
```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>TWKSAA</title>
</head>
<body>
   <h1 style="font-weight:
bold;">Nested Ordered Lists</h1>
   <ol type="I">
      <li>Headings</li>
      <li>Basic Text Sectoons</li>
      <li>Lists
         <ol type="A">
            <li>Ordered</li>
            <ol type="1">
               <li> The OL tag
                  <ol type="a">
                     <li>TYPE</li>
                     <li>START</li>
                     <li>COMPACT</li>
                  </ol>
               </li>
               <li>The LI tag</li>
            </ol>
            <li>Unordered
               <ol type="1">
                  <li>The UL tag</li>
                  <li>The LI tag</li>
               </ol>
            </li>
            <li>Definition
               <ol type="1">
                  <li>The DL tag</li>
                  <li>The DT tag</li>
                  <li>The DD tag</li>
               </ol>
            </li>
         </ol>
      </li>
      <li>Miscellaneous</li>
   </ol>
</body>
</html>
```

Nested Ordered Lists

I. Headings
II. Basic Text Sectoons
III. Lists
    A. Ordered
        1. The OL tag
            a. TYPE
            b. START
            c. COMPACT
        2. The LI tag
    B. Unordered
        1. The UL tag
        2. The LI tag
    C. Definition
        1. The DL tag
        2. The DT tag
        3. The DD tag
IV. Miscellaneous

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>TWKSAA</title>
</head>
<body>
  <h3>HTML NESTED LIST
EXAMPLE</h3>
  <ul type="disc">
    <li>Item 1
      <ul type="circle">
        <li>Item 1.1</li>
        <li>Item 1.2</li>
        <li>Item 1.3</li>
      </ul>
    </li>
    <li>Item 2
      <ul type="circle">
        <li>Item 2.1
          <ul type="square">
            <li>Item 2.1.1</li>
            <li>Item 2.1.2</li>
            <li>Item 2.1.3</li>
          </ul>
        </li>
        <li>Item 2.2</li>
        <li>Item 2.3</li>
      </ul>
    </li>
    <li>Item 3</li>
  </ul>
</body>
</html>
```



Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Document</title>
</head>
<body>
  <h1>List Example</h1>
  <ol type="I">
    <li>List Item 1
      <ol type="a">
        <li>Nested Item 1.1</li>
        <li>Nested Item 1.2</li>
      </ol>
    </li>
    <li>List Item 2
      <ol type="1">
        <li>Nested Item 2.1</li>
        <li>Nested Item 2.2
          <ul type="circle">
            <li>Nested Item 2.2.1</li>
            <li>Nested Item 2.2.2
            <ul type="square">
              <li>Nested Item 2.2.2.1</li>
              <li>Nested Item 2.2.2.2</li>
            </ul>
            </li>
            <li>Nested Item 2.2.3</li>
          </ul>
        </li>
        <li>Nested Item 2.3</li>
      </ol>
    </li>
    <li>List Item 3
      <ul type="disc">
        <li>Nested Item 3.1</li>
        <li>Nested Item 3.2</li>
        <li>Nested Item 3.3</li>
      </ul> </li>
  </ol>
  <dl>
    <dt>COMP 249</dt>
    <dd>Object-Oriented Programming ll.</dd>
    <dt>Soen287</dt>
    <dd>Web Programming. </dd>
  </dl>
</body>
</html>
```

List Example

I. List Item 1
- a. Nested Item 1.1
- b. Nested Item 1.2

II. List Item 2
- 1. Nested Item 2.1
- 2. Nested Item 2.2
  - ○ Nested Item 2.2.1
  - ○ Nested Item 2.2.2
    - ▪ Nested Item 2.2.2.1
    - ▪ Nested Item 2.2.2.2
  - ○ Nested Item 2.2.3
- 3. Nested Item 2.3

III. List Item 3
- • Nested Item 3.1
- • Nested Item 3.2
- • Nested Item 3.3

COMP 249
: Object-Oriented Programming ll.

Soen287
: Web Programming.

HTML FORM

- HTML form is a section of a document which contains controls such as text fields, password fields, checkboxes, radio buttons, submit button, menus etc.
- <form> tag is used to create a form on a web page.
- HTML form facilitates the user to enter data that is to be sent to the server for processing such as name, email address, password, phone number, etc.

❖ Why use HTML Form?

- HTML forms are used to collect user input, submit data to web servers, enable user interaction, authenticate users, and enhance the overall user experience on websites.

Syntax:

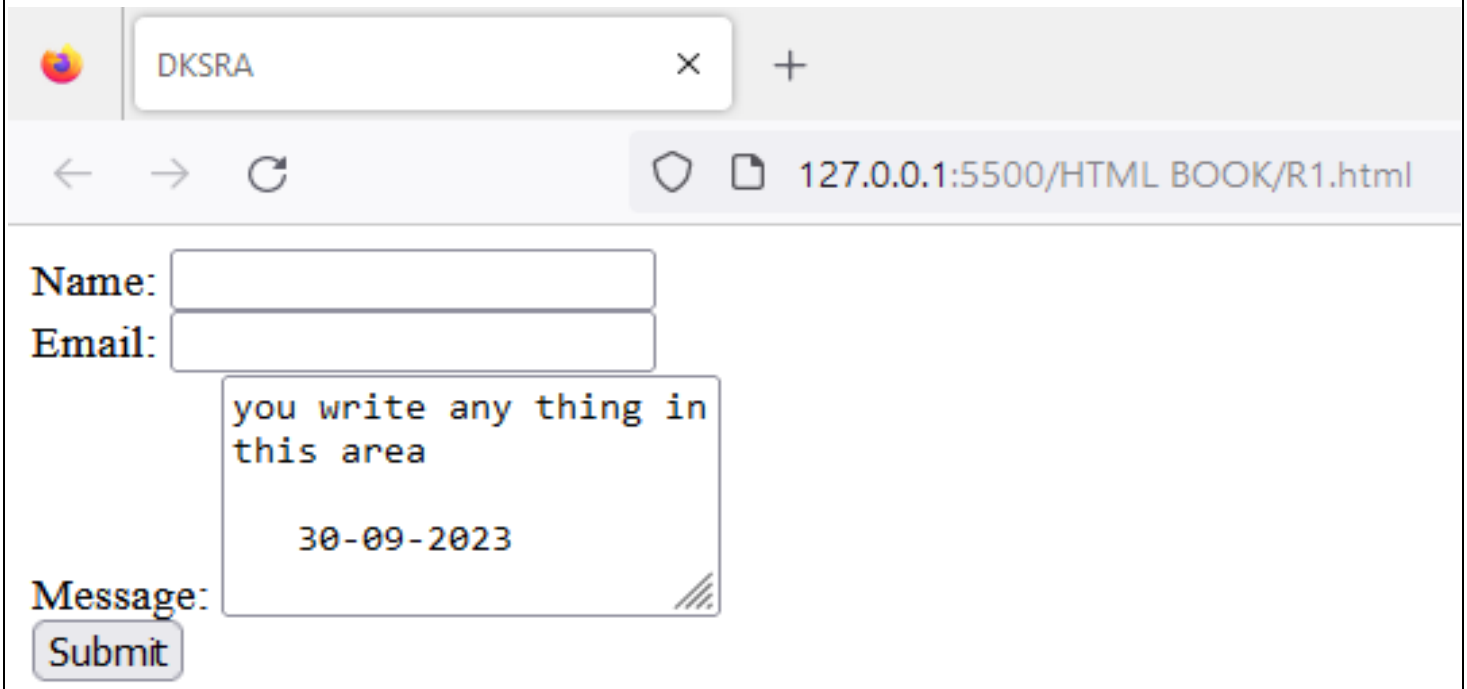
```
<form action="URL_to_handle_form_submission" method="HTTP_method">
  <!-- Form elements go here -->
</form>
```

- **<form>:** This is the opening tag that defines the start of the form.
- **action:** Specifies the URL (Uniform Resource Locator) where the form data will be sent when the user submits it. This URL typically points to a server-side script or endpoint that processes the form data.
- **method:** Specifies the HTTP method to be used when submitting the form data. The two most common methods are "GET" and "POST." "GET" appends the form data to the URL, while "POST" sends it in the request body.
- **Form elements (e.g., input fields, buttons, checkboxes, etc.):** These elements go inside the <form> tags and define the fields and controls that users interact with to input data.
- **<input>, <textarea>, <select>:** These are examples of form elements that collect different types of user input.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>DKSRA</title>
</head>
<body>
  <form action="process_form.js" method="post">
    <label for="name">Name:</label>
    <input type="text" id="name" name="name" required><br>
    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required><br>
    <label for="message">Message:</label>
    <textarea id="message" name="message" rows="4" required></textarea><br>
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

Output:



❖ **List of HTML 5 form tags:**

1) **<input>:** Allows user input, with various types like text, password, email, and more.
2) **<textarea>:** Provides a multiline text input area.
3) **<select>:** Creates a dropdown list for user selection.
4) **<label>:** Provides a text label for form elements, improving accessibility.
5) **<fieldset>:** Groups related form elements and adds a border for visual grouping.
6) **<legend>:** Specifies a title or description for a <fieldset>.
7) **<button>:** Represents a clickable button for form submission or actions.
8) **<form>:** Container for all form elements, defining submission behavior.
9) **<datalist>:** Provides a list of predefined options for autocompletion.
10) **<output>:** Displays the result of a calculation or form submission.
11) **<progress>:** Shows the progress of a task, like file uploads.
12) **<meter>:** Displays a scalar measurement within a known range.
13) **<keygen>:** Generates key pairs for cryptography, mainly for secure data transmission.

1. <input>:

Syntax: <input type="input_type" name="input_name" value="default_value" id="input_id" class="input_class" required>

Explanation:

- **type:** Specifies the type of input. It determines the kind of data the input field can accept (e.g., text, password, email, radio, checkbox, etc.).
- **name:** Defines the name of the input field. This name is used when submitting the form data to the server. Each input field within a form should have a unique name.
- **value:** Specifies a default value for the input field. This value is optional and can be pre-filled in the input field.
- **id:** Provides a unique identifier for the input element. This ID can be used for JavaScript interactions or for associating the input with a <label> element.
- **class:** Assigns one or more CSS classes to the input element
- **required:** An optional attribute that, when present, specifies that the input field must be filled out by the user before submitting the form.

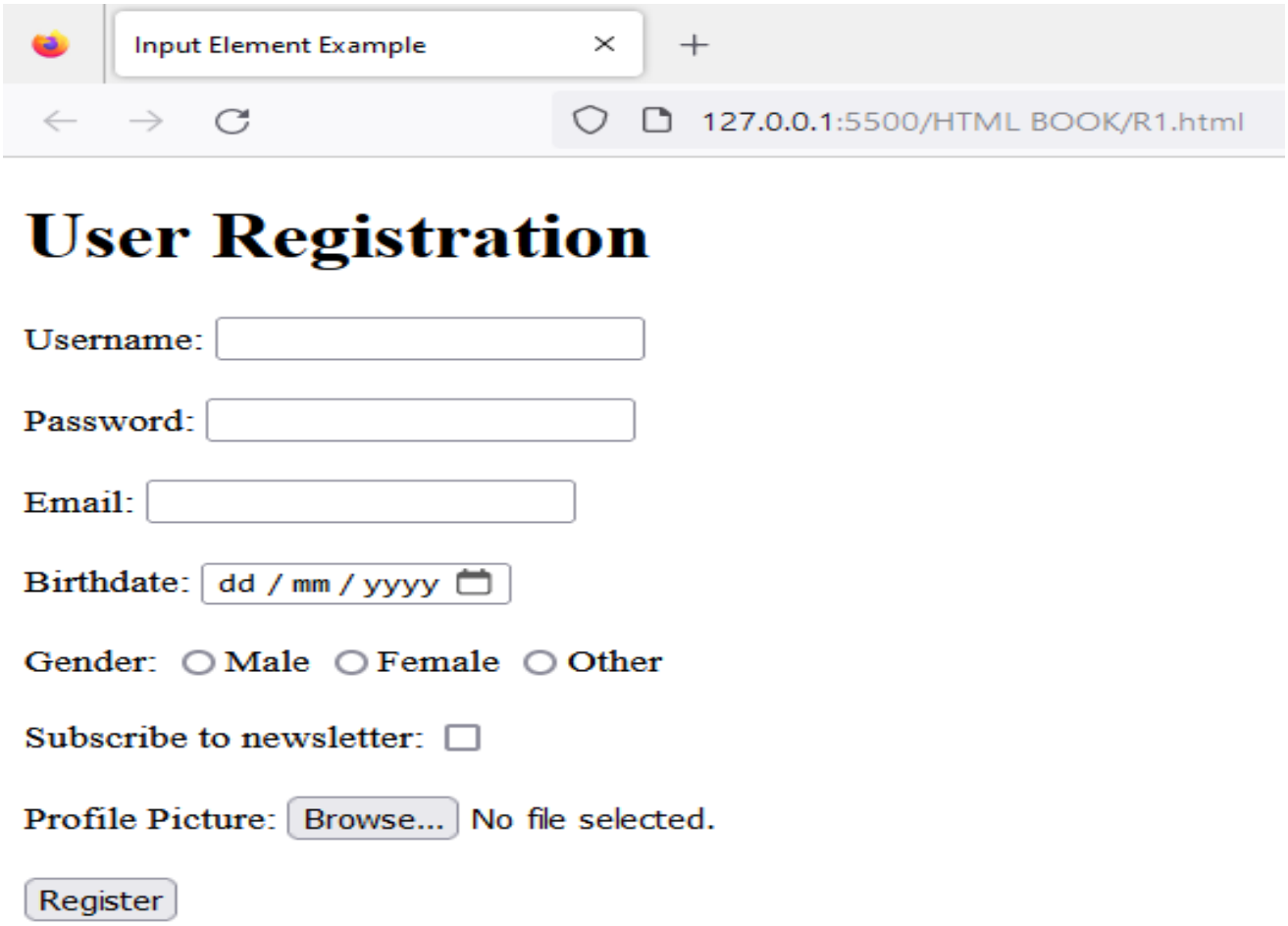
Example-1:

- <input type="text" name="username" id="username" class="input-field" required>

Example-2

```
<!DOCTYPE html>
<html>
<head>
  <title>Input Element Example</title>
</head>
<body>
  <h1>User Registration</h1>
  <form action="process_form.js" method="post">
    <label for="username">Username:</label>
    <input type="text" id="username" name="username" required><br><br>
    <label for="password">Password:</label>
    <input type="password" id="password" name="password" required><br><br>
    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required><br><br>
    <label for="birthdate">Birthdate:</label>
    <input type="date" id="birthdate" name="birthdate"><br><br>
    <label for="gender">Gender:</label>
    <input type="radio" id="male" name="gender" value="male">Male
    <input type="radio" id="female" name="gender" value="female">Female
    <input type="radio" id="other" name="gender" value="other">Other<br><br>
    <label for="subscribe">Subscribe to newsletter:</label>
    <input type="checkbox" id="subscribe" name="subscribe" value="yes"><br><br>
    <label for="file">Profile Picture:</label>
    <input type="file" id="file" name="file"><br><br>
    <input type="submit" value="Register">
  </form>
</body>
</html>
```

Output:



2. <textarea>:

Syntax:

```
<textarea id="textarea_id" name="textarea_name" rows="number_of_rows"
cols="number_of_columns">Initial text</textarea>
```

Explanation:

- **id:** Provides a unique identifier for the <textarea> element. This ID can be used for JavaScript interactions or for associating the <textarea> with a <label> element.
- **name:** Defines the name of the <textarea> element. This name is used when submitting the form data to the server. Each <textarea> element within a form should have a unique name.
- **rows:** Specifies the number of visible text lines (rows) that should be initially displayed in the <textarea>. This attribute is optional but helps set the initial size of the input area.
- **cols:** Specifies the number of visible characters (columns) that should be initially displayed in the <textarea>. This attribute is also optional but helps set the initial width of the input area.
- The content between the opening and closing <textarea> tags (e.g., "Initial text" in the example) represents the initial text that appears in the <textarea>. Users can edit or replace this text.

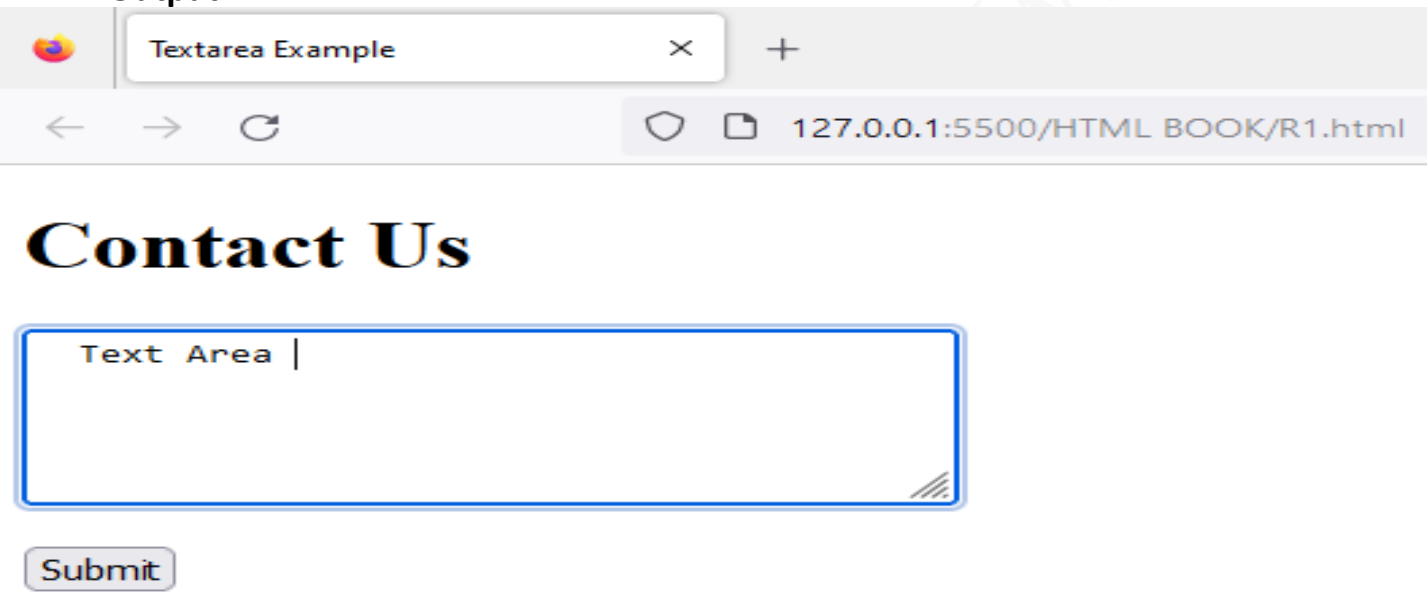
Example-1:

- <textarea id="message" name="message" rows="4" cols="40">Enter your message here...</textarea>

Example-2:

```
<!DOCTYPE html>
<html>
<head>
  <title>Textarea Example</title>
</head>
<body>
  <h1>Contact Us</h1>
  <form action="process_form.js" method="post">
    <textarea id="message" name="message" rows="4" cols="40" required></textarea><br><br
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

Output:



3. <select>

Syntax:

```
<select id="select_id" name="select_name">
  <option value="option_value1">Option 1</option>
  <option value="option_value2">Option 2</option>
  <option value="option_value3">Option 3</option>
  <!-- Add more option elements as needed -->
</select>
```

Explanation:

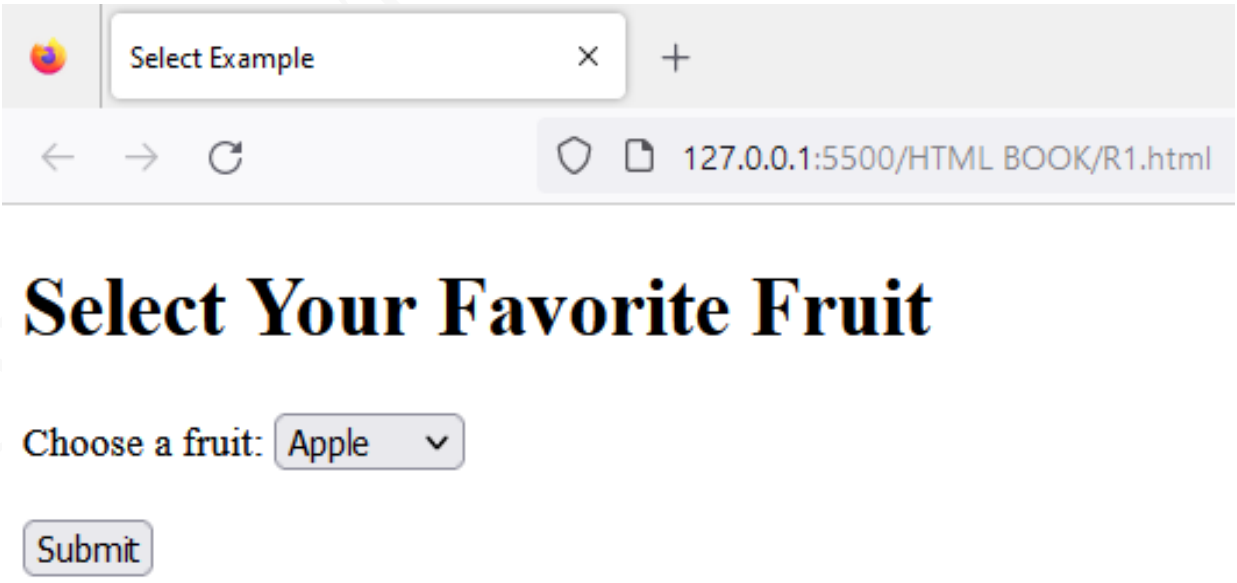
- **<select>:** The opening tag for the <select> element.
- **id:** An optional attribute providing a unique identifier for the <select> element, which can be used for styling or JavaScript interactions.
- **name:** Specifies the name of the form control, which is used when submitting the form data to the server.
- **<option>:** The tag used for each individual option within the dropdown.

- **value:** Specifies the value that will be sent to the server when the user selects a particular option. It's what the server processes.
- Text content between the <option> tags (e.g., "Option 1") is the visible text that the user sees in the dropdown.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Select Example</title>
</head>
<body>
  <h1>Select Your Favorite Fruit</h1>
  <form action="process_form.php" method="post">
    <label for="fruit">Choose a fruit:</label>
    <select id="fruit" name="fruit">
      <option value="apple">Apple</option>
      <option value="banana">Banana</option>
      <option value="cherry">Cherry</option>
      <option value="grape">Grape</option>
      <option value="orange">Orange</option>
    </select>
    <br><br>
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

Output:



4. <label>:

Syntax: <label for="input_id">Label Text:</label>

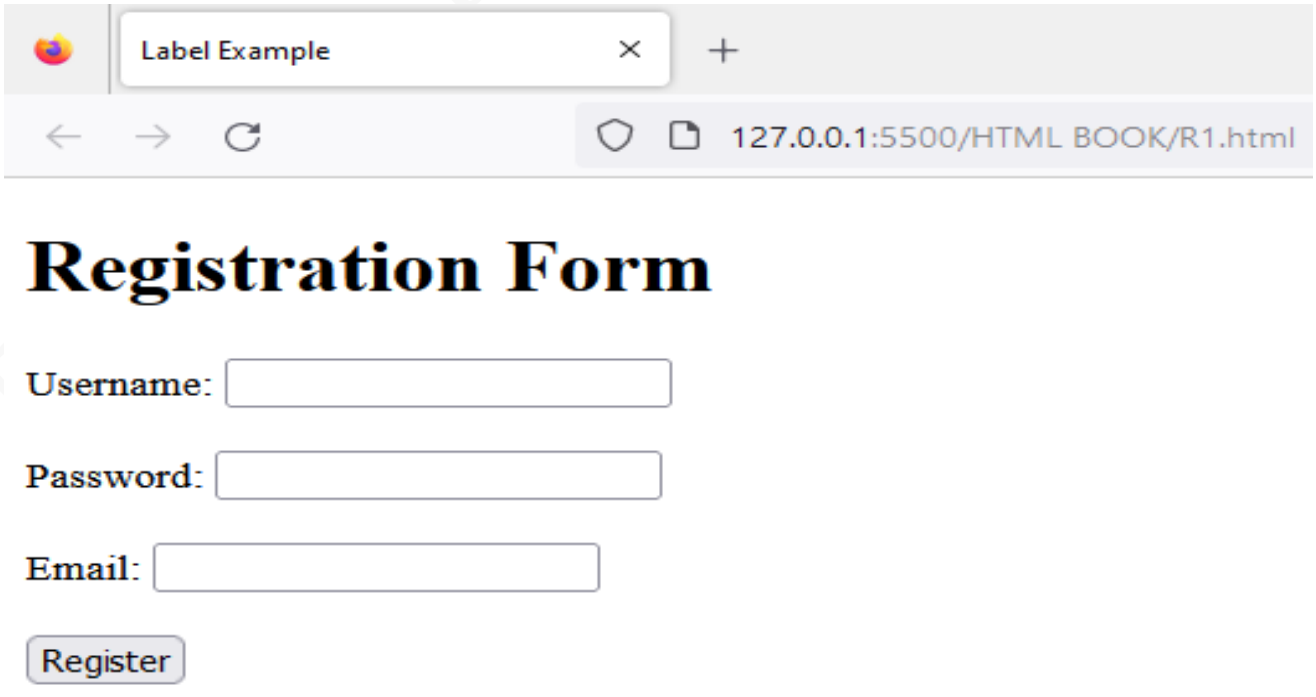
Explanation:

- **<label>:** The opening and closing tags for the <label> element.
- **for:** The for attribute is used to associate the <label> with a specific form control, identified by its id. This association improves accessibility and allows users to click on the label to interact with the associated form element.
- **Label Text:** The text you want to display as the label.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Label Example</title>
</head>
<body>
  <h1>Registration Form</h1>
  <form action="process_form.php" method="post">
    <label for="username">Username:</label>
    <input type="text" id="username" name="username" required><br><br>
    <label for="password">Password:</label>
    <input type="password" id="password" name="password" required><br><br>
    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required><br><br>
    <input type="submit" value="Register">
  </form>
</body></html>
```

Output:



5. <fieldset>:

Syntax:

```
<fieldset>
  <legend>Fieldset Title or Description</legend>
  <!-- Form elements go here -->
</fieldset>
```

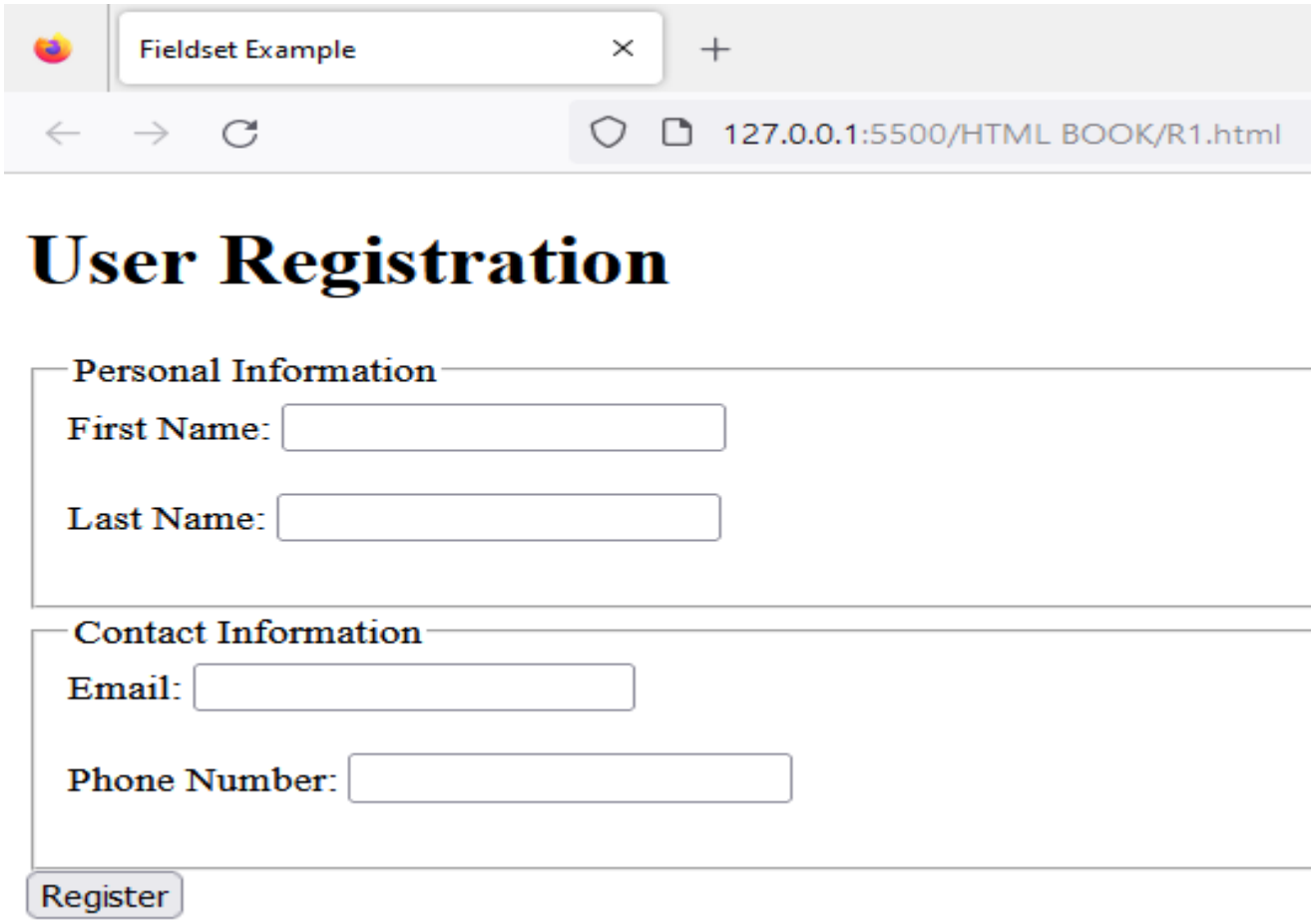
Explanation:

- **<fieldset>:** The opening and closing tags for the <fieldset> element, which defines a group of related form elements.
- **<legend>:** The <legend> element is used to specify a title or description for the group of form elements enclosed within the <fieldset>. It provides context and helps with accessibility.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Fieldset Example</title>
</head>
<body>
  <h1>User Registration</h1>
  <form action="process_form.php" method="post">
    <fieldset>
      <legend>Personal Information</legend>
      <label for="first_name">First Name:</label>
      <input type="text" id="first_name" name="first_name"><br><br>
      <label for="last_name">Last Name:</label>
      <input type="text" id="last_name" name="last_name"><br><br>
    </fieldset>
    <fieldset>
      <legend>Contact Information</legend>
      <label for="email">Email:</label>
      <input type="email" id="email" name="email"><br><br>
      <label for="phone">Phone Number:</label>
      <input type="tel" id="phone" name="phone"><br><br>
    </fieldset>
    <input type="submit" value="Register">
  </form>
</body>
</html>
```

Output:



6. <legend>:

Syntax:

```
<fieldset>
  <legend>Your Legend Text Here</legend>
  <!-- Form controls go here -->
</fieldset>
```

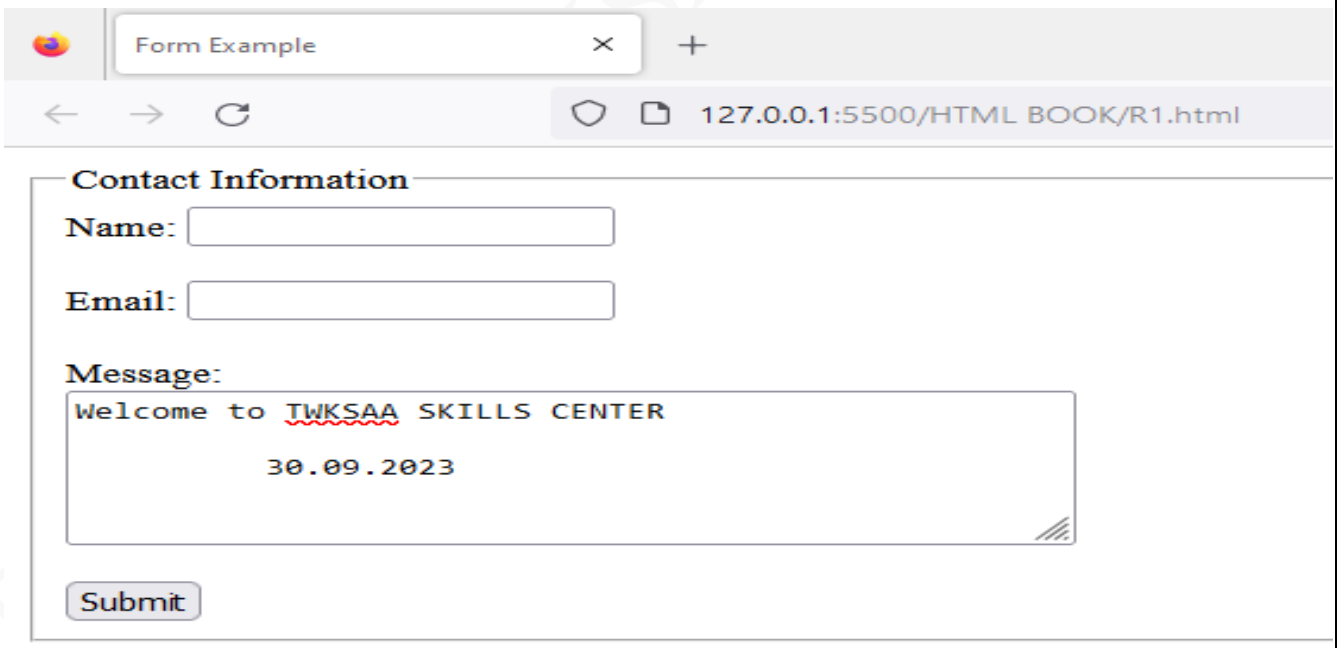
Explanation:

- **<fieldset>:** This is an HTML element used to group related form controls, such as input fields and labels. It creates a visual and semantic separation for these controls.
- **<legend>:** This element is placed inside the <fieldset> and provides a descriptive label or caption for the group of form controls enclosed within that <fieldset>. It helps users understand the purpose of the grouped controls.
- **Your Legend Text Here:** This is where you would replace the text with your own descriptive label or caption that explains what the enclosed form controls are for. For example, if you have a group of form controls related to "Contact Information," you would put "Contact Information" as the content of <legend>.
- **<!-- Form controls go here -->:** This is a comment indicating that you should place your actual form controls, such as input fields, labels, buttons, etc., within the <fieldset>. These controls are related to the caption provided by the <legend> element.

Example:

```
<!DOCTYPE html>
<html>
<head>
 <title>Form Example</title>
</head>
<body>
<form>
 <fieldset>
  <legend>Contact Information</legend>
  <label for="name">Name:</label>
  <input type="text" id="name" name="name"><br><br>
  <label for="email">Email:</label>
  <input type="email" id="email" name="email"><br><br>
  <label for="message">Message:</label><br>
  <textarea id="message" name="message" rows="4" cols="50"></textarea><br><br>
  <input type="submit" value="Submit">
 </fieldset>
</form>
</body>
</html>
```

Output:



7. <button>:

Syntax:

- <button type="button" id="yourButtonID">Button Text</button>

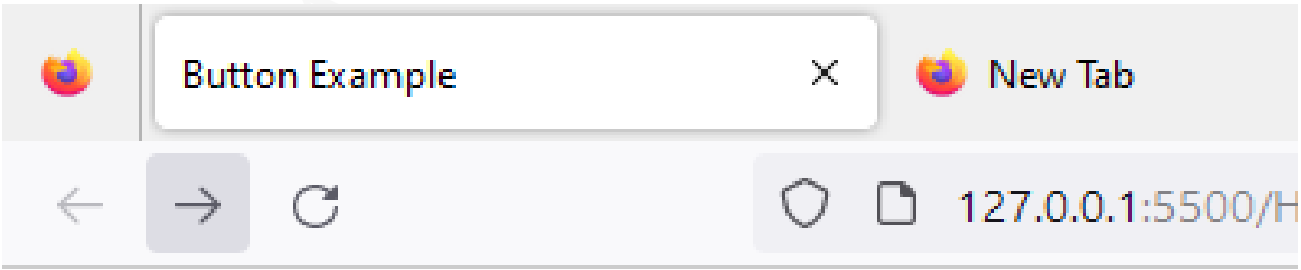
Explanation:

- **<button>:** This is the opening tag of the button element.
- **type="button":** The type attribute specifies the type of the button. In this case, it's set to "button," which creates a regular clickable button. The "button" type is the default, so you can omit this attribute if you don't need to specify a different button type.
- **id="yourButtonID":** The id attribute is used to give the button a unique identifier. This can be useful if you want to apply CSS styles or JavaScript functionality to the button, or if you need to reference the button in your code.
- **Button Text:** This is the text that will appear on the button. Replace it with the actual text you want to display on the button.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Button Example</title>
</head>
<body>
<h1>Click the Button</h1>
<!-- A button element within a form -->
<form action="https://example.com/submit" method="post">
  <button type="submit">Submit Form</button>
</form>
</body>
</html>
```

Output:



Click the Button

Submit Form

8. <form>:

Syntax:

```
<form action="URL" method="HTTP_METHOD">
  <!-- Form controls go here -->
  <input type="text" name="input_name" id="input_id" value="default_value">
  <!-- More form controls -->
  <button type="submit">Submit</button>
</form>
```

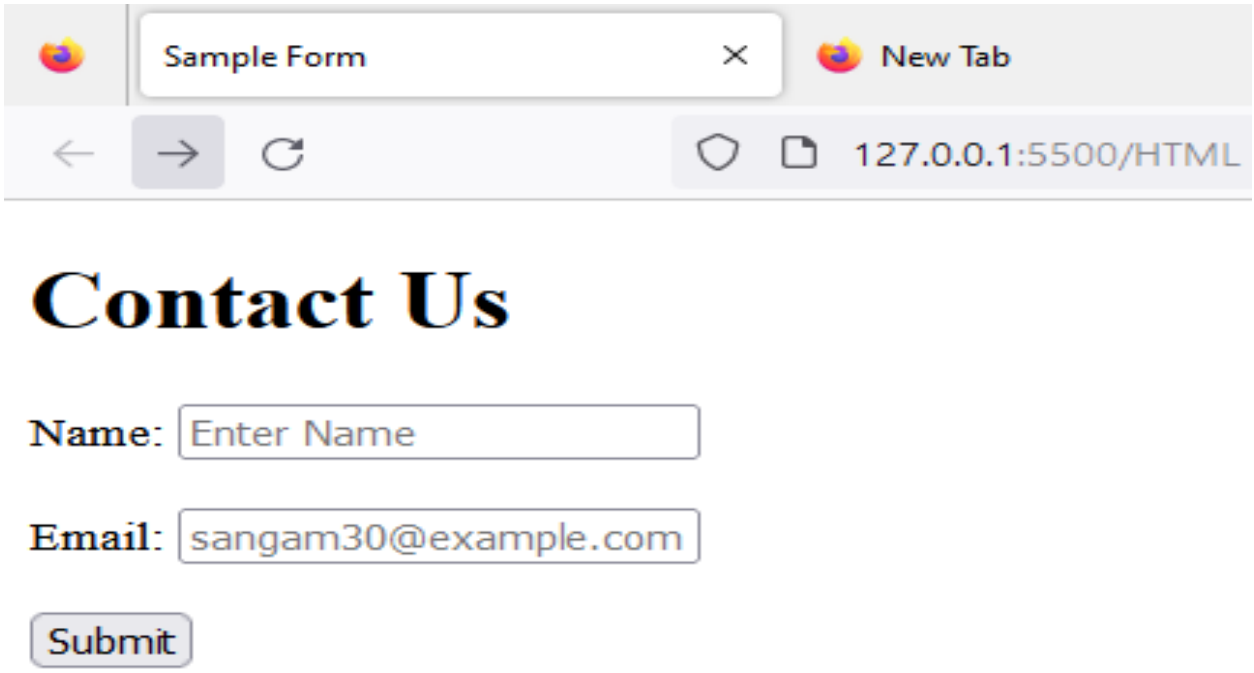
Explanation:

- **<form>:** This is the opening tag of the form element.
- **action="URL":** The action attribute specifies the URL to which the form data should be submitted when the user clicks the "Submit" button. This URL can be a server-side script.
- **method="HTTP_METHOD":** method attribute specifies the HTTP method to be used when submitting the form data. Common values for this attribute are "GET" and "POST." "GET" appends the data to the URL, while "POST" sends the data in the body of the HTTP request.
- **<input>, <select>, <textarea>:** These are form control elements that allow users to input data. You can have various types of form controls within the <form> element, such as text fields, radio buttons, checkboxes, dropdown lists, and more.
- **name="input_name":** The name attribute is used to identify the input element when the form is submitted. It's important for server-side processing.
- **id="input_id":** The id attribute provides a unique identifier for the input element. It's useful for associating labels with form controls and for scripting purposes.
- **value="default_value":** The value attribute sets the default value for the input element. This is optional and can be used to pre-fill form fields with initial values.
- **<button type="submit">Submit</button>:** This is a button element that, when clicked, submits the form data to the URL specified in the action attribute. The type="submit" attribute indicates that this button is used to submit the form.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Sample Form</title>
</head>
<body>
<h1>Contact Us</h1>
<form action="https://example.com/submit" method="POST">
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" placeholder="Enter Name"><br><br>
  <label for="email">Email:</label>
  <input type="email" id="email" name="email" placeholder="Sangam30@example.com"><br><br>
  <button type="submit">Submit</button>
</form>
</body>
</html>
```

Output:



9. <datalist>:

Syntax:

```
<input list="datalist_id">
<datalist id="datalist_id">
  <option value="Option 1">
  <option value="Option 2">
  <!-- Add more options here -->
</datalist>
```

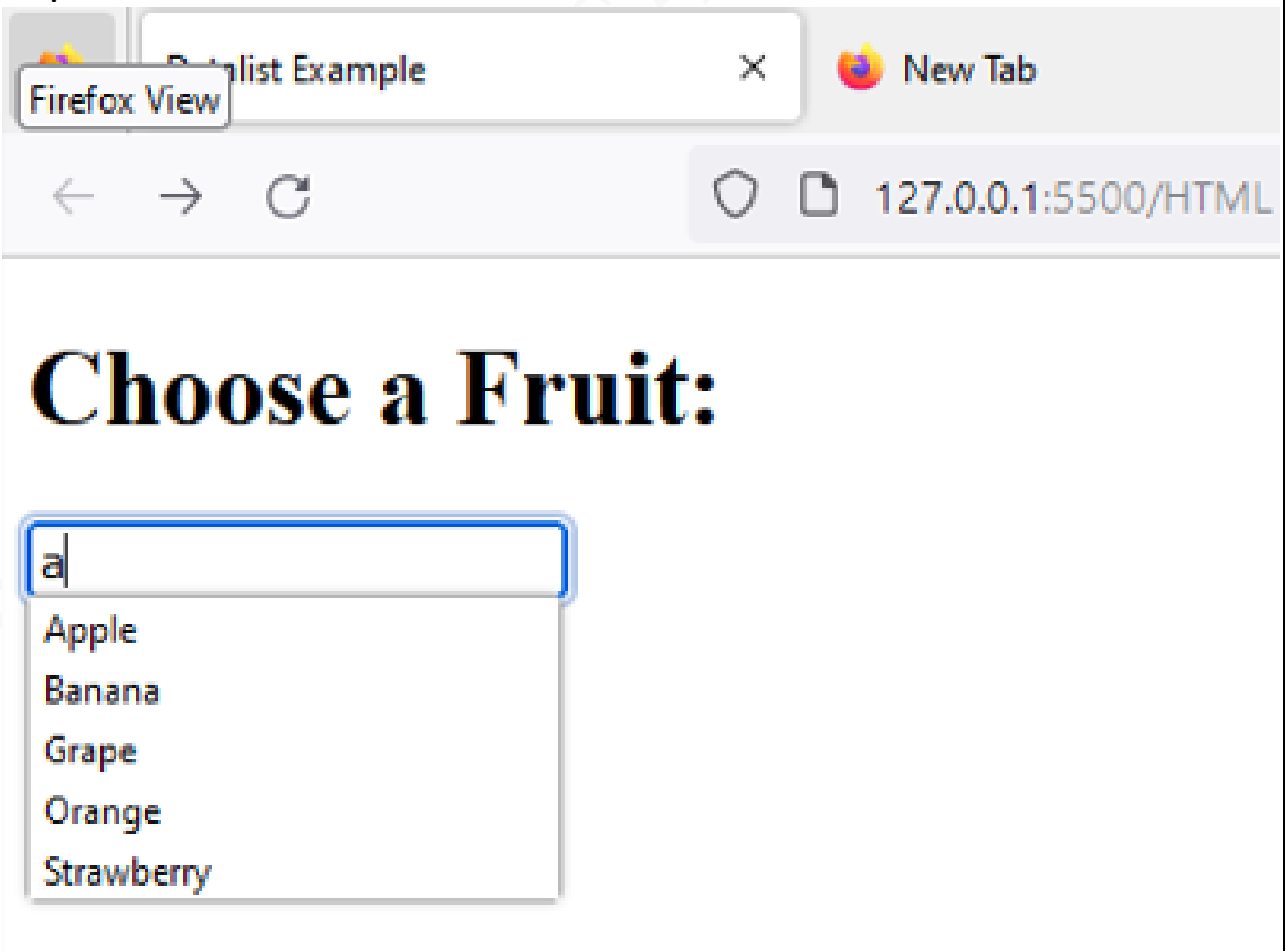
Explanation:

- **<input>:** This is the input element to which you want to associate the <datalist>. It should have a list attribute with the value set to the id of the associated <datalist>.
- **list="datalist_id":** The list attribute of the <input> element specifies the ID of the <datalist> element that provides the predefined options.
- **<datalist>:** This is the opening tag of the datalist element.
- **id="datalist_id":** The id attribute of the <datalist> element serves as a unique identifier for this particular datalist.
- **<option>:** These are the individual options within the <datalist>. Each <option> element represents a choice that the user can select when they interact with the associated <input> element.
- **value="Option 1":** The value attribute of each <option> element specifies the text that will be displayed as a choice in the dropdown list.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Datalist Example</title>
</head>
<body>
<h1>Choose a Fruit:</h1>
<!-- An input element with a list attribute -->
<input type="text" list="fruits" id="fruitInput" placeholder="Start typing...">
<!-- The associated datalist with predefined options -->
<datalist id="fruits">
  <option value="Apple">
  <option value="Banana">
  <option value="Cherry">
  <option value="Grape">
  <option value="Orange">
  <option value="Strawberry">
</datalist>
</body>
</html>
```

Output:



10. <output>:

Syntax: <output for="input_id">Output Text</output>

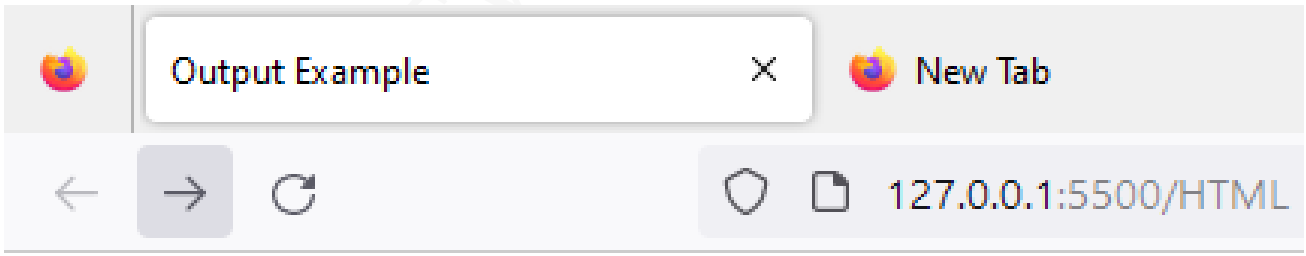
Explanation:

- <output>: This is the opening tag of the output element.
- for="input_id": The for attribute specifies the id of the form element (typically an <input>) whose value the output should be associated with. It links the output to the input, indicating that the displayed content is the result or output of the input element.
- Output Text: This is the initial text or content that will be displayed within the <output> element. It can be static text or serve as a placeholder.

Example:

```
<!DOCTYPE html>
<html>
<head>
 <title>Output Example</title>
</head>
<body>
<h1>Displaying Static Output:</h1>
<!-- An input element for user input -->
<label for="name">Enter your name:</label>
<input type="text" id="name">
<br><br>
<!-- The output element with static content -->
<output for="name">Hello, <span id="outputText">John Doe</span>!</output>
</body>
</html>
```

Output:



Displaying Static Output:

Enter your name: Sangam Kumar

Hello, Sangam Kumar!

11. <progress>:

Syntax:

<progress value="current_value" max="max_value">Fallback Content</progress>

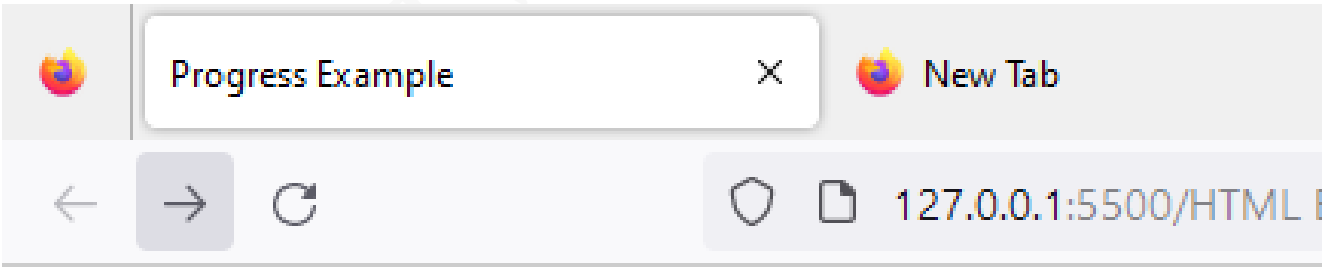
Explanation:

- **<progress>:** This is the opening tag of the progress element.
- **value="current_value":** The value attribute specifies the current value of the progress, indicating how much of the task has been completed. It should be a number between 0 and the max value.
- **max="max_value":** The max attribute sets the maximum value that the progress can reach. This represents the completion point of the task.
- **Fallback Content:** This is the content that will be displayed if the browser does not support the <progress> element. It's a good practice to provide fallback content for compatibility.

Example:

```
<!DOCTYPE html>
<html>
<head>
 <title>Progress Example</title>
</head>
<body>
<h1>File Upload Progress</h1>
<!-- A progress bar representing the file upload progress -->
<progress value="40" max="100">40% Complete</progress>
<p>Uploading... Please wait.</p>
</body>
</html>
```

Output:



File Upload Progress

Uploading... Please wait.

12. <meter>:

Syntax:

<meter min="min_value" max="max_value" value="current_value">Fallback Content</meter>

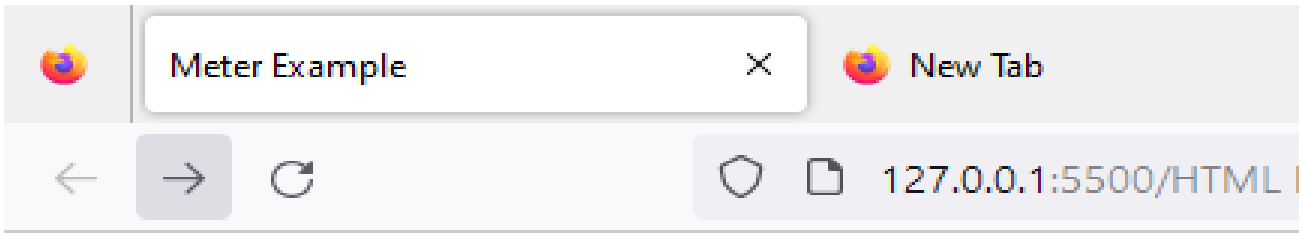
Explanation:

- **<meter>:** This is the opening tag of the meter element.
- **min="min_value":** The min attribute sets the minimum value of the meter's range. It specifies the lowest possible value that can be represented by the meter.
- **max="max_value":** The max attribute sets the maximum value of the meter's range. It specifies the highest possible value that can be represented by the meter.
- **value="current_value":** value attribute specifies the current value of the meter, indicating the measurement or gauge level. This value should be between the min and max values.
- **Fallback Content:** This is the content that will be displayed if the browser does not support the <meter> element. It's a good practice to provide fallback content for compatibility.

Example:

```
<!DOCTYPE html>
<html>
<head>
 <title>Meter Example</title>
</head>
<body>
<h1>Rating</h1>
<!-- A meter element representing a rating -->
<meter min="0" max="5" value="4">4 out of 5</meter>
<p>This product has a rating of 4 out of 5 stars.</p>
</body>
</html>
```

Output:



Rating

This product has a rating of 4 out of 5 stars.

13. <keygen>:

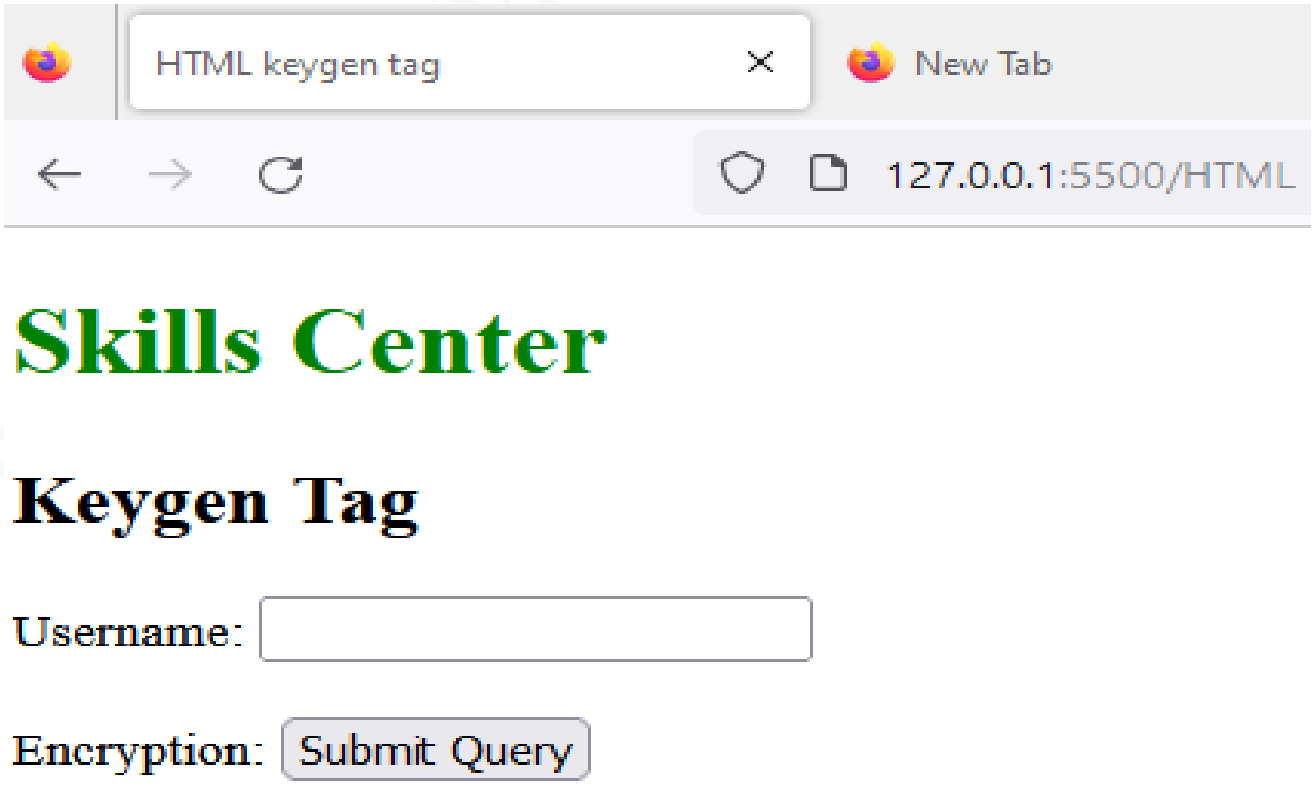
Syntax:

- <keygen name = "name">

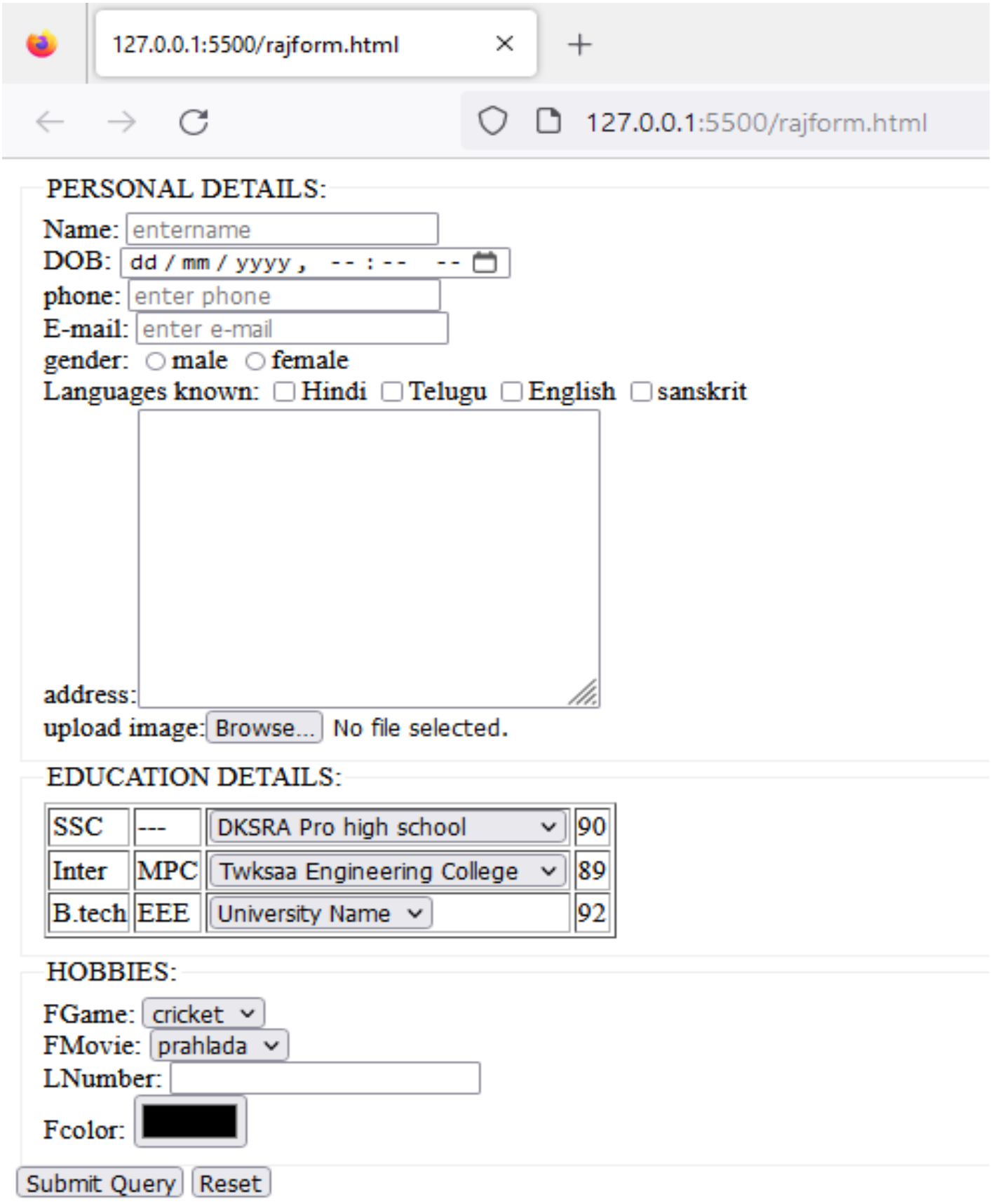
Example:

```
<!DOCTYPE html>
<html>
                    <head>
                    <title>
                        HTML keygen tag
                    </title>
                    </head>
                    <body>
                    <h1 style = "color:green;">
                        Skills Center
                    </h1>
                    <h2>Keygen Tag</h2>
                    <form>
                        Username: <input type="text" name="uname">
                        <br><br>
                        Encryption: <keygen name="secure">
                        <input type="submit">
                    </form>
                    </body>
</html>
```

Output:



Our Requirements:



Example:

```
<html>
<head>
</head>
<body>
  <form>
    <fieldset>
      <legend>PERSONAL DETAILS:</legend>
        Name:
        <input type="text"placeholder="entername"><br
        DOB:
        <input type="datetime-local"placeholder=""><br>
        phone:
        <input type="text"placeholder="enter phone"><br>
        E-mail:
        <input type="text"placeholder="enter e-mail"><br>
        gender:
        <input type="radio"value="male"name="gen">male
        <input type="radio"value="female"name="gen">female</br>
        Languages known:
          <input type="checkbox"value="Hindi"/>Hindi
         <input type="checkbox"value="Telugu"/>Telugu
         <input type="checkbox"value="English"/>English
         <input type="checkbox"value="sanskrit"/>sanskrit<br>
        address:<textarea rows="10" cols="30" readonly></textarea></br>
        <tr><td>upload image:<input type="file"/></td></tr
    </fieldset>
    <fieldset>
      <legend>EDUCATION DETAILS:</legend>
      <table border="1">
        <tr>
        <td colspan="4">SSC</td>
        <td colspan="4">---</td>
        <td>
          <select>
            <option value="1"selected>Sant Shiva Nand high school</option>
            <option value="1"selected>DKSRA high school</option>
            <option value="1"selected>DKSRA Pro high school</option>
         </select>
         </td>
        <td colspan="4">90</td>
        </tr>
        <tr>
        <td colspan="4">Inter</td>
        <td colspan="4">MPC</td>
        <td>
          <select>
```

```
                <option value="1"selected>Dksra college of Engineering</option>
                <option value="1"selected>xyz college name</option>
                <option value="1"selected>Twksaa Engineering College</option>
              </select>
             </td>
             <td colspan="4">89</td>
            </tr>
            <tr>
             <td colspan="4">B.tech</td>
             <td colspan="4">EEE</td>
             <td>
               <select>
                 <option value="1"selected>DU</option>
                 <option value="1"selected>TTU</option>
                 <option value="1"selected>University Name</option>
               </select>
             </td>
             <td>92</td>
           </tr>
           </table>
        </fieldset>
        <fieldset>
           <legend>HOBBIES:</legend>
        FGame:
          <select>
           <option value="1"selected>cricket</option>
          </select><br>
        FMovie:
          <select>
           <option value="1"selected>prahlada</option>
          </select></br>
        LNumber:
           <input type="text"></br>
        Fcolor:
           <input type="color"/>
        </fieldset>
       <form>
        <input type="submit"/>
        <input type="reset"/>
       </form>
</body>
```

HTML Iframe tag:

- HTML <iframe> (short for inline frame) is used to embed another document or web page within the current HTML document.
- It is often used to display content from another website or to embed multimedia elements like videos, maps, or interactive widgets.

syntax:

<iframe src="URL" width="width" height="height" frameborder="0" scrolling="auto"></iframe>

- **src:** This attribute specifies the URL of the web page or content you want to embed within the <iframe>. It is a required attribute.
- **width and height:** These attributes specify the width and height of the <iframe> in pixels. You can adjust these values to control the size of the embedded content. These attributes are optional but recommended.
- **frameborder:** This attribute specifies whether or not to display a border around the <iframe>. A value of "0" (zero) means no border, while a value of "1" means a border should be displayed. Typically, you'd set this to "0" for a clean look. This attribute is optional.
- **scrolling:** This attribute controls the scrollbars within the <iframe>. You can set it to "auto" (default behavior, showing scrollbars as needed), "yes" (always show scrollbars), or "no" (never show scrollbars). This attribute is optional.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Embedding a Google Map</title>
</head>
<body>
  <h1>My Google Map</h1>
  <iframe
    src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3874.188419871245!2d-122.08219688554769!3d37.42199977978261!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x808fbbd84db959c7%3A0x9fc89dc91a5744f3!2sGolden%20Gate%20Bridge!5e0!3m2!1sen!2sus!4v1618225620622!5m2!1sen!2sus"
    width="600"
    height="450"
    frameborder="0"
    scrolling="no">
  </iframe>
</body>
</html>
```

HTML File Paths Tag:

- HTML file paths are used to reference other files, such as images, stylesheets, scripts, or even other web pages, within an HTML document. There are two main types of file paths you can use in HTML:

1. Absolute Paths: These paths specify the full URL or file system path to the resource, starting from the root directory. Absolute paths are often used for resources located on different websites or servers.

2. Relative Paths: These paths specify the location of the resource relative to the current HTML document's location. Relative paths are commonly used for resources within the same website or directory structure.

- **syntax for both types of file paths with examples:**

❖ Absolute Paths:

1. Absolute URL: This is used to reference resources hosted on different websites or servers.

Syntax:

```
<a href="https://www.example.com/page.html">Link Text</a>
<img src="https://www.example.com/images/image.jpg" alt="Image">
```

2. Absolute File Path: This is used to reference files on your local system using a file path.

Syntax (Windows):

```
<img src="file:///C:/path/to/your/image.jpg" alt="Image">
```

Syntax (Unix/Linux):

```
<img src="file:///home/user/path/to/your/image.jpg" alt="Image">
```

❖ Relative Paths:

- Relative paths are more commonly used when referencing resources within the same website or directory structure.

1. Relative to the Current Directory: Use these paths when the resource is located in the same directory as the HTML file.

Syntax:

```
<img src="image.jpg" alt="Image">
<a href="page.html">Link Text</a>
```

2. Relative to a Subdirectory: If the resource is located in a subdirectory of the current directory, specify the path relative to the current directory.

Syntax:

```
<img src="images/image.jpg" alt="Image">
<a href="pages/page.html">Link Text</a>
```

3. Relative to the Root Directory: You can use a forward slash (/) at the beginning of the path to specify a path relative to the root directory of the website.

Syntax:

```
<img src="/images/image.jpg" alt="Image">
<a href="/pages/page.html">Link Text</a>
```

4. Parent Directory: To reference a resource in a parent directory (one level up), use ".." to navigate up the directory structure.

Syntax:

```
<img src="../image.jpg" alt="Image">
<a href="../page.html">Link Text</a>
```

Note: Keep in mind that when using relative paths, the actual structure of your website's directories and the location of the HTML file play a crucial role. Always double-check the file path to ensure it is correct relative to the HTML document's location.

Example of relative paths:

- Assuming the following directory structure:

- mywebsite/
  - index.html
  - images/
    - image.jpg
  - pages/
    - page.html
      - In `index.html`, you can use relative paths like this:

```
<img src="images/image.jpg" alt="Image">
<a href="pages/page.html">Link Text</a>
```

- These paths are relative to the location of `index.html`.

HTML Semantic Elements

- Semantic elements tags are carried meaning about the structure and content of a web page.
- It provides a way to describe type of content contained within the tag, making it clear to both browsers and developers what the purpose. Semantic elements = elements with a meaning.
- By Using this you can improves accessibility and search engine optimization (SEO).

Example of Semantics Tag:

1. <article>:
- Represents a self-contained piece of content, such as a blog post or news article.
- Typically contains content that can be distributed and reused independently.

2. <aside>:
- Represents content that is tangentially related to the main content but can stand alone.
- Often used for sidebars, pull-out quotes, or advertisements within a page.

3. <details>:
- Defines additional details or information that can be shown or hidden.
- Often used with the `<summary>` element to provide a title or label for the details.

4. <figcaption>:
- Typically used within the `<figure>` element.
- Represents a caption or description for an image or other media content.

5. <figure>:
- Used to encapsulate media content, such as images, videos, or diagrams.
- Often accompanied by a `<figcaption>` to provide a caption.

6. <footer>:
- Represents the footer section of a web page or a section.
- Contains information like copyright notices, contact details, or related links.

7. <header>:
- Represents introductory content at the beginning of a section or a page.
- Usually contains headings, logos, navigation menus, and other content related to the top of page.

8. <main>:
- Represents the main content of a document.
- Should appear only once per page and defines the primary content area.

9. <mark>:
- Highlights text within the content, often for indicating search results or relevant terms.

10. <nav>:
- Represents a section of navigation links.
- Typically contains links to various sections or pages within a website.

11. <section>:
- Used to group related content together.
- Provides a way to semantically mark up different sections of a web page.

12. <summary>:
- Typically used within the `<details>` element.
- Provides a visible label or title for the hidden details that can be expanded.

13. <time>:
- Represents a specific period in time or a range of dates.
- Can be used to markup dates, times, or durations.

HTML Non-Semantic Elements

- Non-semantic HTML elements are tags that do not convey any specific meaning about the content they contain.
- These tags are typically used for layout and formatting purposes and may not provide any semantic structure to the content.

Example:

1. (Division):

- The `<div>` element is a generic container used for grouping and styling content.
- It does not convey any specific meaning or semantic information.

```
<div>
  <p>This is some text inside a div.</p>
  <ul>
    <li>Item 1</li>
    <li>Item 2</li>
  </ul>
</div>
```

2. :

- The `<span>` element is similar to `<div>` but is used for inline styling and grouping of text or inline elements.
- It does not provide semantic meaning to the content.

```
<p>This is a <span style="color: blue;">blue</span> word.</p>
```

3.
 (Line Break):

- The `<br>` element is used to insert a line break, which forces content to start on a new line.

```
<p>This is some text.<br>Here's a new line.</p>
```

4. <b> and <i>(Bold and Italic):

- The `<b>` element is used for making text bold, and the `<i>` element is used for italicizing text.
- They are presentational elements and do not convey the semantic meaning of the text.

```
<p><b>This is bold text.</b> <i>This is italic text.</i></p>
```

5. <font> (Font):

- The `<font>` element was historically used to define font styles, sizes, and colors for text.
- It is a non-semantic and largely deprecated element in modern web development.

```
<p><font color="red" size="4">Red text with size 4.</font></p>
```

6. (Center Alignment):

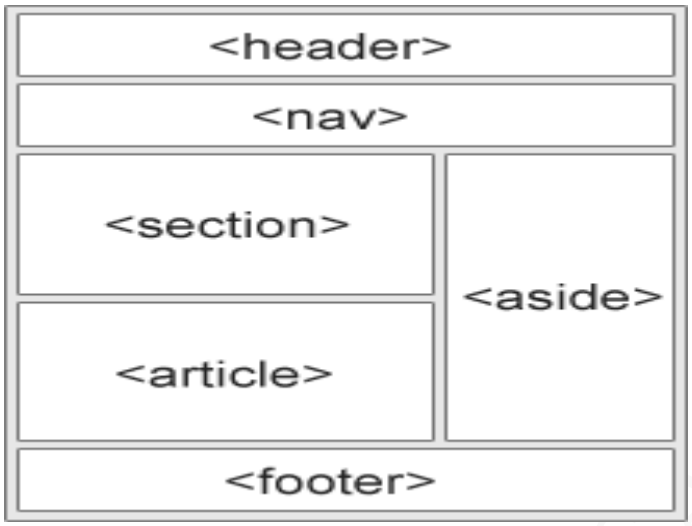
- The `<center>` element was used to center-align content.
- Like `<font>`, it is also deprecated in modern HTML and CSS is preferred for layout and alignment.

```
<center>This content is centered. </center>
```

Note: It's generally recommended to use semantic HTML elements whenever possible because they provide better structure, accessibility, and compatibility with modern web standards. Non-semantic elements like those listed above are less meaningful and can make it harder to maintain and style your web pages consistently.

HTML Layout Elements

- HTML layout elements are used to structure the content of a web page, defining the overall organization and arrangement of elements.
- Properly structuring your web page with layout elements not only enhances its visual appeal but also aids in creating a well-organized and semantically meaningful document.
- **Diagram:**



❖ HTML layout elements and their explanations:

1. <header>:

- The `<header>` element is used to define a container for introductory content or a set of navigation links at the top of a web page.
- It typically contains the site's logo, site title, main navigation menu, or any other content that should appear at the top of every page.

Example:

```
<header>
 <h1>My Website</h1>
 <nav>
  <ul>
   <li><a href="#">Home</a></li>
   <li><a href="#">About</a></li>
   <li><a href="#">Contact</a></li>
  </ul>
 </nav>
</header>
```

2. <nav>:

- -The `<nav>` element is used to define a section of navigation links, typically inside the `<header>` element.
- It helps users navigate different sections or pages of a website.

-Example:

```
<nav>
 <ul>
```

```
    <li><a href="#">Home</a></li>
    <li><a href="#">About</a></li>
    <li><a href="#">Contact</a></li>
  </ul>
</nav>
```

3. <main>:

- The `<main>` element is used to enclose the main content of a web page.
- It should be unique and appear only once per page.
- Screen readers and search engines use it to identify the primary content.

Example:

```
<main>
  <h1>Welcome to My Website</h1>
  <p>This is the main content of the page.</p>
</main>
```

4. <article>:

- The `<article>` element is used to define a self-contained piece of content within a web page.
- It can represent a blog post, a news article, a forum post, or any content that stands alone.
- It should have a unique topic or subject.

Example:

```
<article>
  <h2>Blog Post Title</h2>
  <p>Content of the blog post...</p>
</article>
```

5. <section>:

- The `<section>` element is used to group related content together.
- It provides a way to semantically markup different sections of a web page.

Example:

```
<section>
  <h2>About Us</h2>
  <p>Information about our company...</p>
</section>
<section>
  <h2>Services</h2>
  <p>Details about the services we offer...</p>
</section>
```

6. <aside>:

- The `<aside>` element is used to define content that is tangentially related to the main content, such as sidebars or pull-out quotes.
- It can be used within an `<article>` or `<section>` element.

Example:

```
<article>
  <h2>Article Title</h2>
  <p>Main content of the article...</p>
  <aside>
    <h3>Related Links</h3>
    <ul>
```

```
      <li><a href="#">Link 1</a></li>
      <li><a href="#">Link 2</a></li>
    </ul>
  </aside>
</article>
```

7. <footer>:

- The `<footer>` element is used to define the footer of a web page or a section.
- It typically contains copyright information, contact details, or links to related resources.

Example:

```
<footer>
  <p>&copy; 2023 My Website</p>
  <p>Contact: contact@mywebsite.com</p>
</footer>
```

8. :

- While not a semantic layout element, the `<div>` element is often used as a generic container to group and style content.
- It doesn't carry any specific meaning by itself and should be used when there is no more suitable semantic element.

Example:

```
<div class="container">
  <h1>Header</h1>
  <p>Main content</p>
</div>
```

Note: Properly using these HTML layout elements not only improves the structure and semantics of your web page but also helps search engines and assistive technologies understand and navigate your content effectively.

HTML Entities

- HTML entities, also known as character entities or HTML character references, are codes used in HTML to represent special characters, symbols, and reserved characters that have special meanings or functions in HTML.

Example:

Result	Description	Entity Name	Entity Number
	non-breaking space		
<	less than	&lt;	&#60;
>	greater than	&gt;	&#62;
&	ampersand	&amp;	&#38;
"	double quotation mark	&quot;	&#34;
'	single quotation mark (apostrophe)	&apos;	&#39;
¢	cent	&cent;	&#162;
£	pound	&pound;	&#163;
¥	yen	&yen;	&#165;
€	euro	&euro;	&#8364;
©	copyright	&copy;	&#169;
®	registered trademark	&reg;	&#174;

1. Non-Breaking Space (` ` or ` `):

- Represents a non-breaking space, which prevents line breaks or word wraps at that point.
- **Example:**
  - `"Hello World"` or `"Hello World"` would display "Hello World" with a non-breaking space between the words.

2. Less Than (`&lt;` or `&#60;`):

- Represents the less-than sign `<`.
- **Example:** This is bold
- **Displays:** "This is bold."

3. Greater Than (`&gt;` or `&#62;`):

- Represents the greater-than sign `>`.
- **Example:** <script>alert('Hello > World');</script>
- **Displays:** an alert with "Hello > World."

4. Ampersand (`&amp;` or `&#38;`):

- Represents the ampersand `&`.
- **Example:** "AT&T" or `"AT&#38;T"` would
- **Display:** "AT&T."

5. Double Quotation Mark (`&quot;` or `&#34;`):

- Represents the double quotation mark `"`.
- **Example:** "Quoted Text"
- **Displays:** "Quoted Text" within double quotes.

6. Single Quotation Mark (Apostrophe) (`&apos;` or `&#39;`):

- Represents the single quotation mark or apostrophe `'`.
- **Example:** It's a beautiful day
- **Displays:** "It's a beautiful day."

7. Cent (`&cent;` or `&#162;`):

- Represents the cent symbol ¢.
- **Example:** The price is 50¢
- **Displays**: "The price is 50¢."

8. Pound (`&pound;` or `&#163;`):

- Represents the pound sterling symbol £.
- **Example:** The price is £100
- **Displays**: "The price is £100."

9. Yen (`&yen;` or `&#165;`):

- Represents the yen symbol ¥.
- **Example:** The cost is ¥5000
- **Displays:** "The cost is ¥5000."

10. Euro (`&euro;` or `&#8364;`):

- Represents the euro symbol €.
- Example: The total is €50
- Displays: "The total is €50."

11. Copyright (`&copy;` or `&#169;`):

- Represents the copyright symbol ©.
- **Example:** © 2023 My Company
- **Displays:** "© 2023 My Company."

12. Registered Trademark (`&reg;` or `&#174;`):

- Represents the registered trademark symbol ®.
- **Example**: Product® is a registered trademark
- **Displays:** "Product® is a registered trademark."

Note: These HTML character entities are essential for displaying characters with special meanings in HTML or characters that might otherwise cause issues when used directly in HTML content. They help ensure that web pages are displayed correctly and maintain compatibility with various browsers and devices.

HTML Symbols

❖ Mathematical Symbols:

1. Basic Operators:

- `+` Addition: `&plus;` or `&#43;`
  Example: `5 &plus; 3` displays as "5 + 3."
- `-` Subtraction: `&minus;` or `&#8722;`
  Example: `10 &minus; 4` displays as "10 - 4."
- × Multiplication: `&times;` or `&#215;`
  Example: `6 &times; 9` displays as "6 × 9."
- ÷ Division: `&divide;` or `&#247;`
  Example: `12 &divide; 4` displays as "12 ÷ 4."
- = Equals: `&equals;` or `&#61;`
  Example: `2 &plus; 2 &equals; 4` displays as "2 + 2 = 4."

2. Exponents and Superscripts:

- ² Squared: `&sup2;` or `&#178;`
  Example: `x²` displays as "x²."
- ³ Cubed: `&sup3;` or `&#179;`
  Example: `y³` displays as "y³."
- n^n Superscript n: `&nSup;` (Replace "Sup" with the desired number)
  Example: `a⁴` displays as "a⁴."
- x^n Custom Superscript: `&xsupn;` (Replace "x" with the base and "n" with the exponent)
  Example: `b⁵` displays as "b⁵."

3. Square Root and Radicals:

- √ Square Root: `&radic;` or `&#8730;`
  Example: `√16` displays as "√16."
- $\sqrt[n]{}$ Custom Root: `&nsqrt;` (Replace "n" with the desired root)
  Example: `⁵√243` displays as "⁵√243."

4. Fractions:

- ½ One Half: `&frac12;` or `&#189;`
  Example: `¾ + ½` displays as "¾ + ½."
- ¼ One Fourth: `&frac14;` or `&#188;`
  Example: `1/4 + 3/4` displays as "1/4 + 3/4."
- ¾ Three Fourths: `&frac34;` or `&#190;`
  Example: `⅔ + ¾` displays as "⅔ + ¾."

5. Greek Letters:

- α Alpha: `&alpha;` or `&#945;`
  Example: `Δ = &alpha; + &beta;` displays as "Δ = α + β."
- β Beta: `&beta;` or `&#946;`
  Example: `θ = &beta; × &gamma;` displays as "θ = β × γ."
- Σ Sigma (Summation): `&Sigma;` or `&#931;`
  Example: `&Sigma; i = 1 to n` represents summation.
- π Pi: `&pi;` or `&#960;`
  Example: `π ≈ 3.14159` displays an approximation of pi.
- θ Theta: `&theta;` or `&#952;`
  Example: `θ = 45°` displays an angle in degrees.

6. Inequality Symbols:

- ≤ Less Than or Equal To: `&leq;` or `&#8804;`
  Example: `x &leq; y` displays as "x ≤ y."
- ≥ Greater Than or Equal To: `&geq;` or `&#8805;`
  Example: `a &geq; b` displays as "a ≥ b."
- ≠ Not Equal To: `&ne;` or `&#8800;`
  Example: `x &ne; y` displays as "x ≠ y."

7. Logical Operators:

- ∧ Logical AND: `&and;` or `&#8743;`
  Example: `A &and; B` displays as "A ∧ B."
- ∨ Logical OR: `&or;` or `&#8744;`
  Example: `X &or; Y` displays as "X ∨ Y."
- ¬ Logical NOT: `&not;` or `&#172;`
  Example: `¬P` displays as "¬P."

8. Infinity:

- ∞ Infinity: `&infin;` or `&#8734;`
  Example: `1/∞ = 0` represents the concept of infinity.

9. Summation and Integration:

- ∑ Summation: `&sum;` or `&#8721;`
  Example: `∑(i = 1 to n) $x_i$` represents summation.
- ∫ Integral: `&int;` or `&#8747;`
  Example: `∫(0 to 1) f(x) dx` represents integration.

10. Set Theory:

- ∈ Element Of: `&isin;` or `&#8712;`
  Example: `x ∈ A` displays as "x ∈ A."
- ∉ Not an Element Of: `&notin;` or `&#8713;`
  Example: `y ∉ B` displays as "y ∉ B."
- ∪ Union: `&cup;` or `&#8746;`
  Example: `A ∪ B` represents the union of sets A and B.
- ∩ Intersection: `&cap;` or `&#8745;`
  Example: `X ∩ Y` represents the intersection of sets X and Y.

Example:

Char	Number	Entity	Description
∀	`&#8704;`	`&forall;`	FOR ALL
∂	`&#8706;`	`&part;`	PARTIAL DIFFERENTIAL
∃	`&#8707;`	`&exist;`	THERE EXISTS
∅	`&#8709;`	`&empty;`	EMPTY SET
∇	`&#8711;`	`&nabla;`	NABLA
∈	`&#8712;`	`&isin;`	ELEMENT OF
∉	`&#8713;`	`&notin;`	NOT AN ELEMENT OF
∋	`&#8715;`	`&ni;`	CONTAINS AS MEMBER
∏	`&#8719;`	`&prod;`	N-ARY PRODUCT
∑	`&#8721;`	`&sum;`	N-ARY SUMMATION

1. FOR ALL (∀): `&forall;`
   ∀x, P(x) means "for all x, P(x) is true."
2. PARTIAL DIFFERENTIAL (∂): `&part;`
   ∂f/∂x represents the partial derivative of f with respect to x.
3. THERE EXISTS (∃): `&exist;`
   ∃x, Q(x) means "there exists an x for which Q(x) is true."
4. ELEMENT OF (∈): `&isin;`
   x ∈ A indicates that x is an element of set A.
5. NOT AN ELEMENT OF (∉): `&notin;`
   y ∉ B means that y is not an element of set B.
6. N-ARY PRODUCT (∏): `&prod;`
   The product of a sequence of numbers: ∏(i = 1 to n) xi
7. N-ARY SUMMATION (∑): `&sum;`
   The summation of a sequence of numbers: ∑(i = 1 to n) xi

❖ Some Other Entities:

Char	Number	Entity	Description
©	©	©	COPYRIGHT SIGN
®	®	®	REGISTERED SIGN
€	€	€	EURO SIGN
™	™	™	TRADEMARK
←	←	←	LEFTWARDS ARROW
↑	↑	↑	UPWARDS ARROW
→	→	→	RIGHTWARDS ARROW
↓	↓	↓	DOWNWARDS ARROW
♠	♠	♠	BLACK SPADE SUIT
♣	♣	♣	BLACK CLUB SUIT
♥	♥	♥	BLACK HEART SUIT
♦	♦	♦	BLACK DIAMOND SUIT

Emojis in HTML

- Emojis are characters from the UTF-8-character set: 😄 😍 ❤️

❖ What are Emojis?

- Emojis look like images, or icons, but they are not.
- They are letters (characters) from the UTF-8 (Unicode) character set.
- The HTML charset Attribute
- To display an HTML page correctly, a web browser must know the character set used in the page.
- This is specified in the <meta> tag:
- <meta charset="UTF-8">

❖ UTF-8 Characters

- Many UTF-8 characters cannot be typed on a keyboard, but they can always be displayed using numbers (called entity numbers):
- A is 65
- B is 66
- C is 67

➢ Emoji Characters

- Emojis are also characters from the UTF-8 alphabet:

😄 is 128516

😍 is 128525

❤️ is 128151

Emoji	Value
🏻	🗻
🏼	🗼
🏽	🗽
🏾	🗾
🏿	🗿
😀	😀
😁	😁
😂	😂
😃	😃
😄	😄
😅	😅

HTML Encoding (Character Sets)

- From ASCII to UTF-8
- ASCII was the first character encoding standard. ASCII defined 128 different characters that could be used on the internet: numbers (0-9), English letters (A-Z), and some special characters like ! $ + - ( ) @ < > .
- ISO-8859-1 was the default character set for HTML 4. This character set supported 256 different character codes. HTML 4 also supported UTF-8.
- ANSI (Windows-1252) was the original Windows character set. ANSI is identical to ISO-8859-1, except that ANSI has 32 extra characters.
- The HTML5 specification encourages web developers to use the UTF-8 character set, which covers almost all of the characters and symbols in the world!
- The HTML charset Attribute
- To display an HTML page correctly, a web browser must know the character set used in the page.
- This is specified in the <meta> tag:
- <meta charset="UTF-8">

❖ Differences Between Character Sets:

Number	ASCII	ANSI	8859	UTF-8	Description
32				space	
33	!	!	!	!	exclamation mark
34	"	"	"	"	quotation mark
35	#	#	#	#	number sign
36	$	$	$	$	dollar sign
37	%	%	%	%	percent sign
38	&	&	&	&	ampersand
39	'	'	'	'	apostrophe
40	(	(	(	(	left parenthesis
41	)	)	)	)	right parenthesis
42	*	*	*	*	asterisk
43	+	+	+	+	plus sign
44	,	,	,	,	comma
45	-	-	-	-	hyphen-minus
46	.	.	.	.	full stop
47	/	/	/	/	solidus
48	0	0	0	0	digit zero
49	1	1	1	1	digit one
50	2	2	2	2	digit two
51	3	3	3	3	digit three
52	4	4	4	4	digit four
53	5	5	5	5	digit five
54	6	6	6	6	digit six
55	7	7	7	7	digit seven
56	8	8	8	8	digit eight
57	9	9	9	9	digit nine
58	:	:	:	:	colon

59	;	;	;	;	semicolon
60	<	<	<	<	less-than sign
61	=	=	=	=	equals sign
62	>	>	>	>	greater-than sign
63	?	?	?	?	question mark
64	@	@	@	@	commercial at
65	A	A	A	A	Latin capital letter A
66	B	B	B	B	Latin capital letter B
67	C	C	C	C	Latin capital letter C
68	D	D	D	D	Latin capital letter D
69	E	E	E	E	Latin capital letter E
70	F	F	F	F	Latin capital letter F
71	G	G	G	G	Latin capital letter G
72	H	H	H	H	Latin capital letter H
73	I	I	I	I	Latin capital letter I
74	J	J	J	J	Latin capital letter J
75	K	K	K	K	Latin capital letter K
76	L	L	L	L	Latin capital letter L
77	M	M	M	M	Latin capital letter M
78	N	N	N	N	Latin capital letter N
79	O	O	O	O	Latin capital letter O
80	P	P	P	P	Latin capital letter P
81	Q	Q	Q	Q	Latin capital letter Q
82	R	R	R	R	Latin capital letter R
83	S	S	S	S	Latin capital letter S
84	T	T	T	T	Latin capital letter T
85	U	U	U	U	Latin capital letter U
86	V	V	V	V	Latin capital letter V
87	W	W	W	W	Latin capital letter W
88	X	X	X	X	Latin capital letter X
89	Y	Y	Y	Y	Latin capital letter Y
90	Z	Z	Z	Z	Latin capital letter Z
91	[	[	[	[	left square bracket
92	\	\	\	\	reverse solidus
93	]	]	]	]	right square bracket
94	^	^	^	^	circumflex accent
95	_	_	_	_	low line
96	`	`	`	`	grave accent
97	a	a	a	a	Latin small letter a
98	b	b	b	b	Latin small letter b
99	c	c	c	c	Latin small letter c
100	d	d	d	d	Latin small letter d
101	e	e	e	e	Latin small letter e
102	f	f	f	f	Latin small letter f
103	g	g	g	g	Latin small letter g
104	h	h	h	h	Latin small letter h
105	i	i	i	i	Latin small letter i
106	j	j	j	j	Latin small letter j

107	k	k	k	k	Latin small letter k
108	l	l	l	l	Latin small letter l
109	m	m	m	m	Latin small letter m
110	n	n	n	n	Latin small letter n
111	o	o	o	o	Latin small letter o
112	p	p	p	p	Latin small letter p
113	q	q	q	q	Latin small letter q
114	r	r	r	r	Latin small letter r
115	s	s	s	s	Latin small letter s
116	t	t	t	t	Latin small letter t
117	u	u	u	u	Latin small letter u
118	v	v	v	v	Latin small letter v
119	w	w	w	w	Latin small letter w
120	x	x	x	x	Latin small letter x
121	y	y	y	y	Latin small letter y
122	z	z	z	z	Latin small letter z
123	{	{	{	{	left curly bracket
124	|	|	|	|	vertical line
125	}	}	}	}	right curly bracket
126	~	~	~	~	tilde
127	DEL				
128		€			euro sign
129		•	•	•	NOT USED
130		‚			single low-9 quotation mark
131		ƒ			Latin small letter f with hook
132		„			double low-9 quotation mark
133		…			horizontal ellipsis
134		†			dagger
135		‡			double dagger
136		ˆ			modifier letter circumflex accent
137		‰			per mille sign
138		Š			Latin capital letter S with caron
139		‹			single left-pointing angle quotation mark
140		Œ			Latin capital ligature OE
141		•	•	•	NOT USED
142		Ž			Latin capital letter Z with caron
143		•	•	•	NOT USED
144		•	•	•	NOT USED
145		‘			left single quotation mark
146		’			right single quotation mark
147		“			left double quotation mark
148		”			right double quotation mark
149		•			bullet
150		–			en dash
151		—			em dash
152		˜			small tilde
153		™			trade mark sign
154		š			Latin small letter s with caron

155	›			single right-pointing angle quotation mark
156	œ			Latin small ligature oe
157	•	•	•	NOT USED
158	ž			Latin small letter z with caron
159	Ÿ			Latin capital letter Y with diaeresis
160				no-break space
161	¡	¡	¡	inverted exclamation mark
162	¢	¢	¢	cent sign
163	£	£	£	pound sign
164	¤	¤	¤	currency sign
165	¥	¥	¥	yen sign
166	¦	¦	¦	broken bar
167	§	§	§	section sign
168	¨	¨	¨	diaeresis
169	©	©	©	copyright sign
170	ª	ª	ª	feminine ordinal indicator
171	«	«	«	left-pointing double angle quotation mark
172	¬	¬	¬	not sign
173	-	-	-	soft hyphen
174	®	®	®	registered sign
175	¯	¯	¯	macron
176	°	°	°	degree sign
177	±	±	±	plus-minus sign
178	²	²	²	superscript two
179	³	³	³	superscript three
180	´	´	´	acute accent
181	µ	µ	µ	micro sign
182	¶	¶	¶	pilcrow sign
183	·	·	·	middle dot
184	¸	¸	¸	cedilla
185	¹	¹	¹	superscript one
186	º	º	º	masculine ordinal indicator
187	»	»	»	right-pointing double angle quotation mark
188	¼	¼	¼	vulgar fraction one quarter
189	½	½	½	vulgar fraction one half
190	¾	¾	¾	vulgar fraction three quarters
191	¿	¿	¿	inverted question mark
192	À	À	À	Latin capital letter A with grave
193	Á	Á	Á	Latin capital letter A with acute
194	Â	Â	Â	Latin capital letter A with circumflex
195	Ã	Ã	Ã	Latin capital letter A with tilde
196	Ä	Ä	Ä	Latin capital letter A with diaeresis
197	Å	Å	Å	Latin capital letter A with ring above
198	Æ	Æ	Æ	Latin capital letter AE
199	Ç	Ç	Ç	Latin capital letter C with cedilla
200	È	È	È	Latin capital letter E with grave
201	É	É	É	Latin capital letter E with acute
202	Ê	Ê	Ê	Latin capital letter E with circumflex

203	Ë	Ë	Ë	Latin capital letter E with diaeresis
204	Ì	Ì	Ì	Latin capital letter I with grave
205	Í	Í	Í	Latin capital letter I with acute
206	Î	Î	Î	Latin capital letter I with circumflex
207	Ï	Ï	Ï	Latin capital letter I with diaeresis
208	Ð	Ð	Ð	Latin capital letter Eth
209	Ñ	Ñ	Ñ	Latin capital letter N with tilde
210	Ò	Ò	Ò	Latin capital letter O with grave
211	Ó	Ó	Ó	Latin capital letter O with acute
212	Ô	Ô	Ô	Latin capital letter O with circumflex
213	Õ	Õ	Õ	Latin capital letter O with tilde
214	Ö	Ö	Ö	Latin capital letter O with diaeresis
215	×	×	×	multiplication sign
216	Ø	Ø	Ø	Latin capital letter O with stroke
217	Ù	Ù	Ù	Latin capital letter U with grave
218	Ú	Ú	Ú	Latin capital letter U with acute
219	Û	Û	Û	Latin capital letter U with circumflex
220	Ü	Ü	Ü	Latin capital letter U with diaeresis
221	Ý	Ý	Ý	Latin capital letter Y with acute
222	Þ	Þ	Þ	Latin capital letter Thorn
223	ß	ß	ß	Latin small letter sharp s
224	à	à	à	Latin small letter a with grave
225	á	á	á	Latin small letter a with acute
226	â	â	â	Latin small letter a with circumflex
227	ã	ã	ã	Latin small letter a with tilde
228	ä	ä	ä	Latin small letter a with diaeresis
229	å	å	å	Latin small letter a with ring above
230	æ	æ	æ	Latin small letter ae
231	ç	ç	ç	Latin small letter c with cedilla
232	è	è	è	Latin small letter e with grave
233	é	é	é	Latin small letter e with acute
234	ê	ê	ê	Latin small letter e with circumflex
235	ë	ë	ë	Latin small letter e with diaeresis
236	ì	ì	ì	Latin small letter i with grave
237	í	í	í	Latin small letter i with acute
238	î	î	î	Latin small letter i with circumflex
239	ï	ï	ï	Latin small letter i with diaeresis
240	ð	ð	ð	Latin small letter eth
241	ñ	ñ	ñ	Latin small letter n with tilde
242	ò	ò	ò	Latin small letter o with grave
243	ó	ó	ó	Latin small letter o with acute
244	ô	ô	ô	Latin small letter o with circumflex
245	õ	õ	õ	Latin small letter o with tilde
246	ö	ö	ö	Latin small letter o with diaeresis
247	÷	÷	÷	division sign
248	ø	ø	ø	Latin small letter o with stroke
249	ù	ù	ù	Latin small letter u with grave
250	ú	ú	ú	Latin small letter u with acute

251	û	û	û	Latin small letter with circumflex
252	ü	ü	ü	Latin small letter u with diaeresis
253	ý	ý	ý	Latin small letter y with acute
254	þ	þ	þ	Latin small letter thorn
255	ÿ	ÿ	ÿ	Latin small letter y with diaeresis

❖ The ASCII Character Set:

- ASCII uses the values from 0 to 31 (and 127) for control characters.
- ASCII uses the values from 32 to 126 for letters, digits, and symbols.
- ASCII does not use the values from 128 to 255.

❖ The ANSI Character Set (Windows-1252):

- ANSI is identical to ASCII for the values from 0 to 127.
- ANSI has a proprietary set of characters for the values from 128 to 159.
- ANSI is identical to UTF-8 for the values from 160 to 255.

❖ The ISO-8859-1 Character Set:

- ISO-8859-1 is identical to ASCII for the values from 0 to 127.
- ISO-8859-1 does not use the values from 128 to 159.
- ISO-8859-1 is identical to UTF-8 for the values from 160 to 255.

❖ The UTF-8 Character Set:

- UTF-8 is identical to ASCII for the values from 0 to 127.
- UTF-8 does not use the values from 128 to 159.
- UTF-8 is identical to both ANSI and 8859-1 for the values from 160 to 255.
- UTF-8 continues from the value 256 with more than 10 000 different characters.

.

HTML Versus XHTML

- HTML (Hypertext Markup Language) and XHTML (Extensible Hypertext Markup Language) are both markup languages used for creating web pages and defining the structure and content of web documents.
- differences between HTML AND XHTML including their syntax, rules, and parsing behavior.

1. Syntax:

- **HTML:** HTML has a more forgiving syntax. It allows for many shorthand notations and is less strict about closing tags, attribute quoting, and uppercase/lowercase differences. For example, you can write `<img src="image.jpg">` instead of `<img src="image.jpg" />`.
- **XHTML:** XHTML has a stricter syntax that adheres to XML rules. All elements and attributes must be properly closed, enclosed in lowercase, and attribute values must be quoted. For example, `<img src="image.jpg" />` is the correct format.

2. Document Structure:

- **HTML:** In HTML, certain elements like `<html>`, `<head>`, and `<body>` are often optional, and omitting them won't break the document. Some elements can be self-closing or omitted, like `<img>` and `<br>`.
- **XHTML:** In XHTML, all elements must be properly nested within an `<html>` element, and the document structure must adhere strictly to the Document Type Definition (DTD). Empty elements must be explicitly self-closed (e.g., `<img />`), and all elements must be closed properly

3. Quoting Attributes:

- HTML: In HTML, attribute values can often be unquoted or enclosed in single or double quotes. For example, `class=example` or `class='example'` is acceptable.
- XHTML: In XHTML, attribute values must be enclosed in double quotes, like `class="example"`.

4. Character Encoding:

- HTML: In HTML, specifying a character encoding is recommended but not required. You can use `<meta charset="UTF-8">` to specify the character encoding.
- XHTML: In XHTML, specifying a character encoding is mandatory, and it's typically done using `<meta charset="UTF-8" />`.

5. Error Handling:

- HTML: HTML browsers tend to be more lenient with syntax errors and may attempt to render the page even if there are errors in the markup.
- XHTML: XHTML parsers are stricter and less forgiving of errors. A single syntax error can prevent the entire page from rendering.

6. MIME Type:

- HTML: HTML documents are typically served with the MIME type `text/html`.
- XHTML: XHTML documents are served with the MIME type `application/xhtml+xml`.

7. Compatibility:

- HTML: HTML is more compatible with older browsers and web standards. It's a better choice for projects where compatibility with legacy systems is a concern.
- XHTML: XHTML is seen as a more modern and stricter markup language. It's often used in projects that emphasize standards compliance and adherence to XML rules.

HTML Responsive Web Design

- HTML Responsive Web Design is an approach to web design and development that aims to create web pages that adapt and respond to various screen sizes and devices.
- The goal is to ensure that a website looks and functions well on desktop computers, laptops, tablets, smartphones, and other devices with varying screen dimensions and resolutions.
- Responsive web design (RWD) is achieved by using HTML, CSS (Cascading Style Sheets), and sometimes JavaScript to create a flexible and fluid layout that adjusts to the user's device.

❖ principles and techniques for HTML Responsive Web Design:

1. Fluid Layouts:

- Instead of fixed-width layouts, responsive designs use fluid or flexible grids. This means that elements on the page are sized using relative units like percentages instead of fixed pixels. As the screen size changes, these elements adjust proportionally.

2. Media Queries:

- Media queries are CSS rules that allow you to apply specific styles based on characteristics of the user's device, such as screen width, height, and orientation (landscape or portrait). Media queries enable you to create different layouts and styles for different screen sizes.

3. Flexible Images and Media:

- Images and media elements should also be responsive. CSS properties like `max-width: 100%;` can be applied to ensure that images scale proportionally within their parent containers, preventing them from overflowing on smaller screens.

4. Viewport Meta Tag:

- The viewport meta tag (`<meta name="viewport">`) is used in the HTML `<head>` section to control how a web page is displayed on mobile devices. It allows you to set the initial scale and size of the viewport.
- <meta name="viewport" content="width=device-width, initial-scale=1.0">

5. Mobile-First Approach:

- Many responsive designs follow a mobile-first approach, where the initial design is optimized for smaller screens and progressively enhanced for larger screens. This ensures that the website performs well on mobile devices, which are often the smallest and most challenging to design for.

6. CSS Flexbox and Grid:

- CSS Flexbox and Grid layout techniques are commonly used to create responsive and flexible page structures. They provide powerful tools for arranging elements within containers, making it easier to create complex layouts that adapt to different screen sizes.

7. Content Prioritization:

- On smaller screens, it's essential to prioritize content and decide which elements are most critical to display. Responsive designs often involve hiding or reordering elements for mobile users to improve user experience.

8. Testing and Debugging:

- Responsive web design requires thorough testing on various devices and screen sizes to identify and address any issues. Browser developer tools and online testing services can help with this process.

9. Performance Optimization:

- Performance is a critical aspect of responsive web design. Optimizing images, using CSS and JavaScript efficiently, and minimizing unnecessary requests are essential to ensure a fast-loading and smooth user experience across devices.

❖ How to make HTML Responsive Web Design:

- Creating a responsive web design involves various techniques and best practices to ensure your website looks and functions well across different devices and screen sizes.

Step 1: Planning Your Design:

- Before you start coding, it's essential to plan your design with responsiveness in mind. Consider the following aspects:
- Content Hierarchy: Determine which content elements are most important and should be prioritized for mobile users.
- Grid System: Decide on a grid system to structure your layout. Popular options include CSS Grid and Flexbox.
- Media Queries: Plan when and how you'll use media queries to apply different styles based on screen size.

Step 2: HTML Structure:

- Start with a well-structured HTML document. Use semantic HTML elements to define the structure of your web page. Here's an example of a simple HTML structure:

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Responsive Web Design</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <header>
    <h1>My Responsive Website</h1>
    <nav>
      <ul>
        <li><a href="#">Home</a></li>
        <li><a href="#">About</a></li>
        <li><a href="#">Services</a></li>
        <li><a href="#">Contact</a></li>
      </ul>
    </nav>
  </header>
  <main>
    <section>
      <h2>Welcome to Our Website</h2>
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit...</p>
    </section>
    <!-- More content sections here -->
```

```
  </main>
  <footer>
    <p>&copy; 2023 My Website</p>
  </footer>
</body>
</html>
```

Step 3: CSS Styling

- In your CSS file (e.g., `styles.css`), define the styles for your web page. Start with a base style that works well on larger screens, and then use media queries to adjust styles for smaller screens.

Example:

```
/* Base styles for larger screens */
body {
  font-family: Arial, sans-serif;
  background-color: #f0f0f0;
  margin: 0;
  padding: 0;
}
header {
  background-color: #333;
  color: #fff;
  text-align: center;
  padding: 20px;
}
nav ul {
  list-style: none;
}
nav li {
  display: inline;
  margin-right: 20px;
}
nav a {
  text-decoration: none;
  color: #fff;
}
/* Media queries for smaller screens */
@media screen and (max-width: 768px) {
  body {
    font-size: 16px;
  }
  header {
    padding: 10px;
  }
  nav ul {
    text-align: center;
    padding-top: 10px;
  }
```

```
  nav li {
    display: block;
    margin-bottom: 10px;
  }
}
```

- In the above CSS example, we have defined a base style for larger screens and then applied changes using a media query for screens with a maximum width of 768 pixels.

Step 4: Testing and Iteration:

- Test your responsive design on various devices and screen sizes to ensure that it works as expected. Use browser developer tools to simulate different screen sizes and catch any layout issues or inconsistencies.

Step 5: Additional Considerations:

- Images: Use responsive image techniques like `max-width: 100%;` to ensure images scale properly.
- Mobile-First: Consider using a mobile-first approach, where you start with the smallest screen size and progressively enhance for larger screens.
- Accessibility: Ensure your design is accessible to users with disabilities by following accessibility guidelines.

HTML Multimedia

- Such as images, audio, video, and interactive content (Animation), into web pages.

❖ Multimedia Formats:

- It a is specific file types or encoding methods used to store and transmit multimedia content, it can be combination of text, images, audio, video, and interactive elements.
- Multimedia formats are designed to efficiently represent and package different types of media for playback or display on various devices and software applications.

Example:

1. Image Formats:

- JPEG (Joint Photographic Experts Group): A popular format for storing compressed photographic images. It balances image quality and file size.
- PNG (Portable Network Graphics): A lossless image format that supports transparency and is often used for web graphics.
- GIF (Graphics Interchange Format): Supports simple animations and is widely used for short, looping animations and graphics with limited colors.
- BMP (Bitmap): An uncompressed image format that retains high-quality image data but results in larger file sizes.

2. Audio Formats:

- MP3 (MPEG-1 Audio Layer III): A widely used audio compression format that balances audio quality and file size.
- WAV (Waveform Audio File Format): An uncompressed audio format that retains high-quality sound but results in larger file sizes.
- AAC (Advanced Audio Coding): Known for its high-quality audio and efficiency, often used in streaming and mobile devices.
- FLAC (Free Lossless Audio Codec): A lossless audio format that preserves audio quality without compression loss.

3. Video Formats:

- **MP4 (MPEG-4 Part 14):** A versatile video format that can contain video, audio, subtitles, and metadata. Widely used for online streaming and playback.
- **AVI (Audio Video Interleave):** A multimedia container format developed by Microsoft for storing video and audio data.
- **MKV (Matroska):** An open-source multimedia container format known for its flexibility and support for multiple audio and subtitle tracks.
- **WMV (Windows Media Video):** A video format developed by Microsoft for Windows Media Player.

4. Interactive Formats:

- **SWF (Shockwave Flash):** A deprecated format once used for interactive multimedia content and animations on the web. It has been largely replaced by HTML5 technologies.
- **HTML5:** While not a format per se, HTML5, along with JavaScript and CSS, is used to create interactive web content, including multimedia elements like audio, video, and interactive graphics.

5. 3D and VR Formats:

- OBJ (Wavefront Object): A common format for 3D models that can be used in various 3D software and game engines.

- FBX (Filmbox): A proprietary 3D file format developed by Autodesk for 3D modeling and animation.
- WebVR: A web-based format and technology that enables virtual reality experiences directly in web browsers.

6. Document Formats:

- PDF (Portable Document Format): A versatile document format that can contain text, images, hyperlinks, and multimedia elements like audio and video.

Example with HTML:

1. Images (\<img\>):

- The \<img\> element is used to display static images on a web page. You specify the image source (URL) using the "src" attribute. Here's a complete example:

Example:

```
<!DOCTYPE html>
  <html>
  <head>
    <title>Image Example</title>
  </head>
  <body>
    <img src="image.jpg" alt="Description of the image">
  </body>
  </html>
```

- In this example, "image.jpg" is the image file's URL, and "Description of the image" is alternative text that provides a description of the image for accessibility.

2. Audio (\<audio\>):

- The \<audio\> element is used to embed audio files on a web page. You can include audio controls for playback using the "controls" attribute. Here's an example:

Example:

```
 <!DOCTYPE html>
  <html>
  <head>
    <title>Audio Example</title>
  </head>
  <body>
    <audio controls>
      <source src="audio.mp3" type="audio/mpeg">
      Your browser does not support the audio element.
    </audio>
  </body>
  </html.
```

- In this example, "audio.mp3" is the audio file's URL, and the "controls" attribute adds a play/pause control to the audio player.

3. Video (\<video\>):

- The \<video\> element is used to embed video content on a web page. You can include video controls for playback. Here's an example:

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Video Example</title>
</head>
<body>
  <video controls>
    <source src="video.mp4" type="video/mp4">
    Your browser does not support the video element.
  </video>
</body> </html>
```

- In this example, "video.mp4" is the video file's URL, and the "controls" attribute adds play/pause controls to the video player.

4. Canvas (\ <canvas\>):

- The \<canvas\> element provides a space to draw graphics, animations, and interactive content using JavaScript. Here's a basic example:

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Canvas Example</title>
</head>
<body>
  <canvas id="myCanvas" width="400" height="200"></canvas>
  <script>
    var canvas = document.getElementById("myCanvas");
    var ctx = canvas.getContext("2d");
    ctx.fillStyle = "red";
    ctx.fillRect(50, 50, 100, 100);
  </script>
</body>
</html>
```

- In this example, we create a red square on the canvas using JavaScript.

5. SVG (Scalable Vector Graphics):

- SVG is a vector graphics format that can be embedded directly into HTML documents.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>SVG Example</title>
</head>
<body>
  <svg width="100" height="100">
    <circle cx="50" cy="50" r="40" stroke="black" stroke-width="2" fill="red" />
  </svg>
</body></html>
```

import question and short answer in HTML

1. What does HTML stand for?
   - HTML stands for Hypertext Markup Language.
2. What is the latest version of HTML as of your knowledge cutoff date?
   - HTML5 is the latest version of HTML.
3. How do you create a hyperlink in HTML?
   - You create a hyperlink in HTML using the `<a>` (anchor) element.
4. What is the purpose of the `<head>` element in HTML?
   - The `<head>` element contains metadata about the document, such as the title and links to external resources.
5. What is the purpose of the `<meta>` tag in HTML?
   - The `<meta>` tag is used to provide metadata about the HTML document, like character encoding and keywords.
6. How do you add comments in HTML?
   - Comments in HTML are added using `<!-- comment text -->`.
7. What does the `<br>` element do?
   - The `<br>` element is used to insert a line break in the text.
8. What is the purpose of the `<img>` element in HTML?
   - The `<img>` element is used to display images on a webpage.
9. What does the acronym "URL" stand for?
   - URL stands for Uniform Resource Locator.
10. How do you create an ordered list in HTML?
    - You create an ordered list using the `<ol>` element and list items with `<li>`.
11. What is the purpose of the `<table>` element in HTML?
    - The `<table>` element is used to create tabular data.
12. How do you create a form in HTML?
    - You create a form using the `<form>` element.
13. What is the purpose of the `<input>` element in HTML?
    - The `<input>` element is used to create input fields within a form.
14. How do you add a background color to an HTML element?
    - You can add a background color to an HTML element using the CSS `background-color` property.
15. What is the purpose of the `<div>` element in HTML?
    - The `<div>` element is a block-level container used for grouping and styling content.
16. What is the HTML element used for creating a hyperlink with an email address?
    - The `<a>` element with a `mailto:` URL is used to create an email hyperlink.
17. How do you create an unordered list in HTML?
    - You create an unordered list using the `<ul>` element and list items with `<li>`.
18. What is the purpose of the `<iframe>` element in HTML?
    - The `<iframe>` element is used to embed external content, such as a webpage or video, within a document.
19. How do you create a comment that spans multiple lines in HTML?
    - You can create a multiline comment by using `<!--` at the beginning and `-->` at the end of each line.
20. What does the HTML `<title>` element specify?
    - The `<title>` element specifies the title of the document, which is displayed in the browser's title bar or tab.

21. How do you create a clickable button in HTML?
    - You can create a clickable button using the `<button>` element.
22. What is the purpose of the HTML `<header>` element?
    - The `<header>` element typically contains introductory content or navigation links at the top of a webpage.
23. What is the purpose of the HTML `<footer>` element?
    - The `<footer>` element typically contains information about the author, copyright, or contact details at the bottom of a webpage.
24. How do you create a horizontal line in HTML?
    - You create a horizontal line using the `<hr>` element.
25. What is semantic HTML?
    - Semantic HTML is a way of structuring web content using elements that convey meaning, making it more accessible and search engine friendly.
26. **How do you create a drop-down list in HTML?**
    - You create a drop-down list using the `<select>` element and `<option>` elements.

27. What is the purpose of the `<nav>` element in HTML?
    - The `<nav>` element is used to define navigation links on a webpage.
28. **How do you add a hyperlink that opens in a new browser tab or window?**
    - You add the `target="_blank"` attribute to the `<a>` element.
29. What is the purpose of the `<aside>` element in HTML?
    - The `<aside>` element is used for content that is tangentially related to the content around it, such as sidebars.
30. How do you create a bulleted list in HTML?
    - You create a bulleted list using the `<ul>` element and list items with `<li>`.
31. What is the purpose of the HTML `<article>` element?
    - The `<article>` element represents a self-contained piece of content, such as a blog post or news article.
32. How do you add a background image to an HTML element?
    - You can add a background image to an HTML element using the CSS `background-image` property.
33. What is the purpose of the `<video>` element in HTML?
    - The `<video>` element is used to embed video content on a webpage.
34. How do you create a clickable image in HTML?
    - You wrap an `<img>` element with an `<a>` element to make the image clickable.
35. What is the purpose of the `<figcaption>` element in HTML?
    - The `<figcaption>` element is used to provide a caption for a `<figure>` element, typically used with images or diagrams.
36. How do you create a comment that is not visible on the webpage?
    - You can use `<!-- hidden comment -->` to create a comment that won't be displayed in the browser.
37. What is the purpose of the `<main>` element in HTML5?
    - The `<main>` element is used to indicate the main content of a document.
38. How do you create a hyperlink that points to a specific section within the same webpage?
    - You use an anchor (`<a>`) with a `href` attribute pointing to the section's `id` within the same page.
39. What is the purpose of the `<figure>` element in HTML?
    - The `<figure>` element is used to group and represent content, such as images and their captions.
40. How do you create a numbered list in HTML?
    - You create a numbered list using the `<ol>` element and list items with `<li>`.

41. What is the purpose of the `<details>` and `<summary>` elements in HTML?
- The `<details>` element is used to create a disclosure widget, and the `<summary>` element provides a visible heading for it.

42. How do you create a hyperlink that opens an email client with a pre-filled subject?
- You use the `mailto:` scheme in the `href` attribute and include a `subject` parameter

43. What is the purpose of the `<abbr>` element in HTML?
- The `<abbr>` element is used to define an abbreviation or acronym and can provide an expanded form for accessibility.

44. How do you create a clickable text link in HTML?
- You create a clickable text link using the `<a>` element with the link text enclosed between the opening and closing tags.

45. What is the purpose of the `<mark>` element in HTML?
- The `<mark>` element is used to highlight or mark a specific portion of text.

46. How do you create a subscript or superscript in HTML?
- You can use the `<sub>` and `<sup>` elements for subscript and superscript text, respectively.

47. What is the purpose of the HTML `<time>` element?
- The `<time>` element is used to represent a specific time or date.

48. How do you create a text input field with a placeholder in HTML?
- You use the `<input>` element with the `type="text"` attribute and include a `placeholder` attribute.

49. What is the purpose of the HTML `<meter>` element?
- The `<meter>` element is used to represent a scalar measurement within a known range.

50. How do you create a password input field in HTML?
- You use the `<input>` element with the `type="password"` attribute.

51. What is the purpose of the HTML `<progress>` element?
- The `<progress>` element is used to represent the progress of a task.

52. How do you create a radio button in HTML?
- You use the `<input>` element with `type="radio"` for radio buttons, and each radio button should have a unique `name` attribute.

53. What is the purpose of the HTML `<datalist>` element?
- The `<datalist>` element is used to provide a list of predefined options for an `<input>` element with `list` attribute.

54. How do you create a checkbox in HTML?
- You use the `<input>` element with `type="checkbox"` for checkboxes.

55. What is the purpose of the HTML `<optgroup>` element within a `<select>` element?
- The `<optgroup>` element is used to group related `<option>` elements within a `<select>` dropdown.

56. How do you create a file upload input field in HTML?
- You use the `<input>` element with `type="file"`.

57. What is the purpose of the HTML `<textarea>` element?
- The `<textarea>` element is used to create a multiline text input field.

58. How do you create a button input element in HTML?
- You use the `<input>` element with `type="button"`.

59. What is the purpose of the HTML `<label>` element?
- The `<label>` element is used to associate text with a form element, improving accessibility and usability.

60. How do you create a submit button in an HTML form?
- You use the `<input>` element with `type="submit"`.

61. What is the purpose of the HTML `<fieldset>` and `<legend>` elements?
   - The `<fieldset>` element is used to group related form elements, and the `<legend>` element provides a caption for the `<fieldset>`.

62. How do you create a hidden input field in HTML?
   - You use the `<input>` element with `type="hidden"`.
63. What is the purpose of the HTML `<select>` element?
   - The `<select>` element is used to create a dropdown list of options.
64. How do you create an option group in an HTML select element?
   - You use the `<optgroup>` element to group related `<option>` elements within a `<select>`.
65. What is the purpose of the HTML `<canvas>` element?
   - The `<canvas>` element is used to draw graphics, animations, or interactive content using JavaScript.
66. How do you create a reset button in an HTML form?
   - You use the `<input>` element with `type="reset"`.
67. What is the purpose of the HTML `<iframe>` element's `sandbox` attribute?
   - The `sandbox` attribute is used to restrict the behavior of content within an `<iframe>` for security reasons.
68. How do you create a clickable image map in HTML?
   - You use the `<map>` element to define clickable areas on an image and `<area>` elements to specify the clickable regions.
69. What is the purpose of the HTML `<object>` element?
   - The `<object>` element is used to embed external resources like multimedia content or other HTML documents.
70. How do you create a responsive design in HTML and CSS?
   - You use CSS media queries and flexible layout techniques like CSS Grid and Flexbox to create responsive designs that adapt to different screen sizes.

TWKSAA SKILLS CENTER

❖ Research(अनुसंधान):

- अनुसंधान एक प्रणालीकरण कार्य होता है जिसमें विशेष विषय या विषय की नई ज्ञान एवं समझ को प्राप्त करने के लिए सिद्धांतिक जांच और अध्ययन किया जाता है। इसकी प्रक्रिया में डेटा का संग्रह और विश्लेषण, निष्कर्ष निकालना और विशेष क्षेत्र में मौजूदा ज्ञान में योगदान किया जाता है। अनुसंधान के माध्यम से विज्ञान, प्रोधोगिकी, चिकित्सा, सामाजिक विज्ञान, मानविकी, और अन्य क्षेत्रों में विकास किया जाता है। अनुसंधान की प्रक्रिया में अनुसंधान प्रश्न या कल्पनाएँ तैयार की जाती हैं, एक अनुसंधान योजना डिज़ाइन की जाती है, डेटा का संग्रह किया जाता है, विश्लेषण किया जाता है, निष्कर्ष निकाला जाता है और परिणामों को उचित दर्शाने के लिए समाप्ति तक पहुंचाया जाता है।

❖ Innovation(नवीनीकरण): -

- Innovation (इनोवेशन) एक विशेषता या नई विचारधारा की उत्पत्ति या नवीनीकरण है। यह नए और आधुनिक विचारों, तकनीकों, उत्पादों, प्रक्रियाओं, सेवाओं या संगठनात्मक ढंगों का सृजन करने की प्रक्रिया है जिससे समस्याओं का समाधान, प्रतिस्पर्धा में अग्रणी होने, और उपयोगकर्ताओं के अनुकूलता में सुधार किया जा सकता है।

❖ Discovery (आविष्कार):

- Discovery का अर्थ होता है "खोज" या "आविष्कार"। यह एक विशेषता है जो किसी नए ज्ञान, अविष्कार, या तत्व की खोज करने की प्रक्रिया को संदर्भित करता है। खोज विज्ञान, इतिहास, भूगोल, तकनीक, या किसी अन्य क्षेत्र में हो सकती है। इस प्रक्रिया में, व्यक्ति या समूह नए और अज्ञात ज्ञान को खोजकर समझने का प्रयास करते हैं और इससे मानव सभ्यता और विज्ञान-तकनीकी के विकास में योगदान देते हैं।

Note: अनुसंधान विशेषता या विषय पर नई ज्ञान के प्राप्ति के लिए सिस्टमैटिक अध्ययन है, जबकि आविष्कार नए और अज्ञात ज्ञान की खोज है।

TWKSAA RID MISSION

(Research)	(Innovation)	(Discovery)
अनुसंधान करने के महत्वपूर्ण कारण:	**नवीनीकरण करने के महत्वपूर्ण कारण:**	**खोज करने के महत्वपूर्ण कारण:**
1. नई ज्ञान की प्राप्ति	1. प्रगति के लिए	1. नए ज्ञान की प्राप्ति
2. समस्याओं का समाधान	2. परिवर्तन के लिए	2. ज्ञान के विकास में योगदान
3. तकनीकी और व्यापार में उन्नति	3. उत्पादन में सुधार	3. अविष्कारों की खोज
4. विकास को बढ़ावा देना	4. प्रतिस्पर्धा में अग्रणी होने के लिए	4. समस्याओं का समाधान
5. सामाजिक प्रगति	5. समाज को लाभ	5. समाज के उन्नति का माध्यम
6. देश विज्ञान और प्रौद्योगिकी का विकास	6. देश विज्ञान और प्रौद्योगिकी के विकास।	6. देश विज्ञान और तकनीक के विकास

➢ जो लोग रिसर्च, इनोवेशन और डिस्कवरी करते हैं उन लोगों को ही हमें अपना नायक, प्रतीक एवं आदर्श मानना चाहिए क्योंकि यें लोग हमारे समाज, देश एवं विज्ञान के क्षेत्र में प्रगति, विकास और समस्याओं के समाधान में महत्वपूर्ण भूमिका निभाते हैं।

मैं राजेश प्रसाद एक वीणा उठाया हूँ Research, Innovation and Discovery का जिसका मुख्य उदेश्य हैं आने वाले समय में सबसे पहले New(RID, PMS & TLR) की खोज, प्रकाशन एवं उपयोग भारत की इस पावन धरती से ही हो |

"अगर आप भी Research, Innovation and Discovery के क्षेत्र में रूचि रखतें हैं एवं अपनी प्रतिभा से दुनियां को कुछ नया देना चाहतें तो हमारे इस त्वक्सा रीड मिशन (TWKSAA RID MISSION) से जरुर जुड़ें" |

- **राजेश प्रसाद**