

Contents

Lesson 1:	Introduction To HTML	3
1.1	Why to learn HTML?	4
1.2	How Browser interprets HTML.	7
Lesson 2:	HTML Tag	7
2.1	Container tag & Singular tag	8
2.2	Basic Structure of HTML document	9
2.3	HTML Attributes.....	10
Lesson 3:	Element in HTML	10
3.1	Block Level Element	10
3.2	Inline Level Element.....	11
3.3	Some Key Things to know:	13
Lesson 4:	Mastering Text content with HTML	13
4.1	Heading.....	13
4.2	Paragraph	13
Lesson 5:	Text Formatting.	14
5.1	Bold.....	14
5.2	Highlight.....	14
5.3	Italic.....	14
5.4	Underline.....	14
5.5	Inserted, deleted, or strike-through	15
5.6	Super-script & Sub-script.....	15
5.7	Line Break Tag (tag).....	15
5.8	Horizontal line (<Hr> tag).....	15
5.9	Centering content	15
5.10	Monospaced Font	15
5.11	Larger Text / Smaller text.....	16
5.12	Highlight.....	16
Lesson 6:	HTML Link.....	16
Lesson 7:	HTML Images	17
Lesson 8:	HTML Color	17
Lesson 9:	HTML List	19
9.1	Ordered List ()	19
9.2	Unordered List ().....	19
9.3	Description List (<dl>)	20
9.4	Nested List	20
Lesson 10:	Table in HTML.....	20
Lesson 11:	Marquee In HTML	22
Lesson 12:	Frames in HTML.....	24

Lesson 13:	HTML Layouts	27
13.1	Semantic Element.....	27
13.2	Non-semantic Elements	30
Lesson 14:	Form in HTML	31
14.1	Action	33
14.2	Method	33
14.3	Target	33
14.4	enctype	33
	Form Controls in HTML	33
14.5	Text input controls.....	33
14.6	Checkboxes Controls	34
14.7	Radio Button Controls	34
14.8	Select Box Controls.....	35
14.9	File Select boxes	35
14.10	Buttons Control.....	35
14.11	Hidden Form Controls.....	36
14.12	Submit and Reset Button	36
14.13	New input types in HTML 5.....	36
14.14	Attributes for Form validation.....	38
Lesson 15:	Multimedia in HTML 5	38
15.1	Audio	38
15.2	Video.....	39
15.3	Track.....	39
15.4	Embed	39
15.5	Object.....	40
15.6	Picture	40
Lesson 16:	THE HTML TAGS YOU NEVER LEARN.	40
Lesson 17:	Code Like Pro developer	48
17.1	Nesting.....	48
17.2	Siblings	48
17.3	Multiple Elements	48
17.4	Adding Classes and IDs	48
17.5	Custom Attributes	48
17.6	Grouping.....	48
17.7	Text Content Hyperlink	49
17.8	Complex Structure	49
17.9	Never do these Mistakes in HTML.....	49

Lesson 1: Introduction To HTML

HTML stands for hypertext markup language, which is developed by Tim Berners lee in 1990s at CERN (Council of European Organization for Nuclear Research), CERN is located in Geneva, Switzerland. HTML allows to format the text, sound, graphics and videos and save it in ASCII(American Standard Code for Information Interchange) format so that any computer can read. HTML is the most widely used language on web to develop web pages. HTML 1.0, HTML 2.0, HTML 3.0, HTML 4.0 and HTML 5.0 are the different version of it.

Understanding HTML is essential for everyone who is looking to build or design websites. The extension of HTML document is .HTML or .HTM

HTML comments

➤ Single line comment

<!-- write your comments here -->

➤ Multi-line comment

<!--
write your
Multi line comments here
-->

What does hypertext and markup mean?

- **Hypertext:** Hypertext means, text with a link embedded in it. If you click on that link, it will open a new webpage. Apart from text, hypertext may contain [HTML tables](#), [HTML lists](#), [HTML forms](#), [HTML images](#), etc.
- **Markup language:** Markup language uses tags to define elements within a document. It contains familiar words that are human-readable like [forms](#), [tables](#), [links](#), [titles](#), etc. Every tag in a markup language has a special meaning of its own and performs a particular operation.

Why HTML is not a programming language?

HTML is not a programming language because of the following reasons:

- No programming logic.
- No functions and variables.
- No If...else statement.
- No data modification.
- No mathematics.

Difference between HTML 4 and HTML 5

HTML 4	HTML 5
It was released in 1997	It is released in Oct. 2014
It supports Audio/Video but requires external plugins (like Flash).	It support <Audio> and <Video> directly and doesn't require any external plugin.
It contains Limited semantics elements.	It is rich in semantics elements like: <header>, <Footer>, <Nav>, <article>, etc.
Limited API Support.	Extensive APIs (offline storage, drag-drop, etc.)
It require Cookies for client-side Storage.	It uses Local storage, session storage.
It contains Basic form Controls.	Introduced some new form control like, calendar, date, time, color picker.

Doesn't focus more on document structure.	Strong focus on document structure.
---	-------------------------------------

1.1 Why to learn HTML?

Originally, HTML was developed with the intent of defining the structure of document like Headings, Paragraphs, list and so forth to facilitate the sharing of scientific information between researchers. Now, HTML is being widely used to format web pages with the help of different tags available in HTML language.

HTML is a must for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. Following are the reasons why we have to learn HTML:

- **Create Webpages:** You can create webpages or customize an existing web template if you know HTML well.
- **Become a Web Designer:** If you want to start your career as a professional web designer, then HTML and CSS is a must skill.
- **Understand web :** If you want to optimize your website, to boost its speed and performance, it is good to know HTML to get the best results.

Features of HTML

- HTML describes the structure of a webpage.
- It is platform independent, i.e., it works on all the operating systems.
- HTML is universally supported by all browsers. It is a standard markup language for website development.
- Human-readable HTML tags represent elements in a webpage. Hence, they are easy to remember.

Advantages of HTML

- It is super easy to learn even for anyone.
- The code is clean to write and follow.
- It is user friendly.
- It can easily integrate with other languages like PHP, Python, etc.
- The coding structure is very simple to understand
- It is free for the use which means you don't have to pay for anything.

Disadvantages of HTML

- A lot of code can be complex to handle.
- Sometimes there can be lots of pages that can be hard to handle.
- It is not secure on its own. You need third-party services to provide more security.
- Alone HTML does not have many capabilities. Html alone can create basic web pages nothing much.
- You have to write code for the simplest web pages. It can be a problem if a website has a lot of pages.
- Each page should be programmed separately. It is one of the biggest disadvantages.

Applications of HTML

- **Web pages development** - HTML is used to create pages which are rendered over the web. Almost every page of web is having html tags in it to render its details in browser.
- **Internet Navigation** - HTML provides tags which are used to navigate from one page to another and is heavily used in internet navigation.
- **Responsive UI** - HTML pages now-a-days works well on all platform, mobile, tabs, desktop or laptops owing to responsive design strategy.
- **Offline support** HTML pages once loaded can be made available offline on the machine without any need of internet.

HTML versions

- **HTML 1.0:** The first version of HTML was 1.0, which was the barebones version of HTML language, and it was released in 1993.
- **HTML 2.0:** This was the next version which was released in 24 November, 1995, and it was standard language version for website design. HTML 2.0 was able to support extra features such as form-based file upload, form elements such as text box, option button, etc.
- **HTML 3.2:** HTML 3.2 version was published by W3C in January 1997. This version was capable of creating tables and providing support for extra options for form elements. It can also support a web page with complex mathematical equations. It became an official standard for any browser till January 1997. Today it is practically supported by most of the browsers.
- **HTML 4.0:** Published in December 1997, and it included significant improvements over previous versions. It was reissued with minor edits in April 1998.
- **HTML 4.01:** HTML 4.01 version was released on December 1999, and it is a very stable version of HTML language. This version is the current official standard, and it provides added support for stylesheets (CSS) and scripting ability for various multimedia elements.
- **HTML5 :** HTML5 is the newest version of Hypertext Markup language. The first draft of this version was announced in January 2008. There are two major organizations one is W3C (World Wide Web Consortium), and another one is WHATWG(Web Hypertext Application Technology Working Group) which are involved in the development of HTML 5 version, and still, it is under development.

WHATWG stands for the **Web Hypertext Application Technology Working Group**. It is a community of people interested in evolving the web through standards and specifications. WHATWG was formed in 2004 by individuals from major browser vendors, including Apple, Mozilla, and Opera, to address the slow pace and perceived inadequacies of web standards development under the World Wide Web Consortium (W3C).

HTML Editors

Following are the HTML editor where we can write HTML Code:

- Sublime Text 3
- Notepad ++
- Komodo Edit
- HTML-Kit

- Coffee Cup
- VS-Code (Favourite of all programmers)

Some common terms

- i. WWW**
Full form of WWW is World Wide Web. WWW was originated in 1990s by Tim Burners Lee. It is subset of internet and it presents text, image, animation, video, sound and other multimedia in a single interface. *The world wide web is a system of interlinked Hypertext documents and multimedia content accessed via the internet.*
- ii. ISP**
ISP is an organization or company which provides internet service by taking certain charge. In Nepal, Mercantile, Everest net, World link, HONs etc are few examples of it.
- iii. Website**
A website is a collection of related web pages located under a single domain name, typically produced by a single person or organization.
- iv. Homepage**
The first page of the website is called Homepage. It is also known as landing page or index page. It is the main navigation point of a website. It contains summary and links of other pages.
- v. Webserver**
*A web server is a computer system that hosts websites, serving web pages to the users over the internet. All the web pages in the internet are stored in different web server, the web browser requests to the web server to get information. When the browser **request** to the web server, the server provides the proper **response** to the clients.*
- vi. URL**
A URL is the address used to access web resource on the internet. Exact location of an internet resource is called URL. each website of a webpage has unique address. Example: <https://www.example.com/page>. <https://www.google.com>, etc..
- vii. Web Browser**
The software programs that is used to display the contents of HTML is known as browser. It is also used to access the resource of the internet. Edge, Safari, Google chrome, Lynx, Mosaic, Opera mini, Mozilla Firefox, etc.. are different example of web browser.
- viii. Encryption**
The process of converting plain text into cipher text is known as encryption. it protect data confidentiality. It uses a key to encrypt the data. It is of two types: Symmetric and Asymmetric
- ix. Decryption**
Decryption is the process of converting cipher text into plain text. It uses a key to decode that encrypted data. it makes data readable after after secure transmission.
- x. Domain name**
A domain name is a human-readable address used to access a website. It is easier to remember than IP Address. .com, .org are top-level domain.
- xi. Server**

A server is a computer system that provides data, resources, or services to other computer (client computer) over a network. Server can be web server, email server, file server.

xii. Client

A computer or software application that access a service from the server is known as client computer. In networking, a client computer makes requests to a server which then processes the request and returns the desired information. web browser, email clients, FTP clients are example.

xiii. HTTP

The protocol used for transferring hypertext requests and information on the internet. It is the foundation of data communication for the world wide web. It uses the default port 80 or 8080.

xiv. HTTPS

It is used for secure communication over a computer network and is widely used on the internet. It encrypts the data exchanged between the client and server, providing protection against unauthorized access. It uses the port 443 by default. It uses SSL/TLS protocols for encryption. SSL(Secure Sockets Layer), TLS(Transport layer Security)

xv. Hosting

Hosting is the service of storing and maintaining files for one or more websites to be accessible on the internet. Shared, VPS, Dedicated, cloud hosting are the types of hosting. Bluehost, HostGator, Nest Nepal, Himalyan Host, Gurkha Host, Domain Racer, A2 hosting, etc. are hosting provider.

xvi. Protocol

Protocols are sets of rules or standards that define how data is transmitted and received over a network. HTTP/HTTPS, FTP, SMTP, IMAP/POP, TCP/IP, Telnet, etc.. are an example of protocol.

HTTP : Hypertext transfer protocol/ secure.

FTP: File Transfer Protocol

SMTP: Simple Mail Transfer Protocol.

IMAP / POP: Internet Message Access Protocol. / Post Office Protocol.

TCP/IP: Transmission Control Protocol./Internet Protocol.

Telnet: Telecommunication Network.

1.2 How Browser interprets HTML.

When you view HTML file in your web browser (google chrome, firebox, opera mini, edge, etc...) the browser parses the HTML code and uses it to dynamically render the visual page. HTML tells the browser what each piece of content is (a heading, image, paragraph, etc..) so that it can be displayed appropriately. Anytime you make a change to your HTML page structure, you'll need to refresh the page in the browser to see the updates.

Lesson 2: HTML Tag

HTML Tag are like keywords which defines that how web browsers will format and display the content. Or, HTML tags are the commands which are written inside the angular bracket. With the help of HTML tags, a web browser can differentiate between HTML content and simple content. tags are usually enclosed in angle brackets.

The opening tags begins with left angle bracket(<) followed with tag name and then right-angled bracket(>).

The closing tag is just like opening tag except that tag name start with forward slash (/).

Syntax of HTML Tag:

<tag> content </tag>

HTML tags contains three main parts:

- Opening tag 👉 <h1>
- Content 👉 welcome to day 1 of HTML
- Closing tag 👉 </h1>

HTML Tags are of 2 types:

- i. Container tag or pair tag or non-empty tag
- ii. Singular tag or empty tag or unpaired tag or self-closing tag.

2.1 Container tag & Singular tag

Those HTML tag which have both opening and closing is known as container tag. The content is the information or nested elements between the opening and closing tag.

For example:

```
<a href="www.facebook.com">Facebook </a>
```

The tag which consists of a single tag without a closing tag is known as empty tag. These tags are used for elements that don't have content. Unpaired tag often end with a forward slash before the closing angle bracket. But it also works where we left the forward slash.

Example:

```

```

```

```

Some common HTML tags

- a. <HTML> tag
- b. <Head> tag
- c. <Title> tag
- d. <Body> tag

a. <HTML> tag

This is non-empty tag. This tag defines top most elements and identifies it as HTML document.

Syntax:

```
<HTML>
```

```
.....
```

```
.....
```

```
.....
```

```
</html>
```

b. <Head> tag

This is also non-empty tag. This tag contains header section of HTML document. It is nested inside <HTML> tag.

Syntax:

```
<HTML>
```

```
  <Head>
```

```
.....
```

```

.....
.....
</Head>
.....
.....
.....
</html>

```

c. <Title> tag

This is also non-empty tag. This tag is used to place the text in the title bar of browser window. It is nested <Head> tag.

Syntax:

```

<HTML>
  <Head>
    <title>
      Text goes here
    </title>
  </Head>
  .....
  .....
  .....
</html>

```

d. <Body> tag

This is also non-empty tag. This tag contains main portion of HTML document.

Syntax:

```

<HTML>
  <Head>
    <title>
      Text goes here
    </title>
  </Head>
  <Body>
    .....
    .....
    .....
  </Body>
</html>

```

2.2 Basic Structure of HTML document

```

<!DOCTYPE html>    → specifies this is an HTML5 document.
<html>             → This is a root of an HTML page.
<head>             → Contains Page Metadata.
  <title>Code Plus</title>    → contains title
</head>
<body>             → The main body of the page (rendered by the browser)
  <h1>This is heading 1</h1>    → heading tag
  <p>My paragraph text goes there....</p>    → paragraph tag

```

`</body>` → *closing body tag*
`</html>`

2.3 HTML Attributes

Attributes contains additional piece of information about tag. attribute is written inside the opening tag.

Syntax:

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

Example:

``

In this instance, the image source (**src**) and the alt text (**alt**) are attributes of the `<img>` tag.

➤ Dir Attribute of HTML tag

The dir attributes allows you to indicate to the browser about the direction in which the text should flow either **Left to right** (the default value) or **Right to left** (for language such as Arabic, urdu). When dir attribute is used within the `<html>` tag, it determine how text will be presented within the entire document. When used within another tag, it control the text's direction for just the content of that tag.

➤ Lang attribute of HTML tag

The lang attribute allows you to indicate the main language used in a document, but this attribute was kept in `<HTML>` tag only for drawbacks compatibility with earlier version of HTML. This attribute has been replaced by the **Xml:lang** attribute in new XHTML (Extended Hyper Text Markup Language) document.

Attribute	Options	Function
Align	Right, left, center	Horizontally aligns tags.
Valign	Top, middle, bottom	Vertically aligns tags within an HTML element.
Bgcolor	Numeric, hexadecimal, RGB values	Places a background color behind an element.
Background	URL	Places a background image behind an element.
Id	User defined	Names an element for use with CSS.
Class	User defined	Classifies an element for use with CSS.
Width	Numeric value	Specifies the width of table, images or table cell.
Height	Numeric value	Specifies the height of tables, images or table cells.
Title	User defined	Pop-up title of the elements.

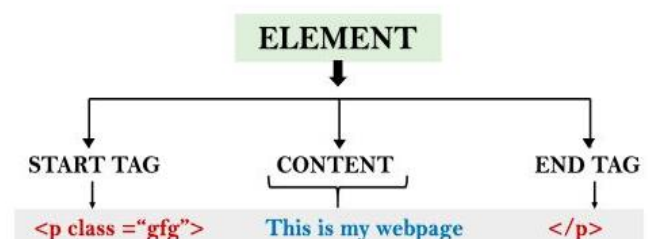
Lesson 3: Element in HTML

An element is a collection of start tag, attribute, end tag and content between them.

- To form an HTML element the start, tag and end tag have to be combined.
- Tags are used to mark the beginning and end of an element and to define the type of the element.

Example:

- `<p>` This is a paragraph `</p>`
- `<p>` is the starting tag `</p>` is the end tag.
- The content between the start and end tag is the actual content of the paragraph. So, in this case `<p>` and `</p>` together define the `<p>` element



3.1 Block Level Element

Block level elements take up the fill width available by default, starting on a new line and placed vertically.

Characteristics of Block level element

- Always start on a new line.
- Occupy the full width of their container (parent element).
- Can contain other block-level elements and inline elements.
- Their height and width can be controlled with CSS.

Example:

```
<div>
  <h1>This is a heading</h1>
  <p>This is a paragraph inside a div.</p>
</div>
```

In the above example, **<div>**, **<h1>**, and **<p>** are block-level elements. Each starts on a new line and takes up the full width of the page.

Some block-level elements

Elements	Description
<address>	Contact information for the author/owner of a document.
<article>	Independent, self-contained content.
<aside>	Content aside from the main content (like sidebars).
<blockquote>	Section quoted from another source.
<details>	Interactive disclosure widget.
<dialog>	Dialog box or window.
<div>	Generic container for flow content.
<dl>	Description list.
<dt>	Term/name in a description list.
<dd>	Description of a term/name in a description list.
<fieldset>	Grouping of related elements in a form.
<figcaption>	Caption for a <figure> element.
<figure>	Self-contained content, like illustrations.
<footer>	Footer for a section or page.
<form>	User-submittable form.
<h1> to <h6>	Headings, <h1> is the highest (most important) level & <h6> the lowest.
<header>	Introductory content for a section or page.
<hr>	Thematic break (horizontal rule).
	List item in , , or <menu>.
<main>	Main content of a document.
<nav>	Navigation links.
	Ordered list.
<p>	Paragraph.
<pre>	Preformatted text.
<section>	Thematic grouping of content.
<table>	Table.
	Unordered list.

3.2 Inline Level Element

Inline level elements do not start on a new line and only take up as much width as necessary.

Characteristics

- Do not start on a new line; they sit within the flow of the text.

- Only take up as much width as necessary.
- Cannot contain block-level elements, only other inline elements or text.
- Their width and height are dependent on the content.

Example:

<p>This is a strong and emphasized text.</p>

In the above example, **** and **** are inline elements that do not break the flow of the text within the <p> element.

Some Inline-level elements

Elements	Description
<a>	Hyperlink.
<abbr>	Abbreviation or acronym.
	Bold text.
<bdo>	Bidirectional override.
 	Line break.
<cite>	Citation.
<code>	Inline code.
<data>	Machine-readable content.
<dfn>	Definition term.
	Emphasized text.
<i>	Italic text.
	Image.
<input>	Input control.
<kbd>	User input (keyboard).
<label>	Label for a form control.
<mark>	Marked/highlighted text.
<meter>	Scalar measurement within a known range.
<output>	Result of a calculation or user action.
<progress>	Progress of a task.
<q>	Inline quotation.
<ruby>	Ruby annotation.
<s>	Strikethrough text.
<samp>	Sample output.
<select>	Drop-down list.
<small>	Smaller text.
	Generic inline container.
	Important text.
<sub>	Subscript text.
<sup>	Superscript text.
<textarea>	Multi-line text input.
<time>	Time value.
<u>	Underlined text.
<var>	Variable.

Difference between Block-level element and Inline-level element

Block-level element	Inline-level element
It always start on a new line.	It do not start on a new line.
It occupy the full available width	It only occupy the width that is required by their content.
It can contain both inline and block- level elements.	It can contain only inline elements or text.
Width and height can be set via CSS.	Width and height depend on content.
Example: <div>, <p>, <h1>, , <table>	Example: , <a>, , ,

3.3 Some Key Things to know:

- HTML Documents are made up of HTML elements created using tags.
- When a web browser reads an HTML document, browser reads It from top to bottom & left to right.
- HTML elements serves to label and apply meaning to content.
- Most elements consist of an opening tag and closing tag.
 - **Example:** <p> This is a paragraph element </p>
- The closing tag includes a forward slash.
- There are also singular elements like images, line break, horizontal line, that don't need closing tag.

Lesson 4: Mastering Text content with HTML

Once you understand the basics of HTML structure and syntax, it's time to dive into text content! Text is at the core of webpages.

4.1 Heading

- HTML Provides six separate heading tag **h1 – h6**
- **H1** is the most important tag where as **H6** is the least important tag.
- **H1** tag has the largest and thickest text while **H6** is the smallest and thinnest.
- Each heading has a default size. However, we can change the size of any heading size with the help of CSS font-size property.

Some tips

- There should be only one **h1** tag in each page.
- Start with **h1**, then subsection with **h2**, down to lower subsections.
- Search engine place more importance on higher heading level.

Example:

```
<h1> My HOMEPAGE </h1>
<h2 style="font-size:60px;" > About Us </h2>
<h3> Contact Us </h3>
```

4.2 Paragraph

Paragraph are the most basic HTML element it allow you to organize text into blocks and paragraphs.

Example:

```
<p> Here is a paragraph of text on my page </p>
```

Lesson 5: Text Formatting.

While Most HTML tags are used to create elements, HTML also provides in-text formatting tags to apply specific text-related styles to portions of text.

Here we'll cover HTML Text Formatting such as **highlighting**, **bold**, **underline**, **super-script**, **sub-script**, and **strike-through** text.

There are so many ways to formatting text in HTML but mainly two ways of formatting text in HTML are:

- a. Preserved Formatting
- b. Inline and Block-level formatting

a. Preserved Formatting

This method involves using tags that maintain the whitespace and line breaks as they are written in the HTML source. The primary tag used for preserved formatting is the `<pre>` tag.

b. Inline and Block-level Formatting

5.1 Bold

Tags like **strong** & **B** allow formatting inside paragraphs and headings for bold.

Example:

`<strong>` this text is bold `</strong>`

`<b>` This is the use of b tag to make text bold `</b>`

What is the difference in strong and B tag?

Strong is used to indicate that the text is fundamentally or semantically important to the surrounding text, while `<b>` indicates no such importance and simply represents text that should be bolded.

If you use `<b>` in a text-to-speech program would not say the word any differently than any other words around it. But if you use `<strong>` to the same program, then it'll speak those words in a different way with different tone of voice to convey that the text is important in some way.

5.2 Highlight

The `<mark>` element is new in HTML 5 & it is used to mark or highlight text in a document.

Example:

`<p>` Here is some content from an article that contains the `<mark>` search results `</mark>` that we are looking for. `</p>`

5.3 Italic

To italicize the text , use the `<em>` or `<i>` tags

Example:

`<em>` This is italicized text using em tag. `</em>`

`<i>` This text is italicized using I tag. `</i>`

5.4 Underline

The `<u>` tag itself was deprecated in HTML 4, but it was reintroduced with alternate semantic meaning in HTML 5 to represent an unarticulated, non-textual annotation. **It is used to show misspelled text on the page.**

Example

<p> This paragraph contains some <u> misspelled </u> text. </p>

5.5 Inserted, deleted, or strike-through

To make text as inserted, use the <ins> tag

Example:

<ins>New txt</ins>

To make text as deleted, use the <del> tag

Example:

~~Deleted Text~~

To strike through text, use the <s> tag

Example:

<s>Strike through </s>

Example:

~~Original Price: 5000~~

<ins style="text-decoration: none;">New price: 999</ins>

<s>Strike through</s>

5.6 Super-script & Sub-script

To offset the text either upward or downward you can use the tag <sup> and <sub>.

To create super script

<p> Power of 2 ⁰ </p>

To create sub script

<p>Molecular formula of Water is H ₂ O </p>

5.7 Line Break Tag (
 tag)

Whenever you use the
 tag, anything following it starts from the next line. This tag is an example of an empty tag, where you do not need opening and closing tags.

The
 tag has a space between the characters **br** and the forward slash. If you omit this space, older browser will have trouble rendering the line break, while if you miss the forward slash character and just use
 it is not valid in XHTML.

Example:

<p> First Sentence
 New line inserted here! </P>

5.8 Horizontal line (<Hr> tag)

Horizontal lines are used to visually break-up sections of a document. The <hr> tag creates a line from the current position in the document to the right margin and breaks the line accordingly. Again, <hr /> tag is an example of the empty element, where you do not need opening and closing tags, as there is nothing to go in between them. The <hr> element has a space between the characters **hr** and the forward slash.

5.9 Centering content

You can use <center> tag to put any content in the center of the page or any table cell.

5.10 Monospaced Font

- The content of a <tt> </tt> element is written in monospaced font.
- Most of the fonts are known as variable-width fonts because different letter of different widths (for example, the letter 'M' is wider than letter 'i').

- In a monospaced font, each letter has the same width.
- This element has been deprecated and the `<code>` element is an appropriate modern replacement for `<tt>`.

5.11 Larger Text / Smaller text

- The content of the `<big> .... </big>` element is displayed one font size larger than the rest of the text surrounding it.
- This tag is depreciated in HTML 5 so use CSS to make your text bigger and use the `font-size:1.3em`.
- The content of the `<small> .... </small>` element is displayed one font size smaller than the rest of the text surrounding it.

5.12 Highlight

The `<mark>` element is new in HTML 5 & it is used to mark or highlight text in a document.

Example; Here is some content from an article that contains the `<mark>` search results `</mark>` that we are looking for `</P>`

Lesson 6: HTML Link

In HTML, link are a fundamental aspect of creating interactive and navigable web pages. links are also known as hyperlinks, allows user to click and navigate from one page to another, whether within same website or to an external site. Links can also direct users to different sections within the same page, initiate email communication, or trigger phone calls on mobile devices. When you move the mouse over a link, the mouse arrow will turn into a little hand.

Syntax:

`<a href="URL" > Link Text </a>`

- **<a>**: This is the anchor tag, which is used to define the link.
- **href**: This attribute specifies the URL or path to which the link points.
- **Link Text**: The text between the opening and closing anchor tags that users see and can click on.

Types of Links

a. External Link

These links point to pages on different websites.

```
<a href="https://www.example.com">Visit Example</a>
```

This links directs users to the example website.

b. Internal Link

These links points to different pages within the same website.

```
<a href="about.html">About Us</a>
```

This link navigate to the “about.html” page within the same website.

Example:

```
<a href="#section1">Go to Section 1</a>
```

```
<h2 id="section1">Section 1</h2>
```

Additional Attributes of Link

target: it specifies where to open the linked document. target attribute can have one of the following values:

- **_self (default)**: Opens the link in the same tab.
- **_blank**: Opens the link in a new tab or window.
- **_parent**: Opens the link in the parent frame.
- **_top**: Opens the link in the full body of the window.

Example:

```
<a href="https://www.example.com" target="_blank">Visit Example in New Tab</a>
```

Title: Provides additional information about the link, which is displayed as a tooltip when the user hovers over the link.

```
<a href="https://www.example.com" title="Visit Example Website">Visit Example</a>
```

Rel: Specifies the relationship between the current document and the linked document. Common values include nofollow, noopener, and noreferrer.

```
<a href="https://www.example.com" rel="nofollow">Visit Example</a>
```

Styling Link with CSS

You can style links using CSS to change their appearance.

Example:

```
<style>
a {
  color: blue; /* Default link color */
  text-decoration: none; /* Remove underline */
}

a:hover {
  color: red; /* Color when hovering */
}
```

```
</style>
```

Use an Image as a link

To use an image as a link, just put the tag inside the <a> tag.

Example:

```
<a href="#">

</a>
```

Button as a link

To use an HTML button as a link, you have to add some JavaScript code. JavaScript allows you to specify what happens at certain events, such as a click of a button:

Example

```
<button onclick="#">HTML Tutorial</button>
```

Lesson 7: HTML Images

HTML images are defined with the tag. The source file (src), alternative text (alt), width and height are provided attributes.

Syntax:

```

```

URL are of two types

a. Absolute URL

Links to an external image that is hosted on another. External images might be under copyright. If you do not get permission to use it, you may be in violation of copyright laws. In addition, you cannot control external images; it can suddenly be removed or changed.

b. Relative URL

Links to an image that is hosted within the same website. Here the URL doesn't include the domain name. if the URL begins without a slash, it will be relative to the current page.

Lesson 8: HTML Color

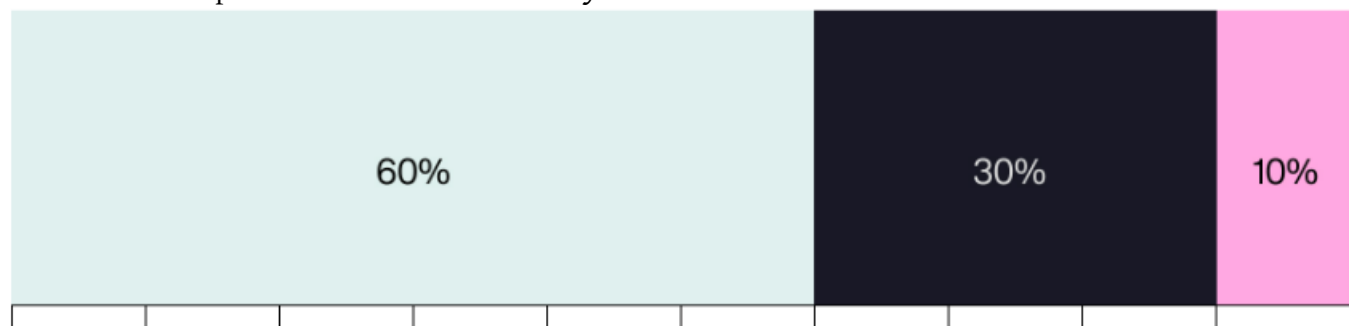
Colors are used to enhance the visual appearance of web pages and to present the information more effectively.

Methods to use colors in HTML

- i. **Color Name**
Html supports 140 color names.
Example: red, green, black, etc...
`<p style="color: red;">This is a red text.</p>`
- ii. **Hex code color**
Colors can be defined using a six-digit hexa-decimal number prefix by a #
Example: #000000 for black, #FFFFFF (for white)
`<p style="color: #FF5733;">This is a text with a hex color.</p>`
- iii. **RGB (Red, Green, Blue) value**
Colors can be defined using `rgb()` function with value between 0 to 255
Example: `rgb(255, 0, 0)` for red, `rgb(0, 128, 0)` for green.
`<p style="color: rgb(0, 128, 0);">This is a green text.</p>`
- iv. **RGBA (Red, Green, Blue, Alpha) value**
Similar to RGB but includes an alpha value for transparency (0.0 to 1.0)
Example: `rgba(255, 0, 0, 0.5)` for semi-transparent red..
`<p style="color: rgba(255, 0, 0, 0.5);">This is a semi-transparent red text.</p>`
- v. **HSL (Hue, Saturation, Lightness) value**
Colors can be defined using the `hsl()` function
Example: `hsl(120, 100%, 50%)` for green
`<p style="color: hsl(120, 100%, 50%);">This is a green text.</p>`
- vi. **HSLA (Hue, Saturation, Lightness, Alpha) Value**
Similar to HSL but includes an alpha value for transparency.
Example: `hsla(120, 100%, 50%, 0.5)` for semi-transparent green.
`<p style="color: hsla(120, 100%, 50%, 0.5);">This is a semi-transparent green text.</p>`

6:3:1 color rule

While using the 6:3:1 Rule, designers have to choose a dominant color and use it in 60% of the space, a secondary color in 30% and a final color in the remaining 10%. The 6:3:1 rule eases the eye of users to move from one point to another comfortably.



- | | | |
|---|--|--|
| <div style="display: inline-block; width: 15px; height: 15px; background-color: #ADD8E6; margin-right: 5px;"></div> Main color
The main color will be used about 60% throughout the design. Choose one that is neutral and is easy on the eyes. | <div style="display: inline-block; width: 15px; height: 15px; background-color: #000080; margin-right: 5px;"></div> Secondary color
The secondary color will be used in 30% of the design. It should contrast with the main color to create visual interest. | <div style="display: inline-block; width: 15px; height: 15px; background-color: #FF00FF; margin-right: 5px;"></div> Accent color
The accent color will be used in 10% of the design, and should contrast well with the other colors. |
|---|--|--|

- **60% - primary color**
This color serves as the main backdrop and sets the overall tone of the design. In web design, this color is often used for backgrounds, large sections, and significant area of the site.
- **30% - Secondary Color**
This color supports and complements the primary color. It is used for 30% of the design, adding variety and visual interest. This color can be applied to smaller areas like sidebars, backgrounds of sections.

➤ 10% - Accent color

This color is used sparingly(काम परेको खण्डमा), making up about 10% of the design. It is used to get attention to key elements such as buttons, links, and other interactive features. This color should stand out against primary and secondary color, providing a clear visual interest.

Example of 6:3:1 rule:

Imagine a website for a nature blog:

- **60% primary Color:** A calming green used for the background.
- **30% Secondary Color:** A complementary brown used for headers, sidebars, and background of certain sections.
- **10% Accent Color:** A bright orange used for buttons, links, and call-to-action elements.

```
<style>
body {
  background-color: #A8D5BA; /* Dominant color: soft green */
  color: #333;
}
header, footer {
  background-color: #8B5A2B; /* Secondary color: brown */
}
a, button {
  color: #FF7043; /* Accent color: bright orange */
}
</style>
```

Lesson 9: HTML List

A list is a collection of related items grouped together in a specific structure. List help to organize content in a logical and readable format, making it easier for user to understand and navigate the information presented on a web page. HTML provides 3 types of lists: Ordered list, unordered list, and description list.

9.1 Ordered List ()

An ordered list display item in a sequentially numbered format, indicating a specific order or ranking of the items. By default, the list item will be marked with number.

Syntax

```
<ol type="A" start="3">
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
</ol>
```

Attributes:

- **type:** Specifies the numbering type (1, A, a, I, i).
- **start:** Sets the starting value of the list.
- **reversed:** Reverses the numbering order.

9.2 Unordered List ()

An unordered list displays the items with bullet points, indicating that the order of items is not important. By default, unordered list items will be marked with small black circles called bullet.

Syntax

```
<ul type="square" >
  <li>First item</li>
  <li>Second item</li>
```

```
</li>Third item</li>
</ul>
```

Attributes

- **type**: Specifies the bullet type (disc, circle, square).

9.3 Description List (<dl>)

A description list is used to pair items with their description, similar to a dictionary or glossary.

Syntax

```
<dl>
  <dt>HTML</dt>
  <dd>Hyper Text Markup Language</dd>
  <dt>CSS</dt>
  <dd>Cascading Style Sheets</dd>
</dl>
```

Tags

- **<dt>**: Defines a term in the description list.
- **<dd>**: Describes the term.

9.4 Nested List

HTML also supports nested lists, where a list item can contain another list. This is useful for creating complex list structures such as outlines.

Example:

```
<ul>
  <li>Item 1
    <ul>
      <li>Subitem 1.1</li>
      <li>Subitem 1.2</li>
    </ul>
  </li>
  <li>Item 2</li>
</ul>
```

Styling list with CSS

Lists can be styled using CSS to change the appearance of the list items, bullet points and numbers.

- **Changing bullet style**

```
ul {
  list-style-type: square;
}
```

- **Custom list item markers**

```
ul {
  list-style-image: url('marker.png');
}
```

Lesson 10: Table in HTML

Table allows web authors to arrange data like text, images, links, and other table, etc.. into rows and columns of cells. Table is a way to present data in a structured format of rows and columns. They are like spreadsheets but designed for displaying information on webpages.

A table is defined using the **<table>** element. Within this element, various tags are used to defined the structure of the table:

Tags	Description
------	-------------

<table>	This tag defines the beginning and end of a table.
<thead>	This optional tag groups the header content in the table.
<tbody>	This optional tag groups the body content in the table.
<tfoot>	This optional tag groups the footer content in the table.
<tr>	This tag defines a table row.
<th>	This tag defines a header cell within a table row.
<td>	This tag defines a standard data cell within a table row.

Basic structure

<table>: defines the start and end of the table.

<tr> : defines a table row.

<td> : defines a table data cell in row (for regular content).

<th> : defines a table header cell (for Headings).

Additional Elements

<Caption>: adds a caption above the table.

<thead> : defines the header section of the table (optional).

<tbody> : defines the main body of the table (optional).

<tfoot> : defines the footer section of the table (optional).

<colgroup>: groups the columns together of the table and can be used for styling.

Example:

```
<colgroup>
    <col span="2" style="background-color: #D6EEEE">
</colgroup>
```

Table Attributes

Border	defines the border thickness of the table.
Width	define the width of the table.
Cellpadding	defines the space between the cell content and its border.
Border-collapse	this value collapses the border between adjacent cells into a single border (CSS).
Align	defines the alignment of the table content.
Cellspacing	used to specify the space between cells to avoid them merging visually.
Colspan	defines the number of columns a table cell should span horizontally.
Rowspan	defines the number of rows a table cell should span vertically.
Caption	defines title for the table.
Background	used to set a background image for the entire table.

Example of an HTML Table

```
<!DOCTYPE html>
<html>
<head>
    <title>HTML Table Example</title>
    <style>
        table {
            width: 100%;
            border-collapse: collapse;
        }
        th, td {
            border: 1px solid #ddd;
            padding: 8px;
        }
        th {
```

```

        background-color: #f2f2f2;
        text-align: left;
    }
    tr:nth-child(even) {
        background-color: #f9f9f9;
    }
    tr:hover {
        background-color: #ddd;
    }
</style>
</head>
<body>
<table>
<thead>
<tr>
<th>First Name</th>
<th>Last Name</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Doe</td>
<td>25</td>
</tr>
<tr>
<td>Jane</td>
<td>Smith</td>
<td>30</td>
</tr>
</tbody>
<tfoot>
<tr>
<td colspan="3">Table Footer</td>
</tr>
</tfoot>
</table>
</body>
</html>

```

Lesson 11: Marquee In HTML

The <marquee> element in HTML is used to create a scrolling piece of text or image horizontally or vertically across the web page. Marquee element is depreciated and non-standard in HTML5. It means it is not recommended for use in modern web development as it is not part of the w3c HTML5 and may not supported by all browser.

Despite being depreciated, the <marquee> element is still understood by many browsers, which means it can still be used, but it is advisable to use CSS and JavaScript for creating similar effect for better control and compatibility.

Attributes of <marquee>

- a. **Direction:** specifies the direction of the scrolling content. Values can be “left, right, up and down”.
Example: <marquee direction="left">This text will scroll from right to left</marquee>
- b. **Behavior:** Defines the behavior of the marquee. Values can be “scroll, slide and alternate”.
Example: <marquee behavior="alternate">This text will alternate</marquee>
- c. **Scrollamount:** Sets the speed of the scrolling. Default value is 6.
Example: <marquee scrollamount="10">This text will scroll faster</marquee>
- d. **Scrolldelay:** sets the delay between each scroll movement in milliseconds. The minimum value is 60 for a smooth scroll.
Example: <marquee scrolldelay="200">This text will scroll with a delay</marquee>
- e. **Loop:** Defines the number of times the marquee will scroll. The default value is -1, which means it will scroll infinite times.
Example: <marquee loop="3">This text will scroll three times</marquee>
- f. **Bgcolor:** sets the background color of the marquee.
Example: <marquee bgcolor="yellow">This text has a yellow background</marquee>
- g. **hspace:** This specifies horizontal space around the marquee. This can be a value like 10 px or 20%, etc.
- h. **vspace:** This specifies vertical space around the marquee. This can be a value like 10px or 20%, etc.

Example of Using Marquee

```
<!DOCTYPE html>
<html>
<head>
  <title>Marquee Example</title>
</head>
<body>
  <marquee behavior="scroll" direction="left" scrollamount="5" hspace="30%">
    This is a scrolling text using the marquee tag.
  </marquee>
  <marquee direction="up" bgcolor="lightblue" vspace="10%">
    This text moves up and down.
  </marquee>
  <marquee behavior="slide" direction="right" loop="3">
    This text slides from left to right three times.
  </marquee>

</body>
</html>
```

Modern Alternative of marquee using CSS

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      .marquee {
        width: 100%;
        overflow: hidden;
```

```

        white-space: nowrap;
        box-sizing: border-box;
        animation: marquee 15s linear infinite;
    }

    @keyframes marquee {
        0% {
            transform: translateX(100%);
        }
        100% {
            transform: translateX(-100%);
        }
    }
</style>
</head>
<body>

    <div class="marquee">This is a scrolling text using CSS animations.</div>

</body>
</html>

```

Lesson 12: Frames in HTML

Frames were traditionally used in HTML to divide the browser window into multiple sections where each section could load a separate HTML document. A collection of frames in the browser window is known as a frameset. The window is divided into frames in a similar way the tables are organized into rows and columns. The primary tags used for frames included **<frame>** , **<frameset>** , **<noframes>** . however, frames have been depreciated and may not supported in HTML 5. There are 3 types of frames supported in HTML.

a. Inline Frame

Inline frame is commonly known as **iframe**, which are used to embed another HTML document within the current HTML document. It allows for embedding content from different domains. It has attribute like “**src, height, width and frameborder**”

Example: `<iframe src="https://www.example.com" width="600" height="400" title="description"></iframe>`

Tip: It is a good practice to always include a title attribute for the <iframe>. This is used by screen readers to read out what the content of the iframe is.

b. Borderless Frame

A borderless frame refers to a frame (usually an inline frame) without visible border. It allows for embedding content from different domains.

Example: `<iframe src="frame.html" frameborder="0"></iframe>`

c. Frameset Frame

Frameset frames are used within a **<frameset>** elements to define individual sections (frames) of a webpage. Each frame display a separate HTML document.

Example:

```

<frameset cols="25%, 75%">
    <frame src="frame1.html">

```

```
<frame src="frame2.html">
</frameset>
```

Frameset attributes

1. Cols
2. Rows
3. Border
4. Frameborder
5. Frame-spacing

Frame tag attributes

1. Src
2. Name
3. Frameborder
4. Margin--width
5. Margin-height
6. No-resize
7. Scrolling
8. Long-desc

Reasons for deprecation of frame:

- a. SEO challenges
- b. Linking issues
- c. Printing Issues
- d. Bookmarking problems
- e. Usability Issues

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
  iframe {
    width: 100%;
    height: 500px;
    border: 1px solid black;
  }
</style>
</head>
<body>

  <iframe src="https://www.codeplus.np.com"></iframe>

</body>
</html>
```

Modern Alternatives to frames

HTML 5 encourages the use of CSS for layout management and JavaScript for dynamic content manipulation. Modern alternative uses **CSS Flexbox** and **Grid layout**.

Example of Flex-Layout:

```
<!DOCTYPE html>
```

```

<html>
<head>
  <style>
    .container {
      display: flex;
      flex-direction: row;
    }
    .frame {
      flex: 1;
      border: 1px solid black;
      margin: 5px;
      padding: 10px;
    }
  </style>
</head>
<body>

  <div class="container">
    <div class="frame">Content 1</div>
    <div class="frame">Content 2</div>
    <div class="frame">Content 3</div>
  </div>

</body>
</html>

```

Example of Grid-Layout

```

<!DOCTYPE html>
<html>
<head>
  <style>
    .grid-container {
      display: grid;
      grid-template-columns: 1fr 2fr 1fr;
      gap: 10px;
    }
    .grid-item {
      border: 1px solid black;
      padding: 20px;
      text-align: center;
    }
  </style>
</head>
<body>

  <div class="grid-container">
    <div class="grid-item">Header</div>

```

```

        <div class="grid-item">Menu</div>
        <div class="grid-item">Main Content</div>
        <div class="grid-item">Footer</div>
    </div>
</body>
</html>

```

Iframe - height & width using CSS: You can add the style attribute and use the CSS height and width.
<iframe src="demo_iframe.html" style="height:200px;width:300px;" title="IframeExample"></iframe>

Iframe - Remove the Border: By default, an iframe has a border around it. To remove the border, add the style attribute and use the CSS border property:

```

<iframe src="demo_iframe.htm" style="border: none;" title="Iframe Example"></iframe>
    ➤ With CSS, you can also change the size, style and color of the iframe's border:
<iframe src="demo_iframe.htm" style="border:2px solid red;" title="Iframe Example"></iframe>

```

Iframe - Target for a Link: An iframe can be used as the target frame for a link. The **target** attribute of the link must refer to the **name** attribute of the iframe:

```

<iframe src="demo_iframe.html" name="iframe_a" title="Iframe Example"></iframe>
<p><a href="https://www.w3schools.com" target="iframe_a">W3Schools.com</a></p>

```

Lesson 13: HTML Layouts

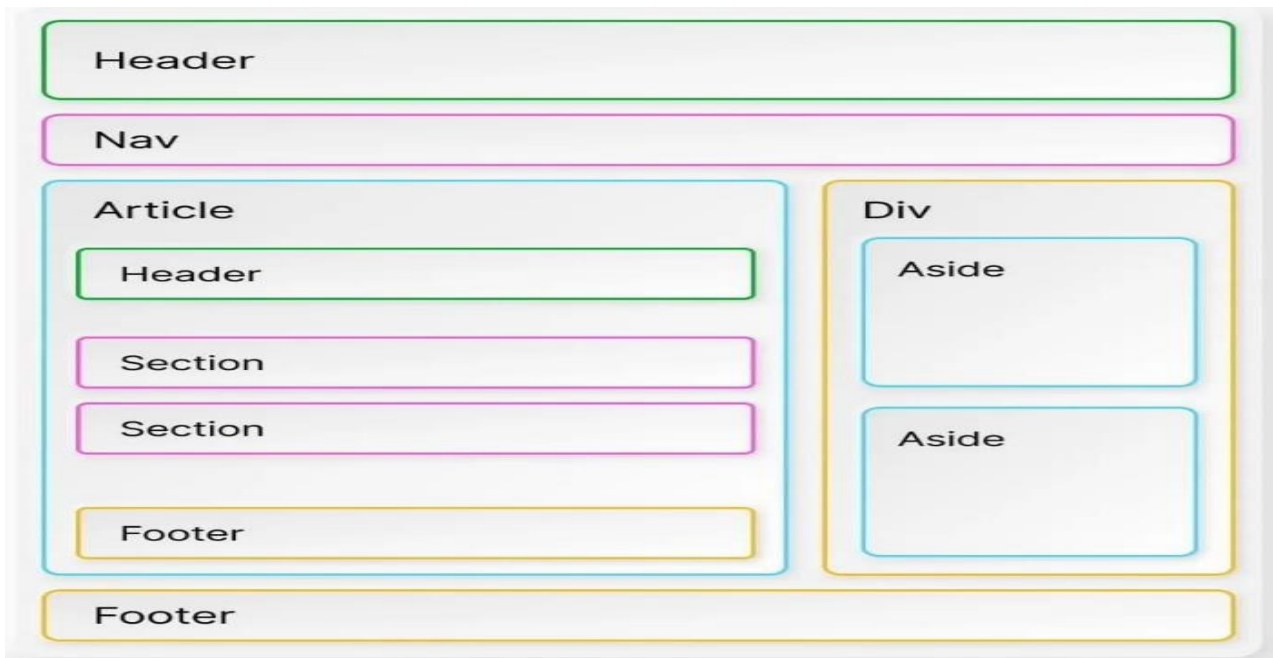
A webpage layout is very important to give better look to your website. It takes considerable time to design a website's layout with great look and feel. Now a days, all modern website are using CSS and JavaScript based framework to come up with responsive and dynamic websites, but you can create a good layout using HTML **semantic** and **non-semantic elements**.

13.1 Semantic Element

Semantic elements in HTML are those that convey meaning about the content they contain, making it clear to both browser and developers what the purpose or role of that content is within the document. Semantic elements provide valuable information for accessibility, search engine optimization (SEO), and code readability.

Following semantic elements define the different parts of a webpage.

Elements	Description
<header>	Defines a header for a document.
<nav>	Defines a set of navigation links
<section>	Defines a section in a document
<article>	Defines an independent, self contained content
<aside>	Defines content aside from the content (like a sidebar)
<footer>	Defines a footer for a document or a section
<details>	Defines additional details that the user can open and close on demand
<summary>	Defines a heading for the <details> element.



Example of HTML Layout using semantic elements

```
<!DOCTYPE html>
<html>
<head>
  <!-- <link rel="stylesheet" href="style.css"> -->
  <style>
    * {
      margin: 10px;
      padding-top: 10px;
    }

    html {
      margin: 0;
      padding: 0;
    }

    body {
      font-family: 'Poppins', sans-serif;
      color: springgreen;
    }

    .container {
      text-align: center;
      margin: 0;
      padding: 0;
    }

    nav {
      background-color: #333;
      padding-bottom: 5vh;
      margin-top: 0;
    }

    header {
```

```

background-color: #666;
padding-bottom: 20vh;
}

main {
background-color: #888;
display: flex;
}

section {
background-color: #999;
width: 65%;
margin: 3%;
}

article {
background-color: #777;
padding-bottom: 10vh;
margin: 10%;
}
aside {
background-color: #999;
width: 35%;
margin: 3%;
}
footer {
background-color: #333;
padding-bottom: 5vh;
margin-bottom: 0;
}
</style>
<title>HTML Layout</title>
</head>

<body>
<div class="container">

<nav>nav</nav>

<header>header</header>

<main>
<section>
section
<article>article</article>
</section>
<aside>aside</aside>
</main>

<footer>footer</footer>
</div>
</body>
</html>

```

13.2 Non-semantic Elements

Non-semantic elements are generic tags that do not convey any specific meaning about the content they contain. They are used primarily for structural purpose and can be styled and positioned using CSS. The most common non-semantic elements are “<div>”, and “”

<div>:

- Block-level element.
- Used to group other block-level and inline elements.
- Often used for layout purposes, such as creating sections of a page.

- Inline element.
- Used to group text or other inline elements.
- Useful for styling parts of text within a block-level element.

Basic Layout Using Non-Semantic Elements

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Non-Semantic Layout Example</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
    }
    .container {
      width: 80%;
      margin: auto;
      overflow: hidden;
    }
    .header, .nav, .main, .aside, .footer {
      padding: 20px;
      margin: 10px;
      border: 1px solid #ddd;
    }
    .header {
      background: #f4f4f4;
    }
    .nav {
      background: #ccc;
    }
    .main {
      background: #e4e4e4;
    }
    .aside {
      background: #ddd;
    }
    .footer {
```

```

        background: #f4f4f4;
    }
    .nav ul {
        list-style: none;
        padding: 0;
    }
    .nav ul li {
        display: inline;
        margin-right: 10px;
    }
</style>
</head>
<body>
    <div class="container">
        <div class="header">
            <h1>Website Header</h1>
        </div>
        <div class="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>
        </div>
        <div class="main">
            <div class="article">
                <h2>Main Article</h2>
                <p>This is the main content of the page.</p>
            </div>
        </div>
        <div class="aside">
            <h2>Related</h2>
            <p>This is some additional information.</p>
        </div>
        <div class="footer">
            <p>Website Footer</p>
        </div>
    </div>
</body>
</html>

```

Non-semantic HTML elements offer flexibility for layout and design, especially when combined with CSS, but using semantic HTML elements whenever possible is recommended to improve the structure, meaning, and accessibility of web content.

Lesson 14: Form in HTML

Forms are used to collect user input. it allows users to submit data to the server for processing. Forms are required, when you want to collect some data from the site visitor. For example: during user registration you can collect information information such as name, email, phone no, username, password, credit card, etc. from the end user.

A form will take input from the site visitor and then it will post it to a backend application such CGI, ASP script or PHP Script etc. The backend application will perform required processing in the passed data based on defined business logic inside the application.

Syntax:

The HTML <form> tag is used to create an HTML form and it has following syntax.

<form action = "script url" method = "GET || POST"> form elements like input, textarea, etc. </form>

Example:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>HTML Form Example</title>
  </head>
  <body>
    <form action="/submit" method="post">
      <label for="firstname">First Name:</label>
      <input type="text" id="firstname" name="firstname" required><br><br>

      <label for="lastname">Last Name:</label>
      <input type="text" id="lastname" name="lastname" required><br><br>

      <label for="email">Email:</label>
      <input type="email" id="email" name="email" required><br><br>

      <label for="password">Password:</label>
      <input type="password" id="password" name="password" required><br><br>

      <label for="birthdate">Birthdate:</label>
      <input type="date" id="birthdate" name="birthdate"><br><br>

      <label for="color">Favorite Color:</label>
      <input type="color" id="color" name="color"><br><br>

      <label for="range">Range:</label>
      <input type="range" id="range" name="range" min="0" max="100"><br><br>

      <label for="gender">Gender:</label>
      <select id="gender" name="gender">
        <option value="male">Male</option>
        <option value="female">Female</option>
        <option value="other">Other</option>
      </select><br><br>

      <label for="bio">Bio:</label><br>
      <textarea id="bio" name="bio" rows="4" cols="50"></textarea><br><br>
```

```

        <input type="submit" value="Submit">
        <input type="reset" value="Reset">
    </form>
</body>
</html>

```

Form Attributes

14.1 Action

The URL where the form data will be sent.

14.2 Method

The HTTP method is used to upload data to the server. The most frequently used are GET and POST methods.

14.3 Target

It specifies the target window or frame where the result of the script will be displayed. It takes values like `_blank`, `_self`, `_parent`, etc.

14.4 enctype

You can use the `enctype` attribute to specify how the browser encodes the data before it sends it to the servers. Possible values are:

- **application/x-www-form-urlencoded** : This is the standard method most forms use in simple scenarios.
- **Multipart/form-data** : This is used when you want to upload binary data in the form of files like image, word file, pdf file, etc.

Form Controls in HTML

- Text Input Controls
- Checkboxes Controls
- Radio Box Controls
- Select Box Controls
- File Select boxes
- Hidden Controls
- Clickable Buttons
- Submit and Reset Button

14.5 Text input controls

- a. **Single-line text input controls** - This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML `<input>` tag.

Example: `<input type="text" name="username" value="John Doe" size="30" maxlength="50">`

Attributes of text input control

- **type:** Indicates the type of input control and for text input control it will be set to **text**.
- **name:** Used to give a name to the control which is sent to the server to be recognized and get the value.
- **Value** : This can be used to provide an initial value inside the control.
- **size:** Allows to specify the width of the text-input control in terms of characters.
- **maxlength** : Allows to specify the maximum number of characters a user can enter into the text box.

- b. **Password input controls** – This is also a single-line text input, but it masks the character as soon as a user enters it. They are also created using HTML `<input>` tag but type attribute is set to **password**.

Example: `<input type="password" name="password" size="30" maxlength="50">`

Attributes of password input control

- **type:** Indicates the type of input control and for password input control it will be set to **password**.
- **name:** Used to give a name to the control which is sent to the server to be recognized and get the value.
- **value:** This can be used to provide an initial value inside the control.
- **size:** Allows to specify the width of the text-input control in terms of characters.
- **maxlength:** Allows to specify the maximum number of characters a user can enter into the text box.

- c. **Multi-line text input controls** – This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML `<textarea>` tag.

Example: `<textarea name="comments" rows="5" cols="40"></textarea>`

Attributes of multi-line text input control

- **name:** Used to give a name to the control which is sent to the server to be recognized and get the value.
- **rows:** Indicates the number of rows of text area box.
- **Cols:** Indicates the number of columns of text area box

14.6 Checkboxes Controls

Checkboxes are used when more than one option is required to be selected. They are also created using HTML `<input>` tag but type attribute is set to **checkbox**.

Example: `<input type="checkbox" name="subscribe" value="newsletter" checked>`

Attributes of checkbox control

- **type:** Indicates the type of input control and for checkbox input control it will be set to **checkbox**.
- **name:** Used to give a name to the control which is sent to the server to be recognized and get the value.
- **value:** The value that will be used if the checkbox is selected.
- **checked:** Set to *checked* if you want to select it by default.

14.7 Radio Button Controls

Radio buttons are used when out of many options, just one option is required to be selected. They are also created using HTML `<input>` tag but type attribute is set to **radio**.

Example:

`<input type="radio" name="gender" value="male" checked> Male`
`<input type="radio" name="gender" value="female"> Female`

Attributes of radio button control

- **type:** Indicates the type of input control and for checkbox input control it will be set to **Radio**.
- **name:** Used to give a name to the control which is sent to the server to be recognized and get the value.
- **value:** The value that will be used if the checkbox is selected.
- **checked:** Set to *checked* if you want to select it by default.

14.8 Select Box Controls

A select box, also called drop down box which provides option to list down various options in the form of drop-down list, from where a user can select one or more options.

Example:

```
<select name="cars" multiple size="3">
  <option value="volvo" selected>Volvo</option>
  <option value="saab">Saab</option>
  <option value="mercedes">Mercedes</option>
  <option value="audi">Audi</option>
</select>
```

Attributes of select box control

- **name:** Used to give a name to the control which is sent to the server to be recognized and get the value.
- **size:** This can be used to present a scrolling list box.
- **multiple:** If set to "multiple" then allows a user to select multiple items from the menu

<option> tag attribute

- **value:** The value that will be used if an option in the select box box is selected.
- **selected:** Specifies that this option should be the initially selected value when the page loads.
- **label:** An alternative way of labeling options

14.9 File Select boxes

If you want to allow a user to upload a file to your web site, you will need to use a file upload box, also known as a file select box. This is also created using the <input> element but type attribute is set to **file**.

Example: <input type="file" name="resume" accept=".pdf,.doc,.docx">

Attributes of file select box

- **type:** "file": Specifies the type of input control.
- **name:** Used to give a name to the control which is sent to the server to be recognized and get the value.
- **accept:** Specifies the types of files that the server accepts.

14.10 Buttons Control

There are various ways in HTML to create clickable buttons. You can also create a clickable button using <input> tag by setting its type attribute to **button**.

Example:

```
<input type="button" value="Click Me">
<input type="submit" value="Submit">
<input type="reset" value="Reset">
<input type="image" src="submit.png" alt="Submit">
```

Attributes

- **submit:** This creates a button that automatically submits a form.
- **reset:** This creates a button that automatically resets form controls to their initial values.
- **button:** This creates a button that is used to trigger a client-side script when the user clicks that button.
- **image:** This creates a clickable button but we can use an image as background of the button.

14.11 Hidden Form Controls

Hidden controls store data that is not visible to users but is sent to the server. This control hides inside the code and does not appear on the actual page. For example, following hidden form is being used to keep current page number. When a user will click next page then the value of hidden control will be sent to the web server and there it will decide which page will be displayed next based on the passed current page.

Example: `<input type="hidden" name="page" value="1">`

Attributes

- **type="hidden":** Specifies the type of input control.
- **name:** The name of the control.
- **value:** The value sent to the server.

14.12 Submit and Reset Button

Submit and reset buttons handle form submission and resetting.

Example:

`<input type="submit" value="Submit">`

`<input type="reset" value="Reset">`

Attributes

- **submit:** Creates a button that submits the form.
- **reset:** Creates a button that resets the form controls.

14.13 New input types in HTML 5

HTML5 introduces several new input types to improve form functionality:

- **email:** The **email** input type validates that the input is a properly formatted email address.

Example:

`<label for="email">Email:</label>`

`<input type="email" id="email" name="email" required>`

- **url:** The **url** input type validates that the input is a properly formatted URL.

Example:

`<label for="website">Website:</label>`

`<input type="url" id="website" name="website" required>`

- **number:** Allows only numeric input, with optional min, max, and step attributes.

Example:

`<label for="age">Age:</label>`

`<input type="number" id="age" name="age" min="18" max="99" step="1">`

- **tel:** For telephone number input.

Example:

`<label for="phone">Phone:</label>`

`<input type="tel" id="phone" name="phone" pattern="[0-9]{3}-[0-9]{3}-[0-9]{4}" placeholder="123-456-7890">`

- **search:** The search input type is optimized for search queries.

Example:

`<label for="search">Search:</label>`

`<input type="search" id="search" name="search">`

- **date:** The **date** input type allows users to select a date.
Example:
`<label for="birthday">Birthday:</label>`
`<input type="date" id="birthday" name="birthday">`
- **datetime-local:** The **datetime-local** input type allows users to select both date and time.
Example:
`<label for="appointment">Appointment:</label>`
`<input type="datetime-local" id="appointment" name="appointment">`
- **month:** The month input type allows users to select a month and year.
Example:
`<label for="month">Select Month:</label>`
`<input type="month" id="month" name="month">`
- **week:** The week input type allows users to select a specific week and year.
Example:
`<label for="week">Select Week:</label>`
`<input type="week" id="week" name="week">`
- **time:** The time input type allows users to select a time.
Example:
`<label for="time">Select Time:</label>`
`<input type="time" id="time" name="time">`
- **color:** The color input type allows users to select a color.
Example:
`<label for="favcolor">Favorite Color:</label>`
`<input type="color" id="favcolor" name="favcolor">`

Full Example

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>HTML5 Input Types Example</title>
</head>
<body>
  <form>
    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required><br><br>

    <label for="website">Website:</label>
    <input type="url" id="website" name="website" required><br><br>

    <label for="age">Age:</label>
    <input type="number" id="age" name="age" min="18" max="99" step="1"><br><br>

    <label for="phone">Phone:</label>
    <input type="tel" id="phone" name="phone" pattern="[0-9]{3}-[0-9]{3}-[0-9]{4}"
    placeholder="123-456-7890"><br><br>
```

```

<label for="search">Search:</label>
<input type="search" id="search" name="search"><br><br>

<label for="birthday">Birthday:</label>
<input type="date" id="birthday" name="birthday"><br><br>

<label for="appointment">Appointment:</label>
<input type="datetime-local" id="appointment" name="appointment"><br><br>

<label for="month">Select Month:</label>
<input type="month" id="month" name="month"><br><br>

<label for="week">Select Week:</label>
<input type="week" id="week" name="week"><br><br>

<label for="time">Select Time:</label>
<input type="time" id="time" name="time"><br><br>

<label for="favcolor">Favorite Color:</label>
<input type="color" id="favcolor" name="favcolor"><br><br>

<input type="submit" value="Submit">
</form>
</body>
</html>

```

14.14 Attributes for Form validation

- **required:** Specifies that an input field must be filled out before submitting the form.
- **pattern:** Specifies a regular expression that the input field's value must match.
- **min and max:** Define the min. & max. values for number, range, date, datetime-local, etc.
- **step:** Specifies the interval between legal numbers in an input field.
- **placeholder:** Provides a hint to the user of what can be entered in the field.
- **autofocus:** Automatically focuses on this element when the page loads.

Lesson 15: Multimedia in HTML 5

HTML 5 introduced several new elements and attributes to support multimedia content directly in webpages, allowing for rich and interactive experience without relying on external plugins.

15.1 Audio

The <audio> tag is used to embed audio content in a webpage. It supports various audio formats and provides built-in playback controls.

Syntax:

```

<audio controls>
  <source src="audio.mp3" type="audio/mpeg">
  <source src="audio.ogg" type="audio/ogg">
  Your browser does not support the audio element.
</audio>

```

Attributes

- **src:** Specifies the URL of the audio file.
- **controls:** Adds playback controls (play, pause, volume) to the audio player.
- **autoplay:** Automatically starts playing the audio when the page loads.
- **loop:** Repeats the audio playback.
- **muted:** Mutes the audio.
- **preload:** Indicates how the audio should be loaded (auto, metadata, or none).

15.2 Video

The <video> element is used to embed video content. It supports various video formats and provides built-in playback controls.

Syntax

```
<video controls width="600" poster="video-poster.jpg">  
  <source src="video.mp4" type="video/mp4">  
  <source src="video.webm" type="video/webm">  
  Your browser does not support the video element.  
</video>
```

Attributes

- **src:** Specifies the URL of the video file.
- **controls:** Adds playback controls (play, pause, volume) to the video player.
- **autoplay:** Automatically starts playing the video when the page loads.
- **loop:** Repeats the video playback.
- **muted:** Mutes the video.
- **poster:** Specifies an image to show while the video is loading or before it starts playing.
- **preload:** Indicates how the video should be loaded (auto, metadata, or none).

15.3 Track

The <track> element provides text tracks for <video> and <audio>, such as subtitles, captions, or descriptions.

Syntax

```
<video controls width="600" poster="video-poster.jpg">  
  <source src="video.mp4" type="video/mp4">  
  <track src="subtitles_en.vtt" kind="subtitles" srclang="en" label="English" default>  
  Your browser does not support the video element.  
</video>
```

Attributes

- **src:** Specifies the URL of the track file.
- **kind:** Defines the kind of text track (subtitles, captions, descriptions, chapters, or metadata).
- **srclang:** Specifies the language of the track.
- **label:** Specifies the title of the track.
- **default:** Indicates that this track should be used by default.

15.4 Embed

The <embed> element is used to embed external content like multimedia or interactive elements, such as Flash or PDF files. While not new to HTML5, it continues to be useful for embedding content.

Syntax

```
<embed src="movie.swf" width="400" height="300">
```

Attributes

- **src:** Specifies the URL of the embedded content.
- **width** and **height:** Specifies the dimensions of the embedded content.

15.5 Object

The <object> element is used to embed objects, such as multimedia or documents, into the webpage. It is more flexible than <embed> and can be used for various types of content.

Syntax

```
<object data="movie.mp4" type="video/mp4" width="600" height="400">  
  Your browser does not support the video element.  
</object>
```

Attributes

- **data:** Specifies the URL of the object.
- **type:** Specifies the MIME type of the object.
- **width** and **height:** Specifies the dimensions of the object.

15.6 Picture

The <picture> element is used to provide multiple sources for responsive images. It allows you to specify different images based on media conditions.

Syntax

```
<picture>  
  <source srcset="image-small.jpg" media="(max-width: 600px)">  
  <source srcset="image-large.jpg" media="(min-width: 601px)">  
    
</picture>
```

Attributes

- **srcset:** Specifies the URL of the image and optional sizes.
- **media:** Specifies media queries to determine which image to display.
- **alt:** Provides alternative text for the image.

Lesson 16: THE HTML TAGS YOU NEVER LEARN.

- **<abbr>:** Abbreviation tag it is used when we have to display the meaning of acronym. when you hover on the text then it will display the value contain in title attribute.

Example: <abbr title="Hyper Text Markup Language">HTML</abbr>

- **<code>:** The code tag is used to define a piece of computer code. This tag is typically used to display code snippets on a webpage. The text inside a <code> tag is usually displayed in a monospace (fixed-width) font.

Example:

```
<!DOCTYPE html>  
<html lang="en">  
  <head>  
    <title>Code Tag Example</title>  
    <style>  
      code {  
        font-family: "Courier New", Courier, monospace;  
        background-color: #f4f4f4;  
        padding: 2px 4px;  
        border-radius: 4px;  
      }  
    </style>  
  </head>  
  <body>  
    <h1>Example of <code>&lt;code&gt;</code> Tag</h1>  
  </body>
```

</html>

- **<kbd>:** The <kbd> tag in HTML is used to define keyboard input. This tag is used to display shortcut keys to the user what they need to type.

Example:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Kbd Tag Example</title>
    <style>
      kbd {
        font-family: "Courier New", Courier, monospace;
        background-color: #f4f4f4;
        padding: 2px 4px;
        border: 1px solid #ccc;
        border-radius: 4px;
        box-shadow: 1px 1px 1px rgba(0,0,0,0.2);
      }
    </style>
  </head>
  <body>
    <h1>Example of <kbd> &lt;kbd>></kbd> Tag</h1>
    <p>To save your work, press <kbd>Ctrl</kbd> + <kbd>S</kbd>.</p>
    <p>To open the search dialog, press <kbd>Ctrl</kbd> + <kbd>F</kbd>.</p>
  </body>
</html>
```

- **<datalist> and <option>:** The <datalist> element in HTML is used to provide a list of predefined options for an <input> element. These options are shown to the user as they type into the input field as suggestion. The <option> tag is nested inside the <datalist> tag. The **list name** in <input> tag and **id name** in <datalist> tag should be same.

Example:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>datalist</title>
  </head>
  <body>
    <label>Choose your favorite browser:</label>
    <input list="browsers">
    <datalist id="browsers">
      <option value="chrome"></option>
      <option value="opera-mini"></option>
      <option value="Firefox"></option>
      <option value="Edge"></option>
    </datalist>
  </body>
</html>
```

- **<dialog>:** The dialog tag in HTML is used to define a dialog box or a Pop-up window. This element is useful for creating modal dialogs, which can be used to display information or error without navigating away from the current page.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Dialog Example</title>
  <style>
    dialog {
      border: none;
      border-radius: 10px;
      padding: 20px;
      box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
    }
  </style>
</head>
<body>
  <button id="showDialog">Show Dialog</button>

  <dialog id="myDialog">
    <h2>Dialog Box</h2>
    <p>This is an example of a dialog box.</p>
    <button id="closeDialog">Close</button>
  </dialog>

  <script>
    const dialog = document.getElementById('myDialog');
    const showDialogButton = document.getElementById('showDialog');
    const closeDialogButton = document.getElementById('closeDialog');

    showDialogButton.addEventListener('click', () => {
      dialog.showModal(); // Opens the dialog
    });

    closeDialogButton.addEventListener('click', () => {
      dialog.close(); // Closes the dialog
    });
  </script>
</body>
</html>
```

- **<details> and <summary>:** used to create a accordion like FAQ. Used to create a disclosure widget that user can open and close. <summary> tag is nested inside <details> tag.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Nested Details Example</title>
</head>
<body>
  <h1>Example of Nested details</h1>

  <details>
```

```

<summary>More Information</summary>
<p>This is the additional information that was hidden. You can provide more details here.</p>

<details>
  <summary>Even More Information</summary>
  <p>This is even more detailed information inside a nested <code>&lt;details&gt;</code>
    element.</p>
</details>
</details>
</body>
</html>

```

- **<ruby> and <rt>** : if you want to add some text in top of the text then you can use <ruby> and <rt> tag. It is useful in educational material for teaching pronunciation of characters.

Example:

```

<body>
  <h1>Example of ruby and rt tags</h1>
  <p>Here is a word with a ruby annotation:</p>
  <ruby>Main Text
    <rt>Little Text</rt>
  <rp>little More</rp>
</ruby>
  <ruby>Main Text
    <rt>Little Text</rt>
  </ruby>
</body>

```

- **<progress>**: The <progress> tag is used to represent the completion progress of a task, such as a download, file upload, or the completion of a form. It contain two attributes **value** and **max**.
 - **value**: value indicates how much task has been completed it should be number between **0** and **max** value.
 - **Max**: It specifies the total value representing the completion of the task. Default value is 1.

Example:

```

<body>
  <h1>Example of the <code>&lt;progress&gt;</code> Tag</h1>
  <label for="file">Downloading file:</label>
  <progress id="file" value="32" max="100">32%</progress>
</body>

```

Write the following CSS code for the above progress tag.

```

<style>
  progress {
    width: 100%;
    height: 30px;
    appearance: none;
  }

  progress::-webkit-progress-bar {
    background-color: #f3f3f3;
    border-radius: 5px;
  }

```

```

progress::-webkit-progress-value {
  background-color: #4caf50;
  border-radius: 5px;
}

progress::-moz-progress-bar {
  background-color: #4caf50;
  border-radius: 5px;
}
</style>

```

Animated progress tag example:

```

<body>
  <h1>Dynamic <code>&lt;progress&gt;</code> Example</h1>

  <label for="progress">Progress:</label>
  <progress id="progress" value="0" max="100"></progress>
  <button onclick="startProgress()">Start Progress</button>

  <script>
    function startProgress() {
      let progress = document.getElementById('progress');
      let value = 0;

      let interval = setInterval(() => {
        value += 1;
        progress.value = value;

        if (value >= 100) {
          clearInterval(interval);
        }
      }, 100);
    }
  </script>
</body>

```

- **<meter>**: meter tag is also same as <progress> tag. It can be used to display CPU usages, Memory, network usages. Also, to show the battery level in web application, strength of password , blood sugar level, other medical measurements., etc.

Attributes of meter tag are:

- **value**: Required. Specifies the current value of the measurement.
- **min**: Optional. Specifies the minimum value of the measurement. The default value is 0.
- **max**: Optional. Specifies the maximum value of the measurement. The default value is 100.
- **low**: Optional. Specifies the lower bound of the "low" range.
- **high**: Optional. Specifies the upper bound of the "high" range.
- **optimum**: Optional. Specifies the optimum value or range.

Example:

```

<!DOCTYPE html>
<html lang="en">
<head>
  <title>Styled Meter Example</title>
</head>

```

```

<body>
  <h1>Styled <code>&lt;meter&gt;</code> Example</h1>
  <label for="temperature">Temperature Level:</label>
  <meter id="temperature" value="65" min="0" max="100" low="30" high="70"
optimum="50">65%</meter>
</body>
</html>

```

Add the following CSS code into the above HTML code

```

<style>
  meter {
    width: 100%;
    height: 20px;
  }

  meter::-webkit-meter-bar {
    background: #eee;
    border-radius: 5px;
  }

  meter::-webkit-meter-optimum-value {
    background: #4caf50;
    border-radius: 5px;
  }

  meter::-webkit-meter-suboptimum-value {
    background: #ffeb3b;
    border-radius: 5px;
  }

  meter::-webkit-meter-even-less-good-value {
    background: #f44336;
    border-radius: 5px;
  }
</style>

```

- **<fieldset>** and **<legend>**: The fieldset and legend tags in HTML are used together to group related elements within a form, providing a visual and structural organization.
 - **<fieldset>**: It creates a box around the grouped elements.
 - **<legend>**: it is used to define a caption or title for the <fieldset>
- <legend>** tag is nested inside the **<fieldset>** tag.

Example:

```

<!DOCTYPE html>
<html lang="en">
<head>
  <title>Styled Fieldset and Legend Example</title>
  <style>
    fieldset {
      border: 2px solid #4CAF50;
      border-radius: 5px;
      padding: 20px;
      margin-bottom: 20px;
      box-shadow: 2px 2px 12px #aaa;
    }
  </style>

```

```

legend {
    background-color: #4CAF50;
    color: white;
    padding: 5px 10px;
    border-radius: 5px;
    box-shadow: 2px 2px 8px #888;
}

label {
    display: block;
    margin-top: 10px;
}

input[type="text"],
input[type="email"],
input[type="password"] {
    width: 100%;
    padding: 8px;
    margin-top: 5px;
    border: 1px solid #ccc;
    border-radius: 4px;
}

input[type="submit"] {
    background-color: #4CAF50;
    color: white;
    padding: 10px 20px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
}

input[type="submit"]:hover {
    background-color: #45a049;
}
</style>
</head>
<body>
<form action="/submit" method="post">
    <fieldset>
        <legend>Personal Information</legend>
        <label for="fname">First Name:</label>
        <input type="text" id="fname" name="fname">
        <label for="lname">Last Name:</label>
        <input type="text" id="lname" name="lname">
        <label for="email">Email:</label>
        <input type="email" id="email" name="email">
    </fieldset>

    <fieldset>
        <legend>Account Details</legend>
        <label for="username">Username:</label>

```

```

        <input type="text" id="username" name="username">
        <label for="password">Password:</label>
        <input type="password" id="password" name="password">
        <label for="confirm-password">Confirm Password:</label>
        <input type="password" id="confirm-password" name="confirm-password">
    </fieldset>
    <input type="submit" value="Submit">
</form>
</body>
</html>

```

- **<datalist>:** The datalist tag is used to provide a set of predefined options to users for an <input> element. It enhances user experience by offering suggestions based on the user's input.

Syntax

```

<input list="exampleList" id="inputField" name="inputField">
<datalist id="exampleList">
    <option value="Option 1">
    <option value="Option 2">
    <option value="Option 3">
</datalist>

```

Example:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Datalist Example</title>
    <style>
        input {
            width: 100%;
            padding: 8px;
            margin-top: 5px;
            border: 1px solid #ccc;
            border-radius: 4px;
        }
    </style>
</head>
<body>
    <h1>Datalist Example</h1>
    <label for="fruit">Choose a fruit:</label>
    <input list="fruits" id="fruit" name="fruit">

    <datalist id="fruits">
        <option value="Apple"></option>
        <option value="Banana"></option>
        <option value="Cherry"></option>
        <option value="Date"></option>
        <option value="Grape"></option>
        <option value="Kiwi"></option>
        <option value="Mango"></option>
    </datalist>

</body>
</html>

```

Lesson 17: Code Like Pro developer

Now in this topic we'll learn how to increase our code productivity 10 times faster than beginner's.

17.1 Nesting

To generate the code like below just type: **div>ul>li**

```
<div>
  <ul>
    <li> </li>
  </ul>
</div>
```

17.2 Siblings

To generate the code like below just type: **div+span**

```
<div> </div>
<span> </span>
```

17.3 Multiple Elements

To generate the code like below just type: **ul>li*4**

```
<ul>
  <li> </li>
  <li> </li>
  <li> </li>
  <li> </li>
</ul>
```

17.4 Adding Classes and IDs

To generate the code like below just type: **div.class , h1#id**

```
<div class="class"> </div>
<h1 id="id"> </h1>
```

To generate the code like below just type: **div.class1.class2**

```
<div class="class1 class2"></div>
```

17.5 Custom Attributes

To add custom attributes to the tag just type: **input[type="text"]**

```
<input type="text">
```

17.6 Grouping

To generate the code like below just type: **(ul>li)*3**

```
<ul>
  <li></li>
</ul>

<ul>
  <li></li>
</ul>

<ul>
  <li></li>
</ul>
```

17.7 Text Content Hyperlink

To generate the code like below just type: **a{click me}**

```
<a href="">click me</a>
```

17.8 Complex Structure

To generate the code like below just type: **div.container>div.row>div.col*3**

```
<div class="container">
  <div class="row">
    <div class="col"></div>
    <div class="col"></div>
    <div class="col"></div>
  </div>
</div>
```

17.9 Never do these Mistakes in HTML

Avoid these common HTML mistakes as a developer to improve your coding skills to the next level.

- i. Don't place **block** elements within **inline** elements

Example:

```
<h2><a href="#">Inline block element</a></h2> [wrong]
```

```
<a href="#">
  <h2>This is main heading</h2>
</a>
```

- ii. Always use **alt** and **loading=lazy** attributes for image tag.

Example:

```


```

- iii. Don't use **** and **<i>** for bold and italic

Example:

```
<b>This is bold text but wrong format</b>
<strong>This is bold and right format</strong>
```

- iv. Avoid Style in inline tag

Example:

```
<h2 style="color: red;">Wrong</h2>
<h2 class="red">Right</h2>
```

Use CSS for h2

```
.red {
  Color: red
}
```

- v. Use the figure element with image.

Example:

```

<figure>
  
  <figcaption>Programmer holding laptop</figcaption>
```

</figure>

- vi. Use **semantic** elements instead of **non-semantic** elements.

Example:

```
<div id="header"></div>
```

```
<div id="footer"></div>
```

```
<header>
```

```
<!-- place your header code here -->
```

```
</header>
```

```
<footer>
```

```
<!-- place your footer code here -->
```

```
</footer>
```

- vii. Always use **Favicon caching busting**

To refresh your website favicon, you can force browser to download a new version of favicon just by adding the code **?v=2** to the filename. This is especially helpful in production to make sure the user get new version.

```
<head>
```

```
<title>Document</title>
```

```
<link rel="shortcut icon" href="/favicon.ico?v=2" type="image/x-icon">
```

```
</head>
```