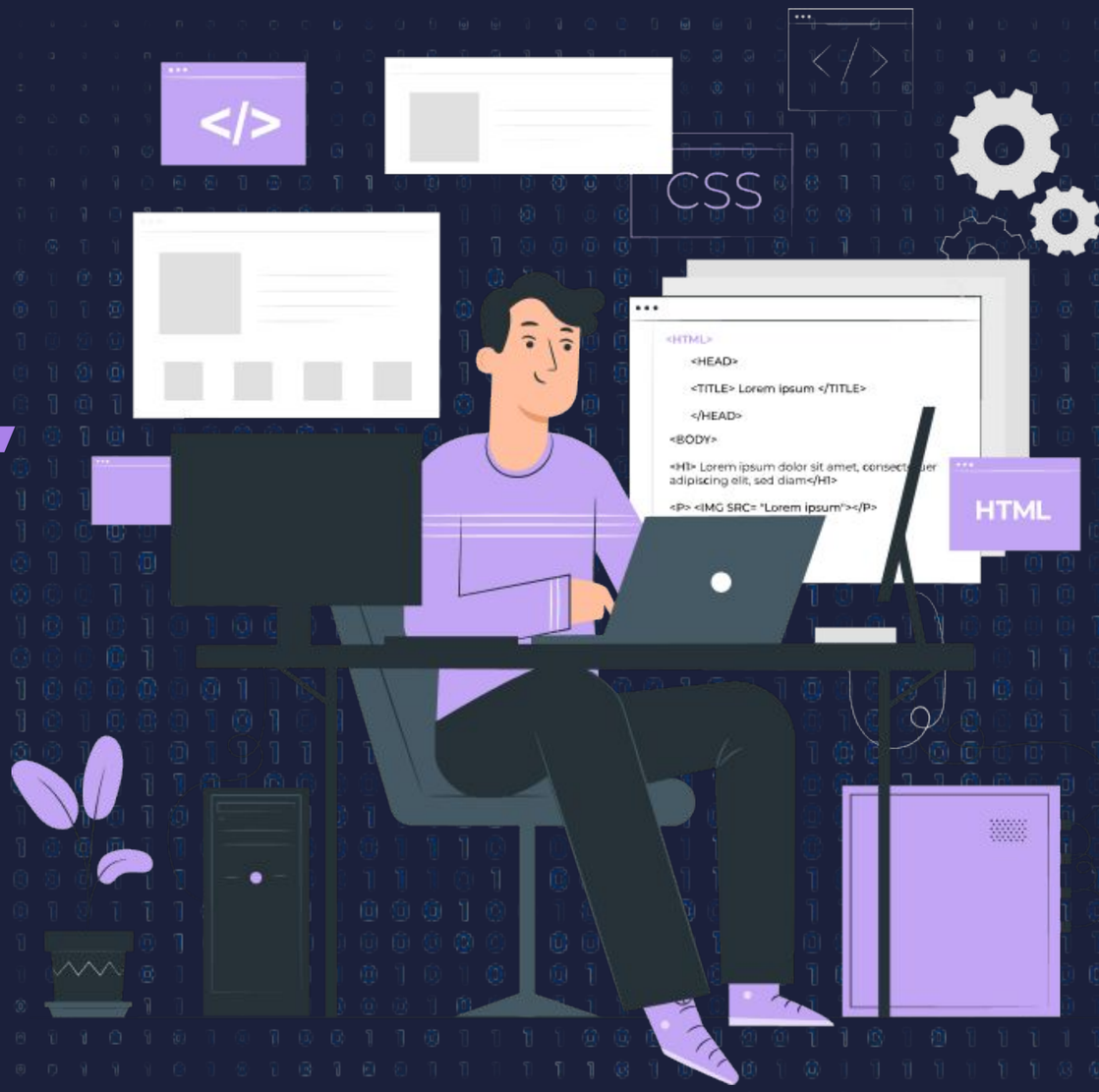
Writing a “Hello World” program in HTML and HTML Comments



Topics Covered

- Introduction to HTML
- Basic Structure of HTML Doc
- Comments in HTML

Introduction to HTML

- Hyper Text **Markup Language**
    1. **Hypertext -** Word Hypertext comes from hyperlinks, which means to connect web pages via links.
    2. **Markup -** To define and present text content.
    3. **Language -** Language gives rules and syntax to follow while writing any code.
- Used to Develop **Web Page Structure**

Basic Structure of HTML Doc

Let's create our First HTML Doc - **“Hello World”**

Step 1 : Create a new file **index.html**

Step 2 : Write following content in **index.html**.

```
<!DOCTYPE html>
<html lang="en-US">
  <head>
    <meta charset="utf-8" />
    <title> Hello, World!</title>
  </head>
  <body>
    <h1>Hello, World!</h1>
    <p>This is my first HTML page.</p>
  </body>
</html>
```

Step 3 : Open **index.html** in any Browser

Doctype Declaration

```
<!DOCTYPE html>
```

Tells that document is written in **HTML5**

<html>

```
<html lang="en-US">
</html>
```

Root of the HTML Document

<head>

```
<head>
   <!-- meta information about page -->
</head>
```

It contains **Meta Information** of HTML Document.

<meta>

```
<meta charset="utf-8" />
```

Specifies the **character encoding** for a webpage.

Character encoding refers to the process of representing characters in such a way that computers can understand.

Similarly, we have other meta tags that we will cover in later lessons.

<title>

```
<title> Hello, World!</title>
```

Sets the **Title of the web page**. In this example, it's set to "Hello, World!".

<body>

```
<body>
    <h1>Hello, World!</h1>
    <p>This is my first HTML page.</p>
</body>
```

It contains **Visible Content** of the Web Page.

Comments in HTML

Comments **Explains your Code Better**

<!-- Here you can your write comments. -->

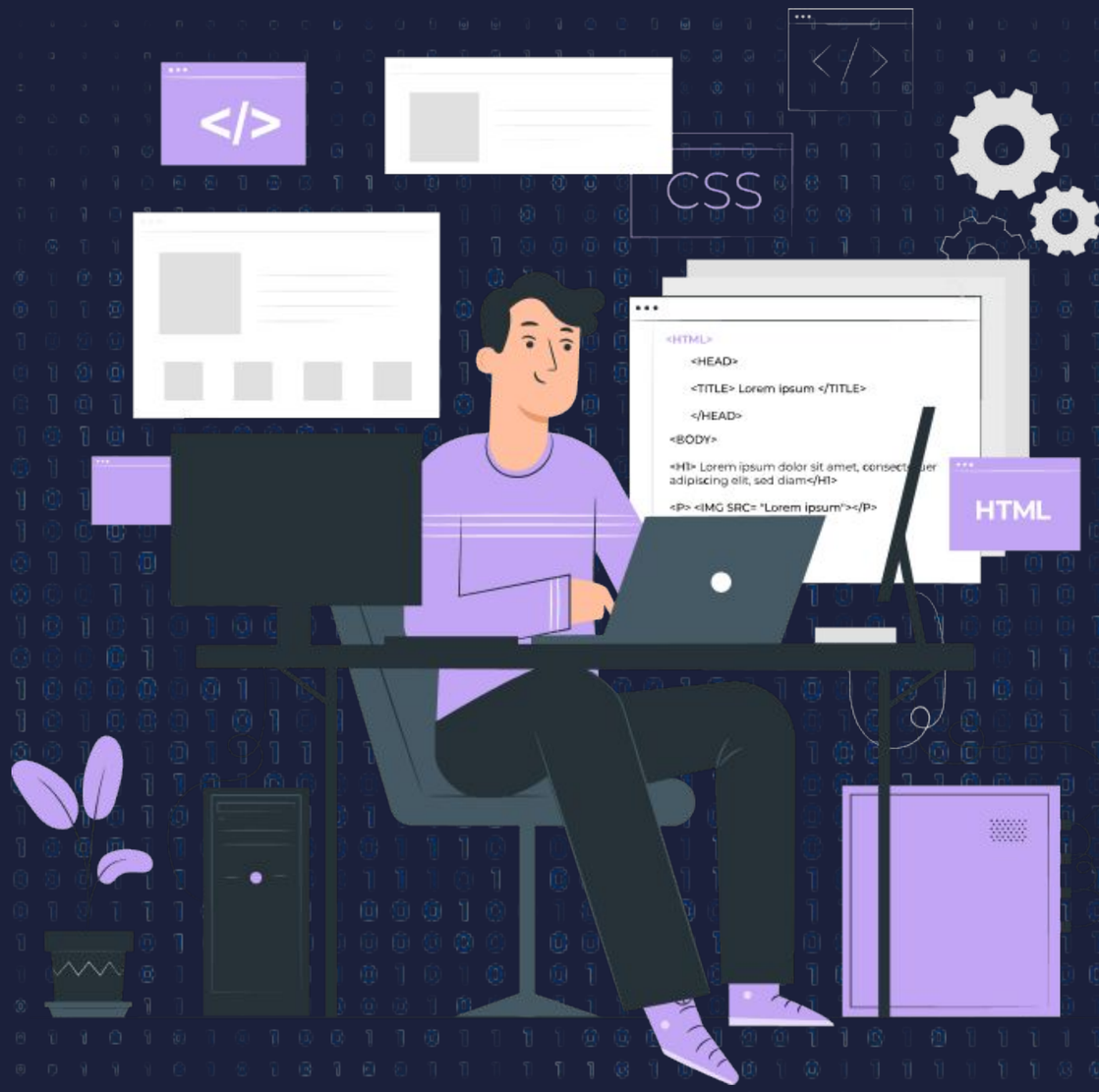
Comments Example

```
<!-- Hello Word Page -->

<!DOCTYPE html>
<html lang="en-US">

<!-- Head contain meta-information about the document -->
 <head>
  <meta charset="utf-8" />
  <title> Hello, World!</title>
 </head>

<!-- Body contains visible content of the document -->
 <body>
  <!-- Page Heading -->
  <h1>Hello, World!</h1>

   <!-- Simple Paragraph -->
  <p>This is my first HTML page.</p>
 </body>
</html>
```


SKILLS
THANK YOU

PW SKILLS

HTML Tag, Elements and Attribute



Topics Covered

- Tag and Element
- Common Tags
- Introduction to Attributes
- Common Tag Attributes

Element and Tag

Element represents most basic component of an HTML document, which can represent

- Text
- Paragraph
- List
- or any other content in a web page.

Tags are used to define elements in an HTML document.

Defining Element

Syntax For Defining HTML Element

Self Closing Tag

‹br/›

‹img/›

‹input/›

‹hr/›

No Need of Closing Tag

Common Tags - Heading

Normal Text

<h1>The Heading<h1> → **The Heading**

Levels of Heading

<h1>Heading Level 1<h1>

<h2>Heading Level 2<h2>

<h3>Heading Level 3<h3>

<h4>Heading Level 4<h4>

<h5>Heading Level 5<h5>

<h6>Heading Level 6<h6>

Heading Level 1

Heading Level 2

Heading Level 3

Heading Level 4

Heading Level 5

Heading Level 6

Wiki Page Example

PW SKILLS

HTML element

Main Heading

29 languages

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

This article is about the HTML elements in general. For information on how to format Wikipedia entries, see Help:Wiki markup and Help:HTML in wikitext.
"nobr" redirects here. For the chemical compound NOBr, see Nitrosyl bromide.
"Font color" redirects here. For OpenType fonts featuring multicolored glyphs, see OpenType § Color fonts.

An **HTML element** is a type of HTML (HyperText Markup Language) document component, one of several types of HTML nodes (there are also text nodes, comment nodes and others).[*vague*] The first used version of HTML was written by Tim Berners-Lee in 1993 and there have since been many versions of HTML. The most commonly used version is HTML 4.01, which became official standard in December 1999.[1] An HTML document is composed of a tree of simple HTML nodes, such as text nodes, and HTML elements, which add semantics and formatting to parts of document (e.g., make text bold, organize it into paragraphs, lists and tables, or embed hyperlinks and images). Each element can have HTML attributes specified. Elements can also have content, including other elements and text.

HTML

Dynamic HTML · HTML5 (article · audio · canvas · video) · XHTML (Basic · Mobile Profile) · **HTML element** (meta · div and span · blink · marquee) · HTML attribute (alt attribute) · HTML frame · HTML editor · Character encodings (named characters · Unicode) · Language code · Document Object Model · Browser Object Model · Style sheets (CSS) · Font family · Web colors · JavaScript (WebGL · WebCL) · W3C (Validator) · WHATWG · Quirks mode · Web storage · Rendering engine

Comparisons

Document markup languages · Comparison of browser engines

V · T · E

Concepts [edit]

Sub Heading

Elements vs. tags [edit]

Sub Sub Heading

As is generally understood, the position of an element is indicated as spanning from a start tag and is terminated by an end tag.[2] This is the case for many, but not all, elements within an

Common Tags - Paragraph

```
<p>
PW Skills is the result of a continual effort to exponentially increase the
employability of every Indian, irrespective of their socioeconomic status.
</p>
```

PW Skills is the result of a continual effort to exponentially increase the employability of every Indian, irrespective of their socioeconomic status.

Common Tags - Div

Creates a **Container**

Group Other **Elements**

```
<div>
   <h1>Header</h1>
   <p>This is a paragraph of text.</p>
</div>
```

Tags to Study

Tag Class	Tags	Description
Formatting Text	<b>, <s>, <i>, <u> etc.	For text formatting, **Bold,** ~~Strikethrough,~~ *Italic,* Underline
Hyperlinks	<a>	Insert hyperlinks to interlink pages.
Table	<table>, <tr>, <td>, <th> etc.	To arrange data in rows and columns
Lists	,	To arrange data in the form of a list.
Forms	<form>, <input> etc.	To take input from the user and submit it to the server.
Embed Media	<img>, <audio>, <video> etc.	To include media in HTML doc.
Structuring Tags	, <nav>, <header>, <main>, <section> etc.	To give the web page a structure.

Introduction to Attribute

```
<p id="main"></p>
```

Extending **HTML** Element Behaviour

Common Tag Attributes

Id Attribute

```
<div id="price-1">Rs 399</div>
```

Identifies Element Uniquely

Common Tag Attributes

class attribute

```
<div class="content popup"></div>
```

Allows to assign multiple CSS classes

Common Tag Attributes

style Attribute

```
<div style="color:red"></div>
```

Apply Inline CSS styles

Common Tag Attributes

Multiple HTML Attribute

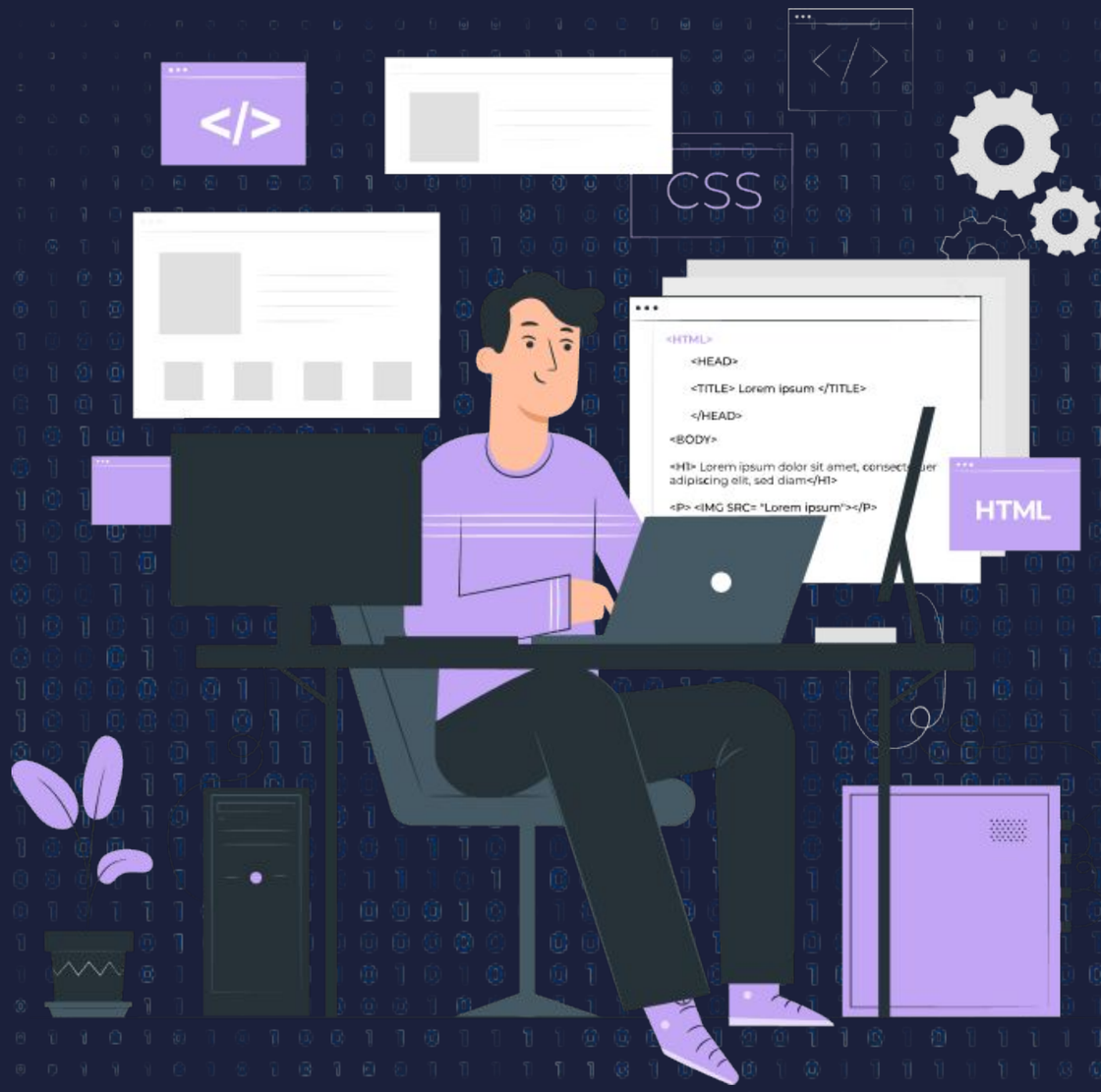
```
<p id="para" class="main"></p>
```

Always Put **Space** between two attributes


SKILLS
THANK YOU

PW SKILLS

Introduction to Emmet and Speedy HTML



Topics Covered

- What is Emmet?
- Basic Syntax of Emmet
    - Emmet Abbreviations
- Advantages of Emmet

What is Emmet?

Emmet is a **Shorthand Abbreviation Syntax**

Write Code HTML Syntax **Faster**

Emmet Abbreviations - Element

Syntax: element_name + Tab Key

div	➡	
span	➡	

Emmet Abbreviations - Text {}

Syntax: element_name{your text} + Tab Key

div{PW}	➡	PW
span{Skill}	➡	Skill

Emmet Abbreviations - id

Syntax: element_name#idName + Tab Key

div#main ➡ ** **

Emmet Abbreviations - class

Syntax: element_name.className + Tab Key

span.price ➡

Emmet Abbreviations - General Attribute []

Syntax: ementType[attributeName=attributeValue]

input[name="email"] ➡ <input name="email"'> </input>

img[alt="alt text"] ➡ <img alt="alt text"> </img>

Advantages of Emmet

Abbreviations and Shortcuts

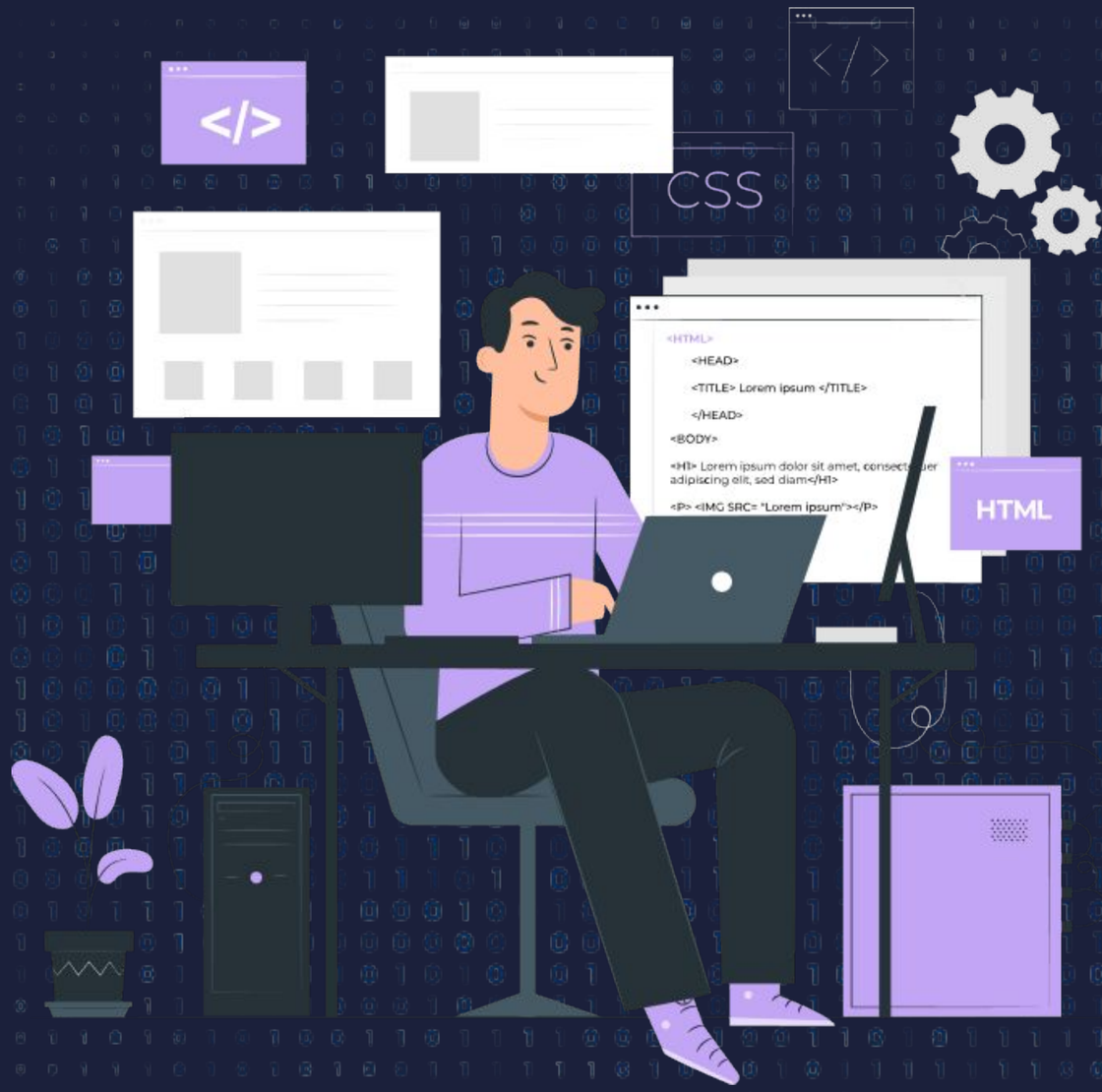
Increased productivity

Write complex HTML code with just a

Few Keystrokes.


SKILLS
THANK YOU

Grouping of HTML elements



Topics Covered

- What is Grouping of HTML elements?
- Why Grouping?
- Grouping Tags in HTML
- Emmet Syntax for Grouping
  - Child >
  - Sibling +
  - Climb Up ^
  - Multiplication *
  - Grouping ()

What is Grouping of HTML elements?

Combining Two or more Elements in Emmet Syntax

- Adding two elements side by side
- Adding child element to parent

We have **Grouping Tags** to group elements in HTML

Why Grouping?

To **apply** certain **styles** or properties **to multiple elements at once**, rather than individually applying them to each element.

Grouping also helps in **providing a logical structure** to the content.

Grouping Tags Examples

The tags used to group multiple other elements inside it are called grouping tags.

Some common grouping tags are, **div, section, article, main, header, footer, ul, ol, table** etc.

Examples of Grouping

1. Using the ** ** element to group elements together:

```
<div>
  <h1>Heading 1</h1>
  <p>Paragraph 1</p>
  <p>Paragraph 2</p>
</div>
```

Examples of Grouping

2. Using the **<section> and <article>** element to group related content together

```
<section>
  <article>
     <h2>Article Heading</h2>
     <p>Paragraph 1</p>
     <p>Paragraph 2</p>
  </article>
</section>
```

Examples of Grouping

3. Using the <ul> and <li> elements to create a list

```
<ul>
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ul>
```

Emmet Syntax for Grouping

We can combine two or more elements in emmet syntax using **Nesting Operators** in Emmet.

Grouping in Emmet is useful,

- When you want to write complex html structures faster.
- It helps in combining different syntax in a single line and helps developers to code faster.

Nesting Operators Child >

Syntax: parentElement > childElement + Tab Key

div>a>h1{Link} ➡

```
<div>
    <a>
        <h1>Link</h1>
    </a>
</div>
```

Nesting Operators - Sibling +

Syntax: element1 + element2 + Tab Key

header+main+footer ➡

```
<header></header>
<main></main>
<footer></footer>
```

Nesting Operators - Climb Up ^

Syntax: element1 ^ element2 + Tab Key

div>p>span^a

```
<div>
  <p>
    <span></span>
  </p>
  <a></a>
</div>
```

Nesting Operators - Multiplication *

Syntax: element1 * multiplier + Tab Key

ul>li*3 ➞

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

Nesting Operators Grouping ()

Syntax: (Valid Emmet Syntax) + Tab Key

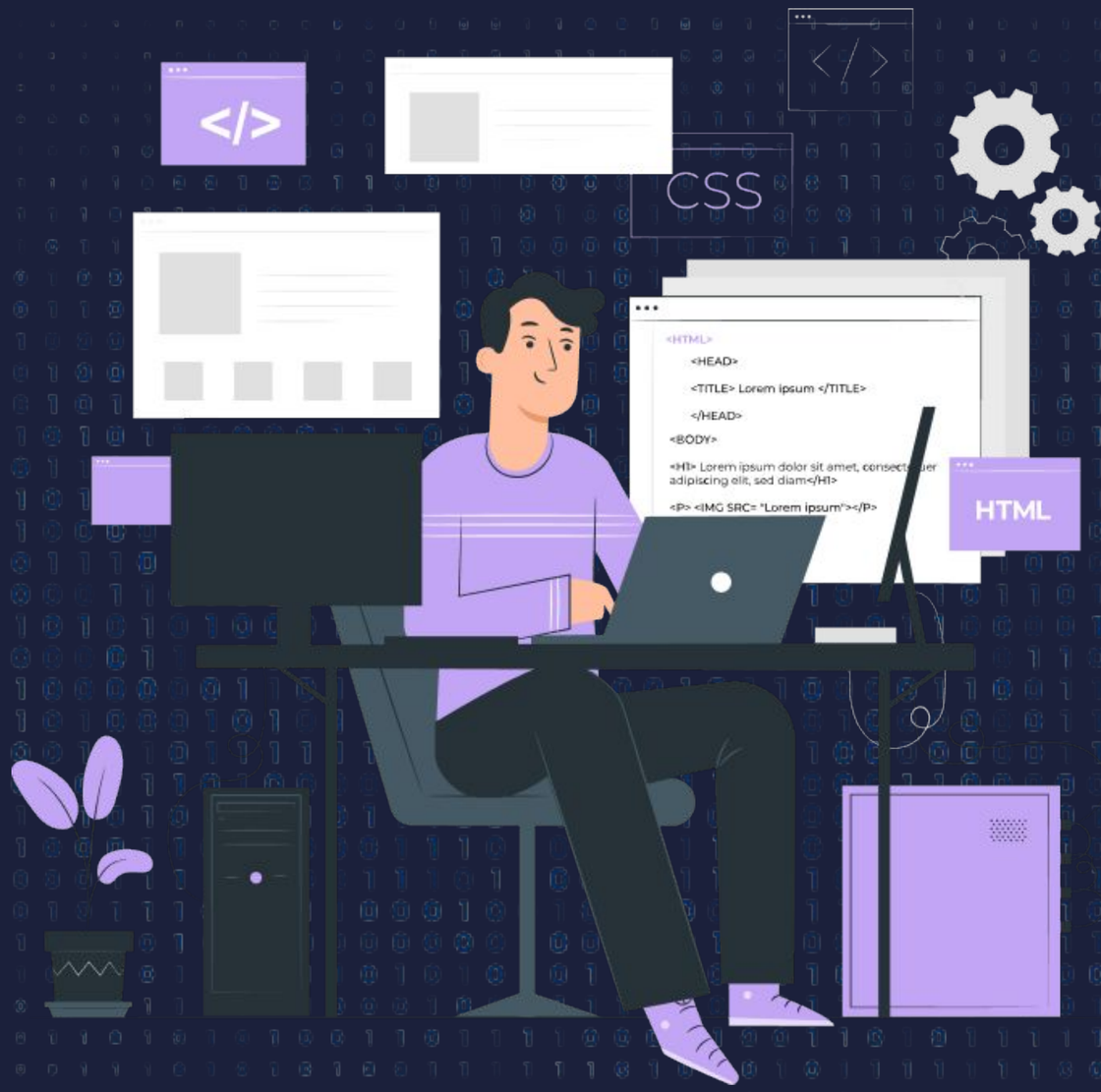
```
div>(ul>li>a)*3
```

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


SKILLS
THANK YOU

PW SKILLS

Understanding Formatting Tag



Topics Covered

- Formatting Tag and Need of It
- Commonly used Formatting Tags

Formatting tags

Formatting Tags used to Format Text.

eg. bold, italic, underline, and much more.

Why? It controls the appearance of text content, making it easier to read and understand.

Commonly used Formatting Tags

1. Bold ** Tag**

The Bold Text ➡ **The Bold Text**

Alternatively, we can use **** tag, only difference between **** and **
** is that **** has semantics.

Commonly used Formatting Tags

2. Italics tag

The Italics Text ➡ The *Italics* Text

Commonly used Formatting Tags

3. Underline **<u> tag**

<u>The Underline Text</u> ➡ <u>The Underline Text</u>

Commonly used Formatting Tags

4. Strikethrough **<s> tag**

<s>Strikethrough Text</s> ➡ ~~Strikethrough Text~~

Commonly used Formatting Tags

5. Emphasis ** tag**

Emphasis Text ➡ ***Emphasized Text***

Increases importance of text as **compared to surrounding text.**

Commonly used Formatting Tags

6. Subscript formatting to text **<sub/> Tag**

The chemical formula of water is H₂0

➡

The chemical formula of water is $H_2$0

Commonly used Formatting Tags

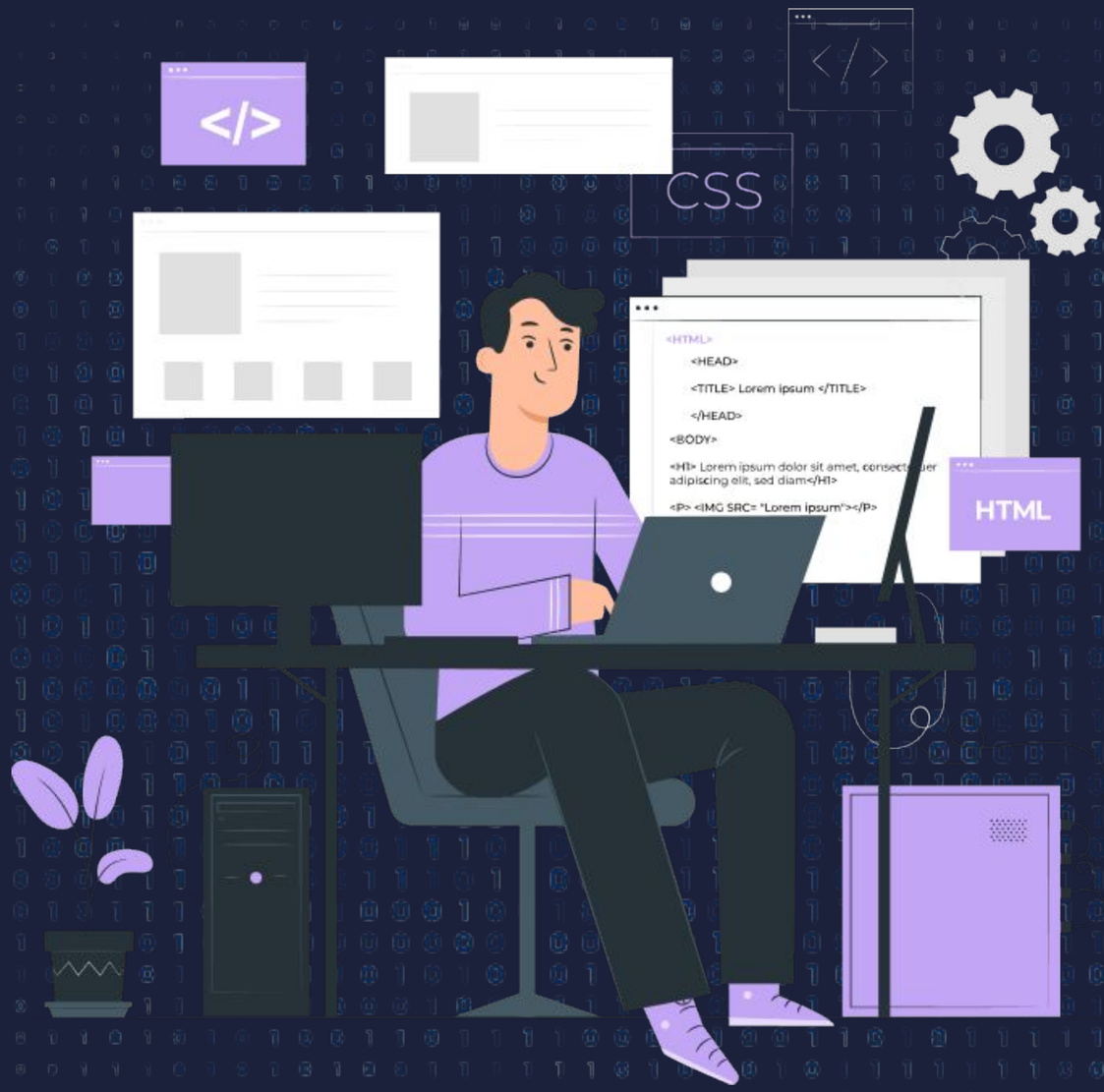
7. Superscript formatting to text **<sup/>Tag**

2² + 2² = 8 ⟶ $2^2 + 2^2 = 8$


SKILLS
THANK YOU

PW SKILLS

List and Anchor Tag



Topics Covered

- Introduction of List tag
- Types of Listing Tags (ul, ol, dl)
- Type Attribute of the list tags
- Introduction of Anchor tag
- Different attributes of anchor tag

Introduction of List Tag

Collection of Items in organised manner

List Types

- UnOrdered
- Ordered
- DataList (description list)

UnOrdered List

Items **Order Does Not Matter**

```
<ul>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
</ul>
```

- Item 1
- Item 2
- Item 3

Ordered List

Items **Order Matter**

```
<ol>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
</ol>
```

1. Item 1
2. Item 2
3. Item 3

Data List (description list)

Items **Order Matter**

```
<dl>
  <dt>Front-end</dt>
  <dd>– HTML, CSS, and JavaScript</dd>
   <dt>Back-end</dt>
  <dd>– Nodejs, MongoDB, and Express</dd>
</dl>
```

A Description List Demo

Front-end
- HTML, CSS, and JavaScript

Back-end
- Nodejs, MongoDB, and Express

Type Attribute of the list tags

Four makers for Unordered list -

- disc
- circle
- square
- none

disc marker example -

```
<body>
   <p>disc marker</p>
   <ul type="disc">
     <li>Item one</li>
     <li>Item two</li>
     <li>Item three</li>
   </ul>
 </body>
```

disc marker

- Item one
- Item two
- Item three

circle marker example -

```
<body>
<p>circle marker</p>
    <ul type="circle">
      <li>Item one</li>
      <li>Item two</li>
      <li>Item three</li>
    </ul>
  </body>
```

circle marker

- Item one
- Item two
- Item three

Type Attribute of the list tags

Example of square marker -

```
<body>
    <p>square marker</p>
    <ul type="square">
      <li>Item one</li>
      <li>Item two</li>
      <li>Item three</li>
    </ul>
</body>
```

square marker

- Item one
- Item two
- Item three

Example of none marker -

```
<body>
    <p>none marker</p>
    <ul type="none">
      <li>Item one</li>
      <li>Item two</li>
      <li>Item three</li>
    </ul>
 </body>
```

none marker

Item one
Item two
Item three

Five makers for ordered list -

- number [1]
- lowercase alphabets [a]
- uppercase alphabets [A]
- lowercase roman numbers[i]
- uppercase roman numbers[I]

Example - capital alphabets

```
<body>
    <p>alphabet marker</p>
    <ol type="A">
      <li>Item one</li>
      <li>Item two</li>
      <li>Item three</li>
    </ol>
  </body>
```

alphabet marker

A. Item one
B. Item two
C. Item three

Example - number

```
<body>
    <p>number marker</p>
    <ol type="1">
      <li>Item one</li>
      <li>Item two</li>
      <li>Item three</li>
    </ol>
  </body>
```

number marker

1. Item one
2. Item two
3. Item three

Example - lowercase alphabets

```
<body>
    <p>alphabet marker</p>
    <ol type="a">
      <li>Item one</li>
      <li>Item two</li>
      <li>Item three</li>
    </ol>
  </body>
```

alphabet marker

a. Item one
b. Item two
c. Item three

Five makers for ordered list -

Example - number

```
<body>
    <p>number marker</p>
    <ol type="i">
      <li>Item one</li>
      <li>Item two</li>
      <li>Item three</li>
    </ol>
  </body>
```

lowercase roman numbers

i. Item one
ii. Item two
iii. Item three

Example - number

```
<body>
    <p>number marker</p>
    <ol type="I">
      <li>Item one</li>
      <li>Item two</li>
      <li>Item three</li>
    </ol>
  </body>
```

uppercase roman numbers

I. Item one
II. Item two
III. Item three

Introduction of Anchor <a> Tag

Link Pages using HyperLinks

```
<a
href="https://www.google.com/">
Search on Google
</a>
```

On Clicked

Block Level Link

Wrap Link around a **Block** Element

```
<a
href="https://pwskills.com/">
  <h2>Home</h2>
</a>
```

Image Link

Wrap Link around a Image Element

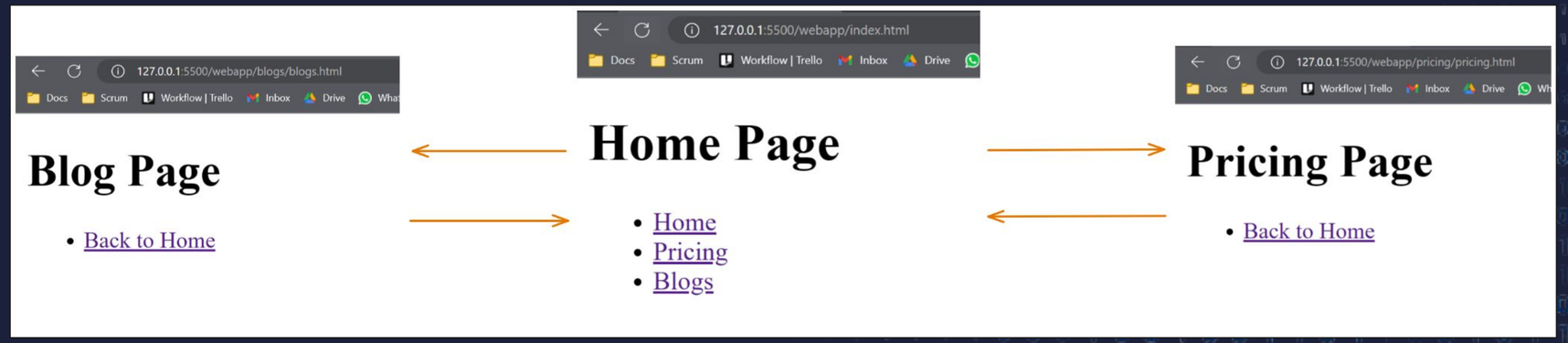
```
<a
href="https://pwskills.com">
  <img
  src="PWSkills-main.png"
  alt="logo"/>
</a>
```

Inline link

On click of any of the link i.e HTML and CSS it will direct to the particular content in the same page.

```
<body>
      <ul>
        <li><a
href="#pt-one">HTML </a></li>
        <li><a
href="#pt-two">CSS </a></li>
      </ul>
      <h1 id="pt-one"> HTML
Content </h1>
      <p>Lorem...</p>
      <h1 id="pt-two"> CSS
Content</h1>
      <p> lorem....</p>
    </ol>
  </body>
```

Navigating Between Pages




```
<a
href="../index.html">
Back to Home</a>
```

```
<a href="#">Home</a>
<a href="pricing/pricing.html">Pricing</a>
<a href="blogs/blogs.html">Blogs</a>
```

```
<a
href="../index.html">
Back to Home</a>
```

Different Attributes of Anchor Tag

href : URL of the page

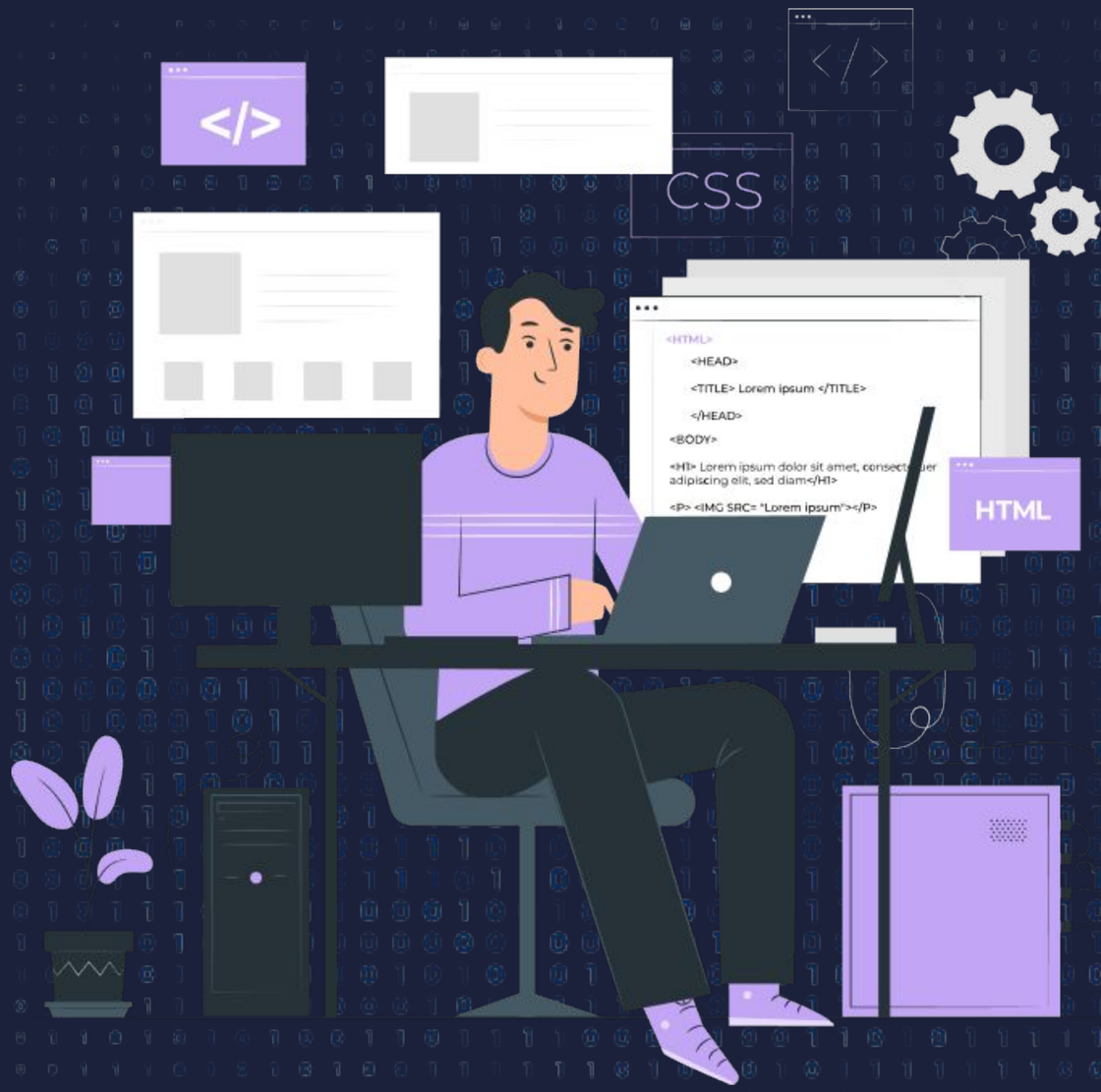
target : Specify where to open URL

target="_blank"	→	Open the URL in a new Tab
target="_self"	→	Opens the URL in the same Tab
target="_parent"	→	Opens the URL in the parent frame
target="_top"	→	top-level browsing context, usually window


SKILLS
THANK YOU

PW SKILLS

Inline Element vs Block Level Element



Topics

- Introduction to Inline elements
- Introduction to Block elements
- Inline vs Block level elements

Introduction to Inline Elements with Example

Starts on **Same Line** if space available.

Occupies **Minimal Space** necessary for content.

```
<span style="background-color: yellow;">Inline Element 1</span>
<span style="background-color: red;">Inline Element 2</span>
<span style="background-color: orange;">Inline Element 3</span>
```

Inline Element 1 Inline Element 2 Inline Element 3

Introduction to Block elements with example

Starts on a **New Line**.

Takes **Full Width** available.

```
<div style="background-color: yellow;">Block Element 1</div>
<div style="background-color: red;">   Block Element 2</div>
<div style="background-color: orange;">Block Element 3</div>
```

Block Element 1
Block Element 2
Block Element 3

Examples

Inline	Block
<a>, <b>, , <i>, <sub>, <sup>, <button>, <img>, <input> <label>, ,	<article>, <aside>, , <header>, <main>, <footer>, <section>, <table>, <tfoot>, <form>, <h1> . . . <h6>, <p>, <nav>, ,

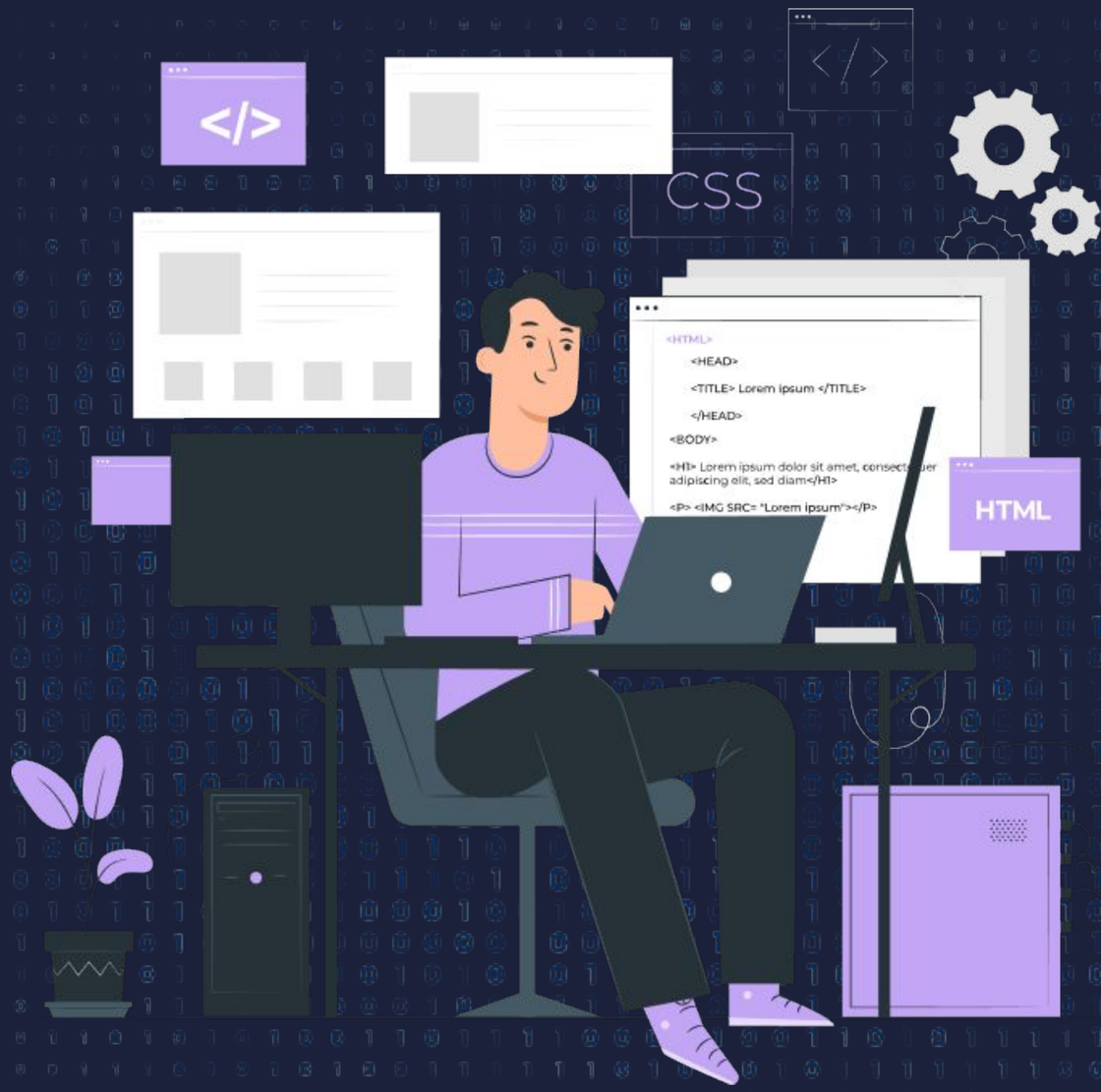
Inline vs Block Level Elements

	Inline level element	Block level Element
Display	Inline elements flow within the text of a web page.	Block-level elements create a new line and take up the full width of their parent container.
Content	Inline elements are typically used for small chunks of content such as hyperlinks.	Block-element are used for larger sections of content such as paragraph or heading
Nesting	Inline element can be nested within other inline elements or block elements	Block level element cannot be nested within inline element
Styling	Inline elements typically affect only the content within the element itself.	Block level elements affect the layout of the surrounding content.


SKILLS
THANK YOU

PW SKILLS

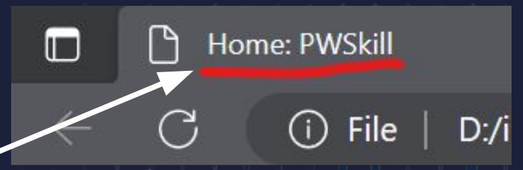
What is SEO, understanding the meta tags and Favicon



Topics Covered

- Document Title
- Meta tags
- Meta Tag - Description
- Meta Tag - Author
- What is SEO? Need of it
- What is a favicon and how to use it ?

Adding Document Title




```
<head>
  <title>Home : PWSkill </title>
</head>
```

It Appears as **Page Title** on Browser Tab

Adding Meta Data - Description

```
<head>
  <meta
    name="description"
   content="This is a sample description.">
</head>
```

It provides **Description** of Web Page.

Adding Meta Data - Author

```
<head>
  <meta name="author" content="Anurag Tiwari">
</head>
```

Tells, **Who Developed** this web page?

What is SEO?

SEO stands for **Search Engine Optimization**.

Making **Website Rank Higher** in Search Engine Results pages.

Why SEO? Increase website **Visibility, User Traffic**.

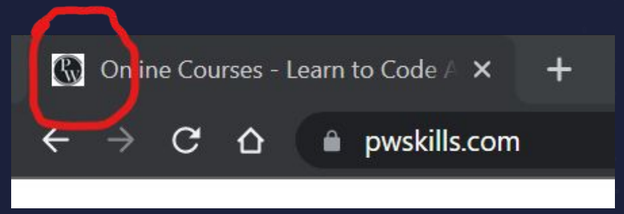
In HTML, some common ways to help search engines is to,

- Add Document Titles
- Add meta tags.

Favicon

A favicon (favourite icon) is a tiny icon displayed in places like the browser's address bar, page tabs and bookmarks menu.

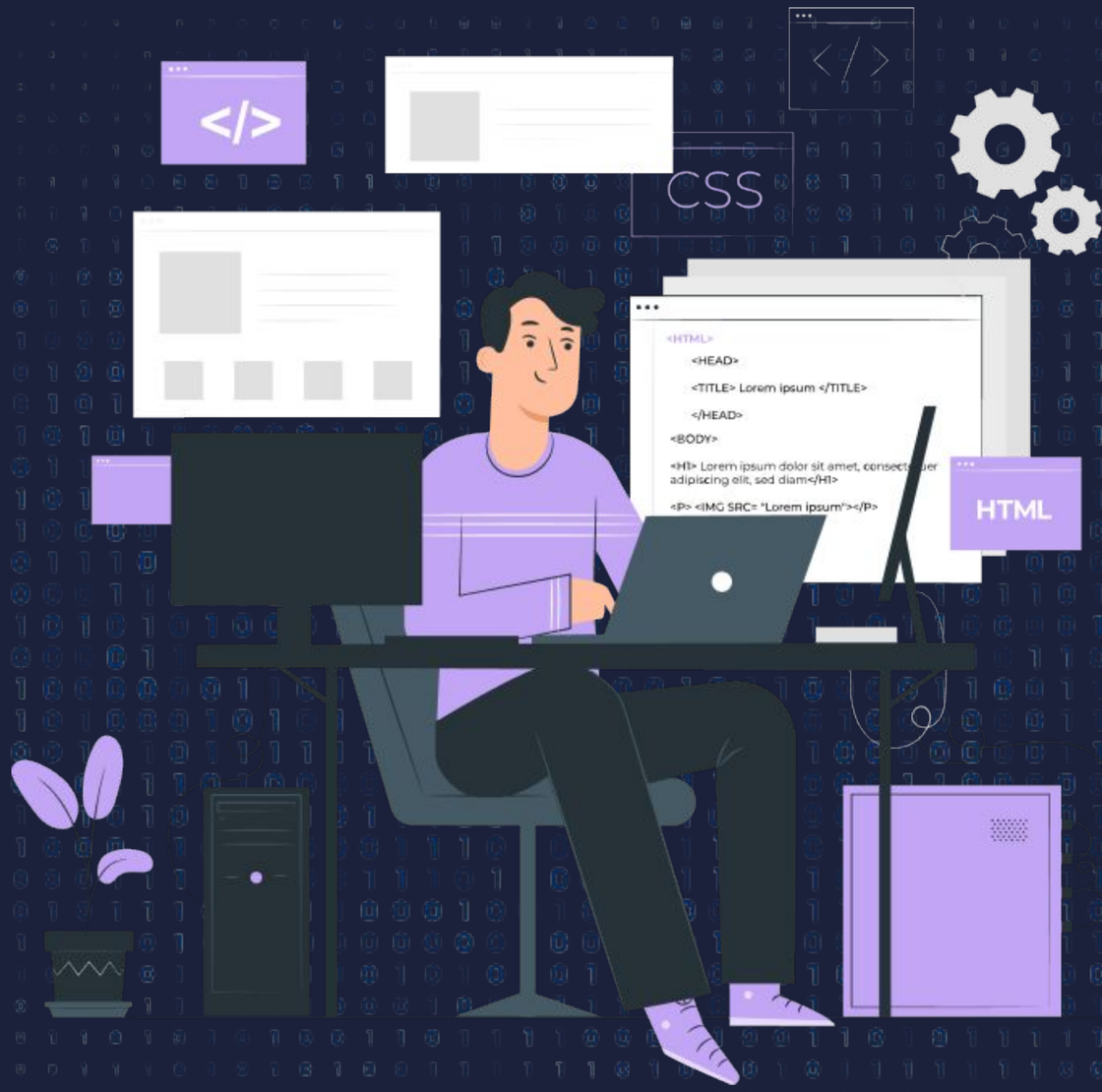
```
<link rel="icon" href="image/favicon.ico" />
```




SKILLS
THANK YOU

PW SKILLS

HTML Table



Topics Covered

- Introduction to <table> Tag
- Simple Table
- Expanding Cells
- Adding Caption
- Giving Structure to Table (thead, tfoot, tbody)

Simple Table

Let's say we have following Table Data

Name	Age	City
John	25	New York
Jane	30	Los Angeles
Michael	22	Chicago

Simple Table

Lets build table in HTML for given data step by step.

Step 1: Create Table headings

```
<table border="1">
    <tr>
      <th>Name</th>
      <th>Age</th>
      <th>City</th>
    </tr>
</table>
```

Name	Age	City

Simple Table

Step 2: Add one Row

```
<table border="1">
    <tr> <!— Heading —> </tr>
    <tr>
      <td>John</td>
      <td>25</td>
      <td>New York</td>
    </tr>
</table>
```

Name	Age	City
John	25	New York

Simple Table

Step 2: Similarly add all 3 Rows, and Table is Ready!

```
<table border="1">
  <tr>
    <th>Name</th>
    <th>Age</th>
    <th>City</th>
  </tr>

  <tr>
    <td>John</td>
    <td>25</td>
    <td>New York</td>
  </tr>

  <tr> ... </tr>
  <tr> ... </tr>
</table>
```

Name	Age	City
John	25	New York
Jane	30	Los Angeles
Michael	22	Chicago

Expanding Cells - colspan, rowspan

```
<table border="1">
  <tr>...</tr>
  <tr>...</tr>
  <tr>...</tr>
  <tr>...</tr>
  <tr>...</tr>
  <tr>
    <td colspan="2">Sum</td>
    <td>$110.00</td>
  </tr>
</table>
```

Product Name	Brand	Price
Product 1	Brand 1	$10.00
Product 2	Brand 2	$20.00
Product 3	Brand 3	$30.00
Product 4	Brand 4	$50.00
Sum		$110.00

Adding a caption

```
<table border="1">
  <caption>
      Products Data
  </caption>

  <tr>...</tr>
  <tr>...</tr>
  <tr>...</tr>
  <tr>...</tr>
  <tr>...</tr>
</table>
```

Products Data

Product Name	Brand	Price
Product 1	Brand 1	$10.00
Product 2	Brand 2	$20.00
Product 3	Brand 3	$30.00
Product 4	Brand 4	$50.00
Sum		$110.00

Giving Structure to Table

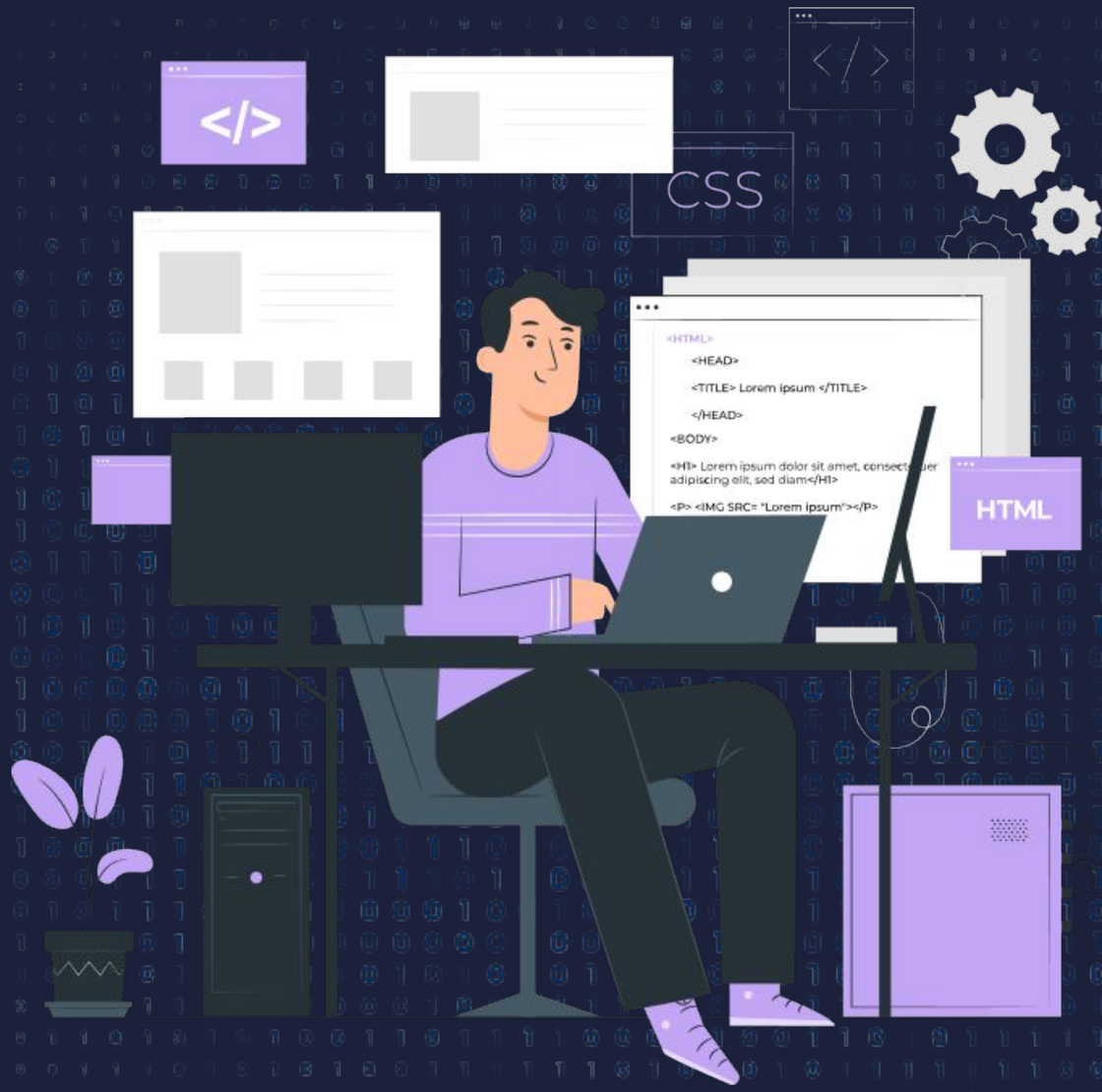
- caption
- thead
- tbody
- Tfoot

Note: Above tags does not add any semantic behaviour to the page.

```
<table border="1">
        <caption>...</caption>

        <thead>
            <tr>...</tr>
        </thead>

        <tfoot>
          <tr>...</tr>
        </tfoot>

        <tbody>
          <tr>...</tr>
          <tr>...</tr>
        </tbody>
</table>
```


SKILLS
THANK YOU

Bring in Media



Topics Covered

- What is media and benefits of using media in HTML
- Adding Image using <img> Tag
- Adding Audio using <audio> Tag
- Adding Video using <video> Tag
- Embed another Document using <iframe> Tag

What is media and benefits of using media in HTML

In HTML, "media" refers to any content that can be displayed or played back on a web page, such as images, audio files, and video files.

Benefits :

- Make web pages more engaging, interesting, and visually appealing.
- Convey information more effectively, make applications interactive.
- Providing alternative means of consuming information, such as audio, videos for specially abled users.

Adding Image using <img> Tag

```
<img
  src="https://picsum.photos/200"
  alt="image"
/>
```

Image Tag Attributes

src : URL of an image file

alt : Alternative text for an image in case the image cannot be displayed.

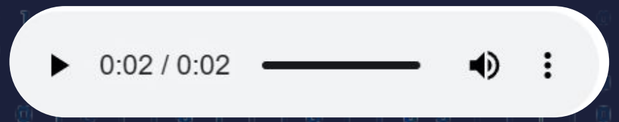
width and height: Specify the width and height of the image in pixels

Supported Image Formats

- **APNG (Animated Portable Network Graphics) -** Good choice for lossless animation sequences
- **AVIF (AV1 Image File Format) -** Good choice for both images and animated images due to high performance.
- **GIF (Graphics Interchange Format) -** Good choice for simple images and animations.
- **JPEG (Joint Photographic Expert Group image) -** Good choice for lossy compression of still images
- **PNG (Portable Network Graphics) -** Good choice for lossless compression of still images
- **SVG (Scalable Vector Graphics) -** Vector image format. Use for images that must be drawn accurately at different sizes.
- **WebP (Web Picture format) -** Excellent choice for both images and animated images.

Adding Audio using <audio> Tag

```
<audio
    Controls
      src="https://interactive-examples.mdn.mozilla.net
      /media/cc0-audio/t-rex-roar.mp3">
</audio>
```



src: URL or path of of audio

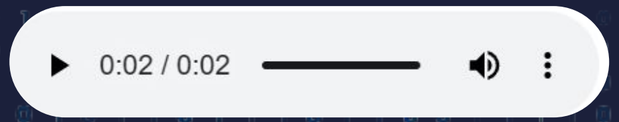
controls: show audio controls, such as play, pause, and seek

Other Audio Attributes

autoplay: Audio will start playing automatically

loop: Audio will repeat in loop

```
<audio  src="audio.mp3"  autoplay  loop controls >
</audio>
```



Other Audio Attributes

preload: It can have values like,

- **auto -** loads the audio file automatically
- **metadata -** loads only the metadata of the audio file
- **none -** does not preload the audio file

controlslist: It can have values like

- **nodownload -** disables download button
- **nofullscreen -** disables fullscreen button
- **nodownload nofullscreen -** disables both download and fullscreen buttons

muted: When this attribute is present, the audio will be muted by default.

Audio with multiple sources

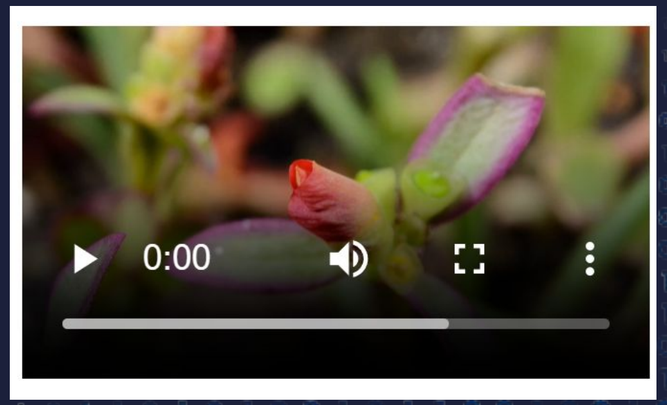
<source> tag is used to specify multiple sources of audio formats.

Using multiple **<source>** tags in the **<audio>** tag allows for the audio file to be played in different formats on different browsers and devices.

```
<audio controls>
  <source src="sample.opus" type="audio/ogg; codecs=opus" />
  <source src="sample.ogg" type="audio/ogg; codecs=vorbis" />
  <source src="sample.mp3" type="audio/mpeg" />
</audio>
```

Adding Video using <video> Tag

```
<video
    src="video.mp4"
    controls>
</video>
```



src: URL or path of of Video

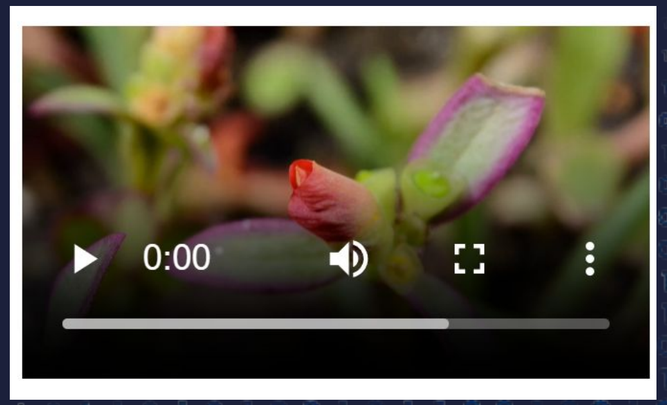
controls: video controls, such as play, pause, and seek

Other Audio Attributes

autoplay: Video will start playing automatically

loop: Video will repeat in loop

```
<video
    src="video.mp4"
    autoplay
    loop>
</video>
```



Other Audio Attributes

preload: It can have values like,

- **auto** - loads the video file automatically.
- **metadata** - loads only the metadata of the video file.
- **none** - does not preload the video file.

controlsList: This attribute specifies the controls that should be displayed in the video player's controls. It can have values like,

- **nodownload** - disables download button.
- **nofullscreen** - disables fullscreen button.
- **nodownload nofullscreen** - disables both download and fullscreen buttons

muted: When this attribute is present, the audio will be muted by default.

poster: This attribute specifies an image URL that should be displayed as a poster frame before the video starts playing.

width and height: These attributes specify the width and height of the video player in pixels.

Video with multiple sources

<source> tag is used to specify multiple sources of video formats.

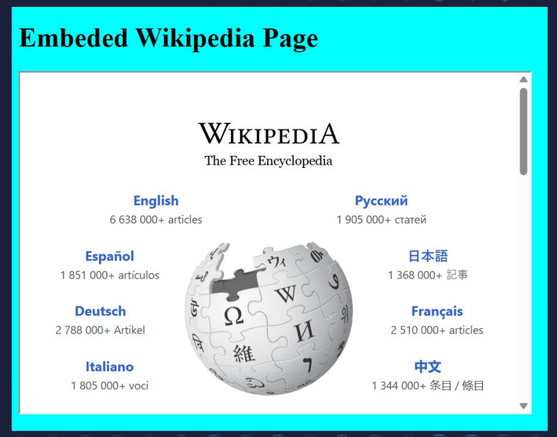
multiple <source> tags in the <video> tag allows for the video file to be played in different formats on different browsers and devices.

```
<video width="620" controls>
    <source src="https://archive.org/download/ElephantsDream/ed_hd.ogv " type="video/ogg" />
    <source src="https://archive.org/download/ElephantsDream/ed_hd.avi " type="video/avi" />
   <source src="https://archive.org/download/ElephantsDream/ed_1024_512kb.mp4 ” type="video/mp4" />
  Sorry, your browser doesn't support embedded videos.
</video>
```

Embed another Document using <iframe> Tag

Embed Document Inside Another Document

```
<iframe
  src="https://www.wikipedia.org/"
  frameborder="1"
  width="600" height="400">
</iframe>
```



Iframe Attributes

src: URL of the web page that you want to embed.

frameborder: A value of "0" means no border, and "1" means to display a border.

width and height: These attributes specify the width and height of the <iframe>, respectively.

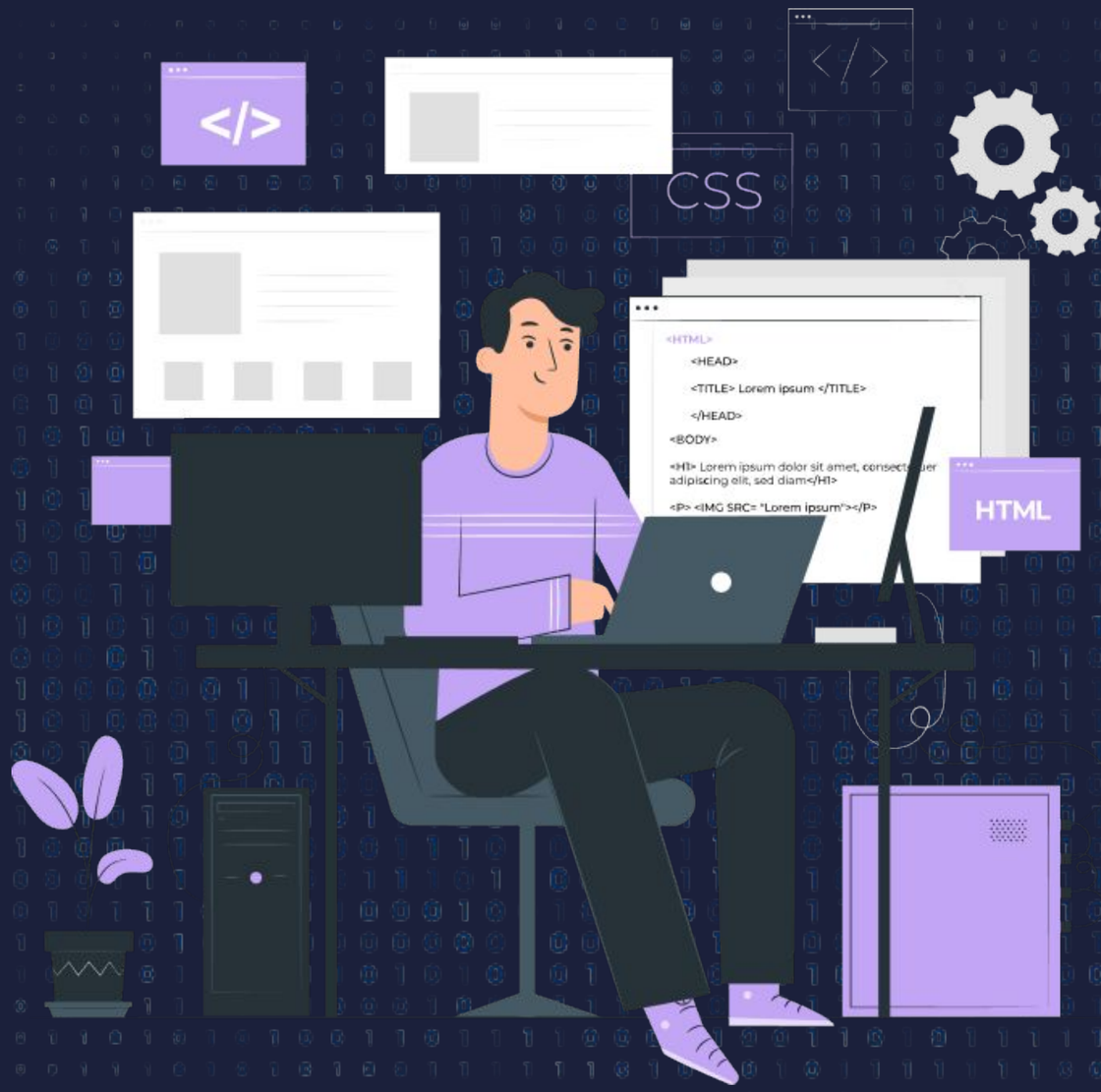
allow: It can be used to enable specific permissions, such as allowing full screen mode or access to the microphone.

scrolling: This attribute specifies whether or not to display scrollbars within the <iframe>. It can have values of "yes", "no", or "auto".


SKILLS
THANK YOU

PW SKILLS

Input tags and their attributes



Topics Covered

- Input
- Input Attributes
- Common Input Types
- Legend
- Code example

Input

The **<input>** HTML element defines Interactive controls.

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

Here **<label>** tags help users to know the purpose of the form control element (input).

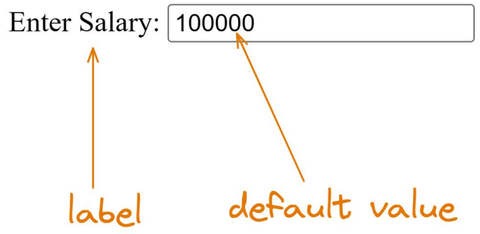
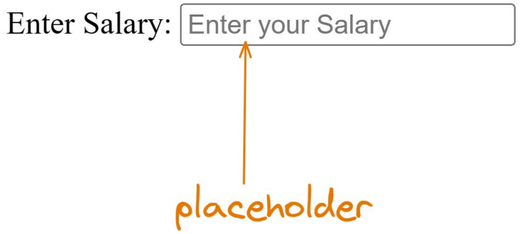
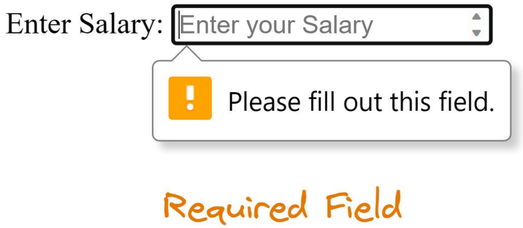
for attribute specifies the target control element, which allows users to click on the label to focus on the associated form control.

Input Attributes

- **type:** Type of Input Control
- **name:** Name of Input Control
- **value:** Default value of Input
- **placeholder:** Hint for Input Control
- **required:** Specify input required to be filled

Example of input attribute

```
<form action="/subscribe" method="get">
    <label for="email">Enter Email: </label>
    <input
       type="number"
       name="salary"
       placeholder="Enter your Salary"
       value="100000"
       required
    >
</form>
```

Initially Input will look like below	When the input is empty, we can see placeholder value.	When a user tries to submit, without filing a value.
		

Text

Accepts **Single Line Text**

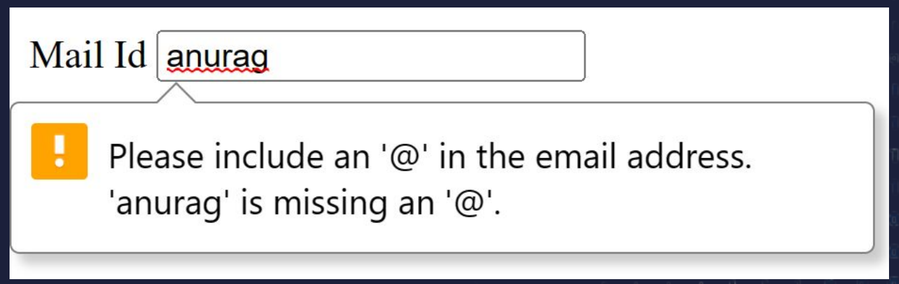
```
<label for="mailid">Mail Id</label>
<input type="text" placeholder="Text Input" id="mailid"
name="mailid" required/>
```

Text Input Write Here

Email

Accepts **Email Address** Only and throws the warming message ,if we try to submit the form with invalid email address

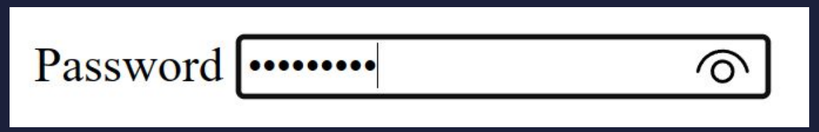
```
<label for="mailid">Mail Id</label>
<input type="text" placeholder="Text Input" id="mailid"
name="mailid" required/>
```



Password

Allows used to **Enter Password Securely**

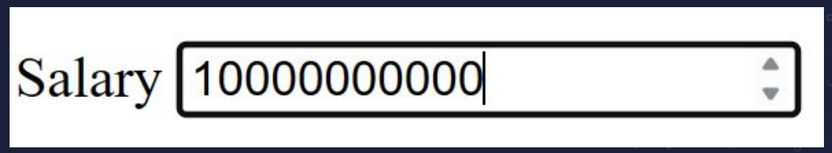
```
<label for="pass">Password</label>
<input type="password" id="pass" name="pass"/>
```



Number

Accepts **Numbers Only**

```
<label for="salary">Salary</label>
<input type="number" id="salary" name="salary"/>
```



Radio

Allows user to **Select One Options.**

```
<label for="option1">option1</label>
<input type="radio" id="option1" name="selected" value="option1"/>

<label for="option2">option2</label>
<input type="radio" id="option2" name="selected" value="option2"/>

<label for="option3">option3</label>
<input type="radio" id="option3" name="selected" value="option3"/>
```

option1 ○ option2 ◉ option3 ○

Note: Don't forget to give value in-case of checkbox and radio buttons.

Select and Options

Allows users to **Select Option from a Dropdown**

```
<select name="selected">
    <option value="Asus">Asus</option>
    <option value="Lenovo">Lenovo</option>
    <option value="Apple">Apple</option>
    <option value="Dell">Dell</option>
</select>
```

TextArea

Accepts **Multiple Line Text**

```
<textarea name="selected" rows="5" cols="33"> </textarea>
```

Mutiple
Line
Input

Details

Disclose Widget, **Shows Info in Open State**

```
<details>
    <summary>Topic Name</summary>
    Topic Details Here
</details>
```

Close State	Open State
▶ Topic Name	▼ Topic Name Topic Details are Here

Submit

Allows to **Submit Information** to the server.

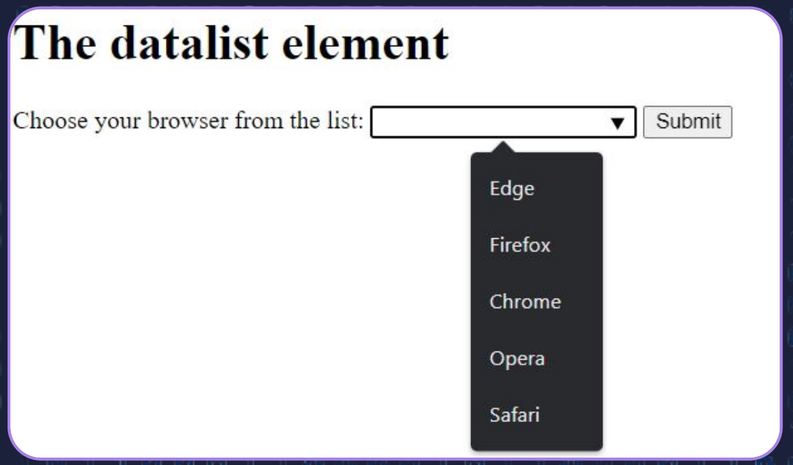
```
<input type="submit" />
```

Submit

Datalist

The <datalist> tag is used to provide an "autocomplete" feature for <input> elements. Users will see a drop-down list of pre-defined options as they input data.

```
<form>
 <input list="browsers" name="browser" id="browser">
 <datalist id="browsers">
    <option value="Edge">
    <option value="Firefox">
    <option value="Chrome">
    <option value="Opera">
    <option value="Safari">
  </datalist>
  <input type="submit">
</form>
```



Legend

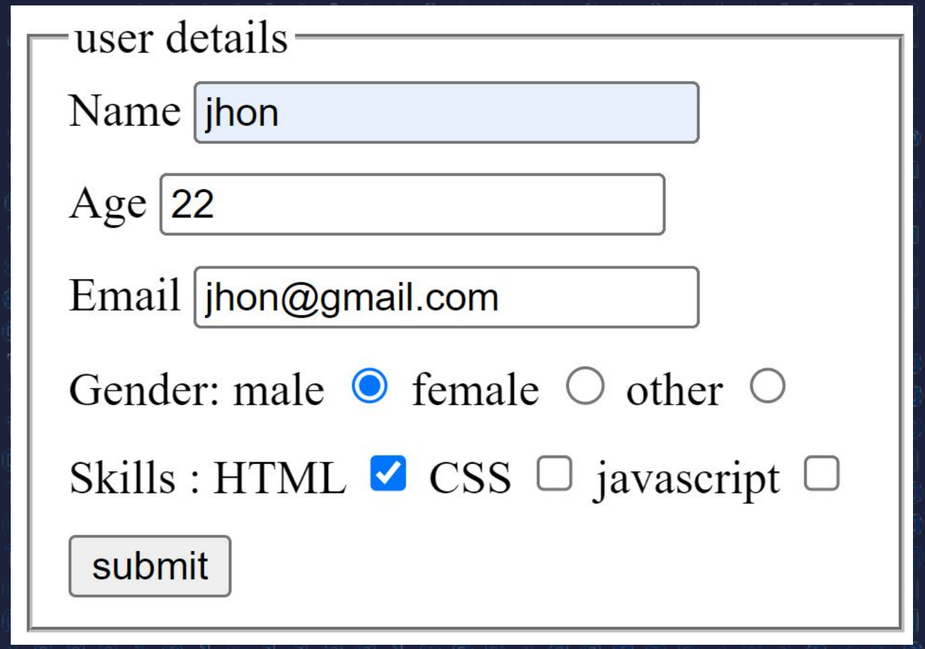
The **<legend>** HTML element represents a caption for the content of its parent **<fieldset>**.

```
<fieldset>
    <legend>Choose your favorite monster</legend>

    <input type="radio" id="kraken" name="monster" value="K">
    <label for="kraken">Kraken</label><br>

    <input type="radio" id="sasquatch" name="monster" value="S">
    <label for="sasquatch">Sasquatch</label><br>

    <input type="radio" id="mothman" name="monster" value="M">
    <label for="mothman">Mothman</label>
</fieldset>
```

Code example

```
        <fieldset>
      <legend>user details</legend>
<form action="#" method="GET">
        <!— name (input text) —>
        <div>
          <label for="name"> Name </label>
          <input type="text" name="Name"  id="name"  required  placeholder="Name"   />
        </div>
        <!— age (input number)—>
        <div>
          <label for="age">Age</label>
          <input type="number" name="Age" id="age" required placeholder="Age" />
        </div>
<!— email (input email)—>
        <div>
          <label for="email">Email</label>
          <input   type="email"  name="Email" id="email"  required  placeholder="Email" />
        </div>
        <!— gender (input radio) —>
        <div>
          Gender:
          <label for="male">male</label>
          <input type="radio" id="male" name="Gender" value="male" />

          <label for="female">female</label>
          <input type="radio" id="female" name="Gender" value="female" />

          <label for="other">other</label>
          <input type="radio" id="other" name="Gender" value="other" />
        </div>
        <!— skills  (input checkbox)  —>
        <div>
          Skills :
          <label for="HTML">HTML</label>
          <input type="checkbox" id="HTML" name="skill" value="HTML" checked/>

          <label for="CSS">CSS</label>
          <input type="checkbox" id="CSS" name="skill" value="CSS" />

          <label for="javascript">javascript</label>
          <input type="checkbox" id="javascript" name="skill" value="javascript"  />
        </div>
        <!— submit —>
        <input type="submit">
      </form>
      </fieldset>
```

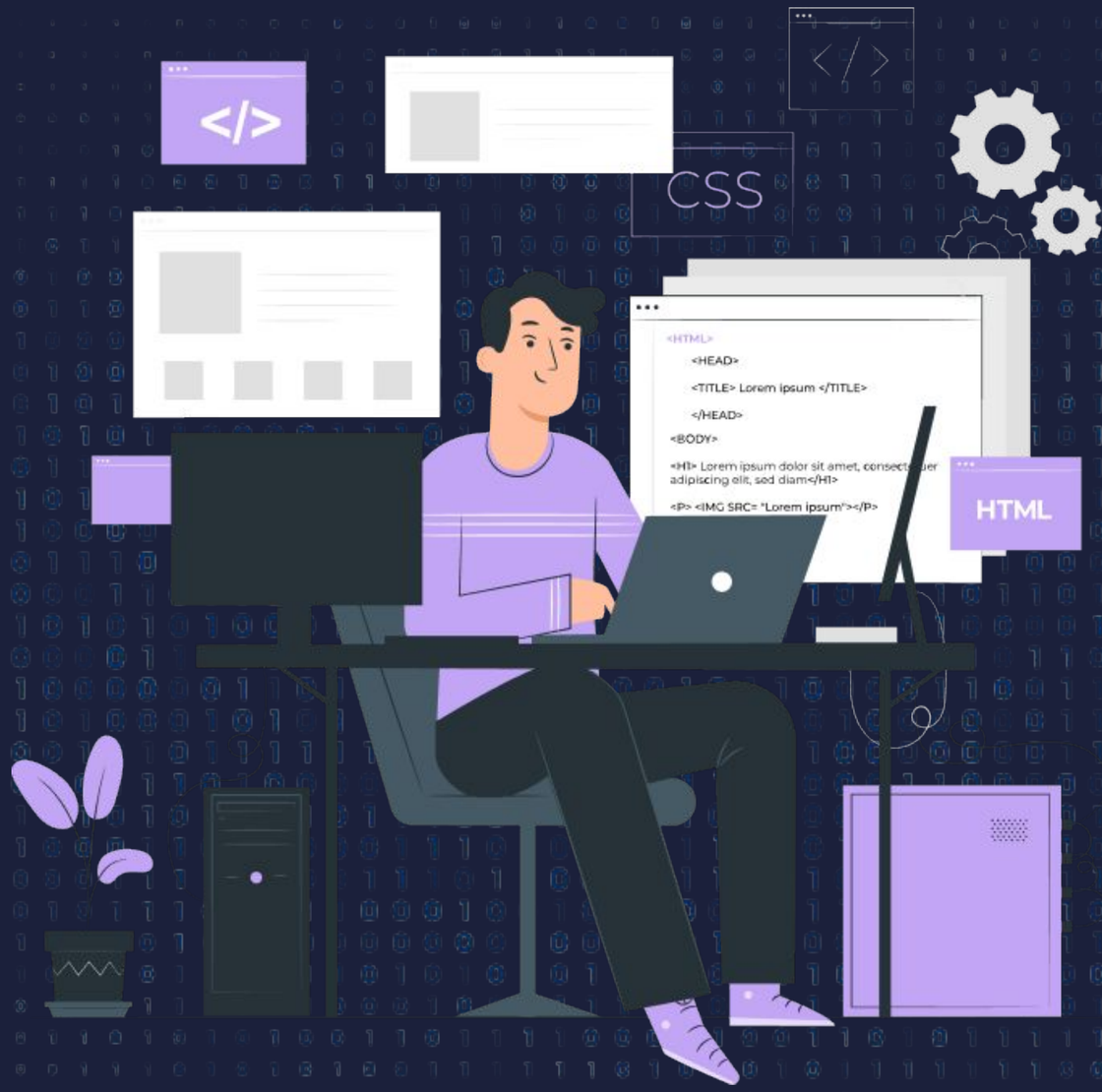
Browser output




SKILLS
THANK YOU

PW SKILLS

Html Forms



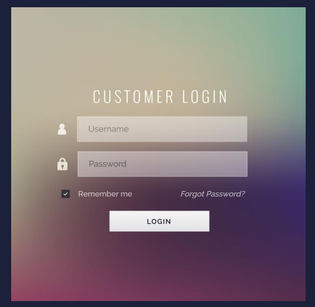
Topics Covered

- Introduction to HTML form
- Form attributes and their usage
- Login Form Example

Introduction to HTML form

Take Input From User

Submitting Information to Server



Login Form

Send Login Form Data to Server

Server

Lets Create a Simple Form

```
<form action="/subscribe" method="get">
  <!-- input controls will come here --->
</form>
```

action - URL to which form data sent for processing

method - HTTP Method used to submit data

Email Input control

```
<label for="email">Enter Email: </label>
<input type="email" name="email" required>
```

<input> tag defines input element for entering an email address. It also checks for a valid email address.

<label> tag helps users to know the purpose of the form control element (input).

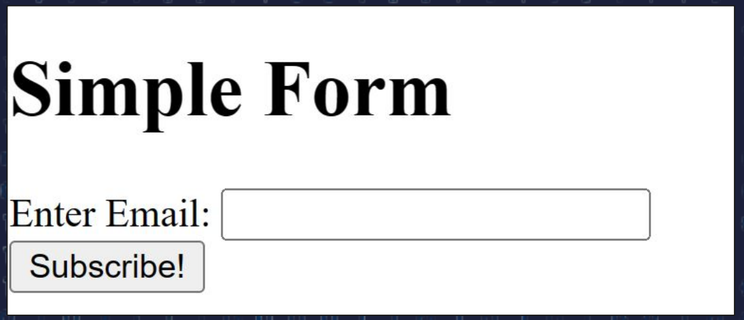
Submit Control

```
<div>
    <input type="submit" value="Subscribe!">
</div>
```

<input type="submit"> defines a submit controls which allows user to **submit form data** on clicking the button.

Let’s combine Form, Input control and Submit control

```
<form action="/subscribe" method="get">
  <label for="email">Enter Email: </label>
  <input type="email" name="email" id="email" required>
  <div>
    <input type="submit" value="Subscribe!">
  </div>
</form>
```



Form attributes and their usage

The **action** and **method** attributes are used to specify how the form data will be submitted to the server for processing.

- **action:** Defines the URL to which the form data will be sent for processing.
- **method:** HTTP method that will be used to submit the form data to the server. Examples GET, POST etc.

GET method, after submitting the form, form data is visible in the address bar. But the

POST method prevents form data from appearing in the address bar after the form has been submitted, via sending data in the request body.

Login Form Example

Lets create a simple login form using html form.

Step 1: Create an html file “index.html” with below template code.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Login</title>
  </head>
  <body>
    <h1>Login</h1>
  </body>
</html>
```

Login Form Example

Step 2 : Create an empty form element inside the html body.

```
<body>
    <h1>Login</h1>
    <form method="POST">
          <!-- input controls will come here -->
    </form>
</body>
```

Here, **<form> Tag** defines a form element, with method **POST, as discussed previously, POST** method prevents form data from appearing in the address bar after the form has been submitted, via sending data in the request body.

Login Form Example

Step 3 : Add username input with label

```
<body>
<form method="POST">
   <label for="username">Username:</label>
   <input
      type="text"
      id="username"
      name="username"
      placeholder="Enter your username"
      required
   />
<form>
</body>
```

Username is a required input!

Login Form Example

Step 4 : Add password input with label

```
<body>
<form method="POST">
    ...
   <label for="password">Password:</label>
   <input
      type="password"
      id="password"
      name="password"
      placeholder="Enter your password"
      required
   />
<form>
</body>
```

Password is also a required input!

Enter password securely!

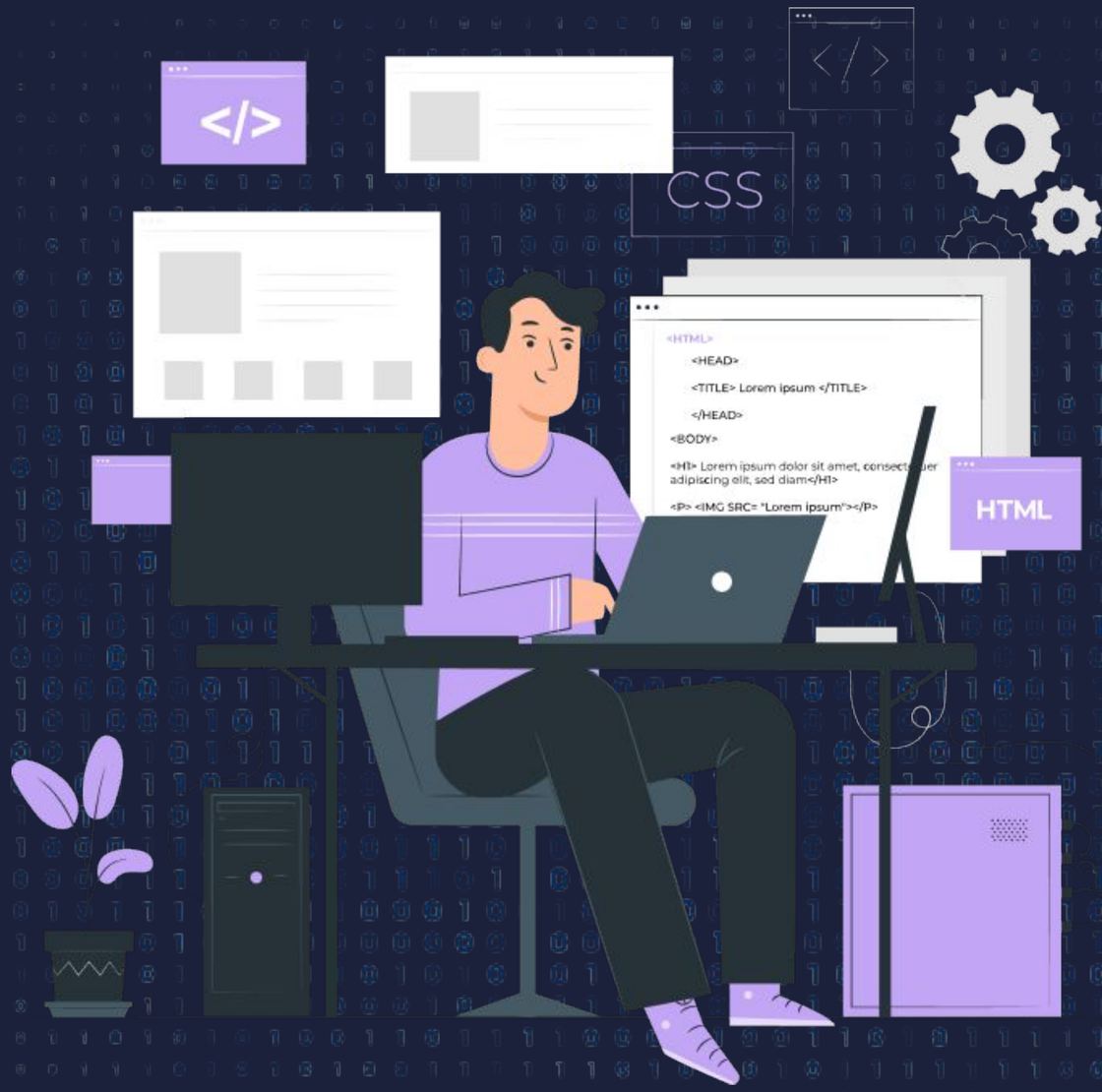
Login Form Example

Step 5 : At last add “login” submit input

```
<body>
<form method="POST">
    ...
    <input type="submit" value="Login" />
<form>
</body>
```


SKILLS
THANK YOU

History of HTML and Features of HTML5



TOPIC

- History of HTML
- Features of HTML5
- HTML vs HTML5

History of HTML

HTML

HTML was initially introduced in late 1991, and since then HTML has undergone many changes.

1991

HTML 2

This version introduced tables and form elements.

1995

HTML 3

This version introduced support for style sheets and tables with merged cells

January 1997

HTML 4

This version introduced support for frames, scripts, and improved forms.

December 1997

HTML 5

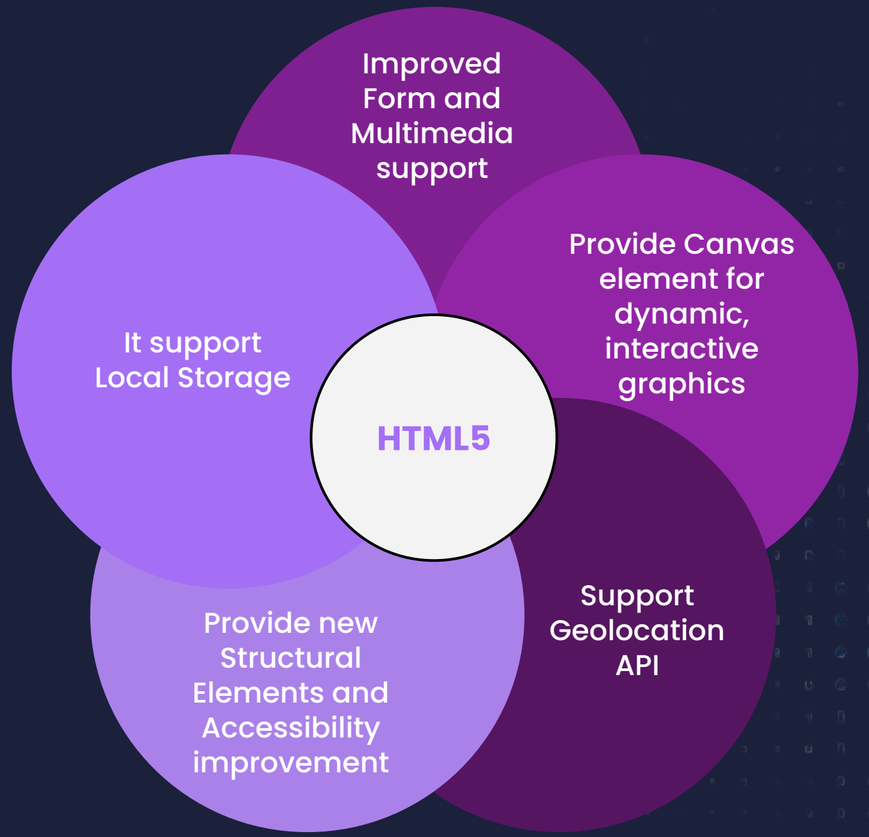
This is the latest version of HTML and includes several new features and improvements over its predecessors, such as better support for multimedia elements, improved semantic markup, and better mobile support.

2014

History of HTML

Type of content	HTML 1.2	HTML 4.01	HTML5	Purpose
Heading	Yes	Yes	Yes	Organize page content by adding headings and subheadings to the top of each section of the page
Paragraph	Yes	Yes	Yes	Identify paragraphs of text
Address	Yes	Yes	Yes	Identify a block of text that contains contact information
Anchor	Yes	Yes	Yes	Link to other web content
List	Yes	Yes	Yes	Organize items into a list
image	Yes	Yes	Yes	Embed a photograph or drawing into a web page
table	No	Yes	Yes	Organize data into rows and columns
style	No	Yes	Yes	Add CSS to control how objects on a web page are presented
script	No	Yes	Yes	Add Javascript to make pages respond to user behaviours (more interactive)
Audio	No	No	Yes	Add audio to a web page with a single tag
Video	No	No	Yes	Add video to a web page with a single tag
Canvas	No	No	Yes	Add an invisible drawing pad to a web page, on which you can add drawings (animations, games, and other interactive features) using Javascript

Features of HTML5



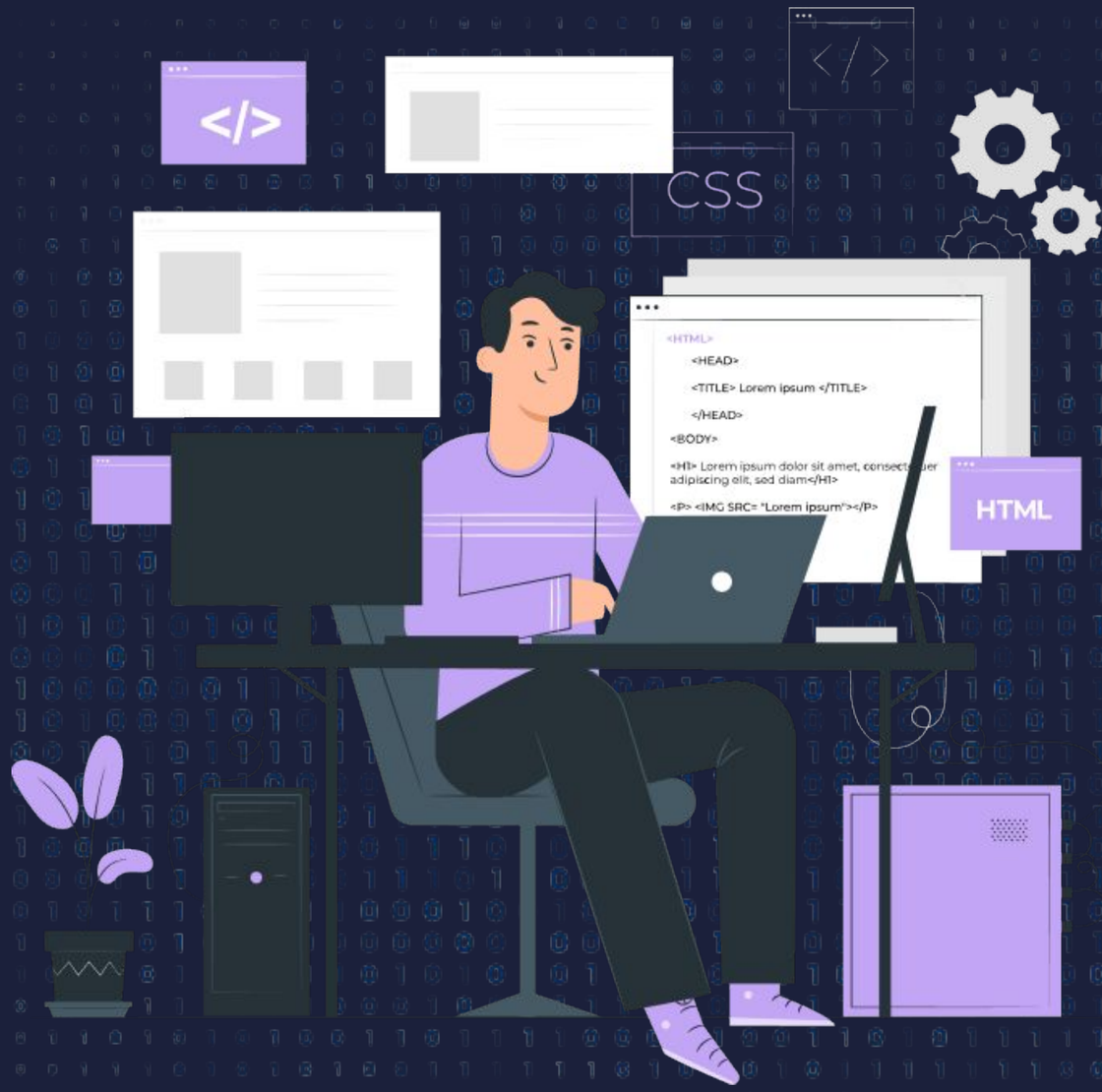
HTML vs HTML5

HTML	HTML5
It required plugins like Adobe Flash to support audio and video content.	Provides built in support for multimedia elements such as **video** and **audio** without need for plugin
It has less elements as compared to HTML5	It includes new elements and form attributes such as **time, date, and color**. Required and **autofocus** in input types of the tag element as well.
It does not have support for local storage.	It has support for **local storage** i.e localStorage.
Less semantic elements, thereby provides less web accessibility features.	It support more semantic element, such as **<header>, <footer>, <nav> and <article>** etc., thereby Improved accessibility


SKILLS
THANK YOU

PW SKILLS

Introduction to HTML entities



Topics

- What is HTML Entities
- Why use HTML Entities
- Common HTML Entities

What are HTML Entities

The HTML Entities are strings of text that start with an ampersand symbol(&) and end with a semicolon(;).

These entities are commonly utilized to demonstrate characters that are reserved and imperceptible characters such as non-breaking spaces.

Format of HTML Entities

&code;

Where code may be a

- Decimal form like **&#203**
- A hex form like **&#x00CB**, or stripped off leading zeros, simply **&xCB**
- A named value if available, such as **&pound**

```
<p>Numeric entity decimal: &#163</p>
<p>Numeric entity hex: &#x00A3</p>
<p>Named entity : &pound;</p>
```

Numeric entity decimal: £

Numeric entity hex: £

Named entity : £

Why use HTML Entities

- Displaying special characters
- Avoiding conflicts with HTML syntax
- Supporting different character sets
- Improving accessibility

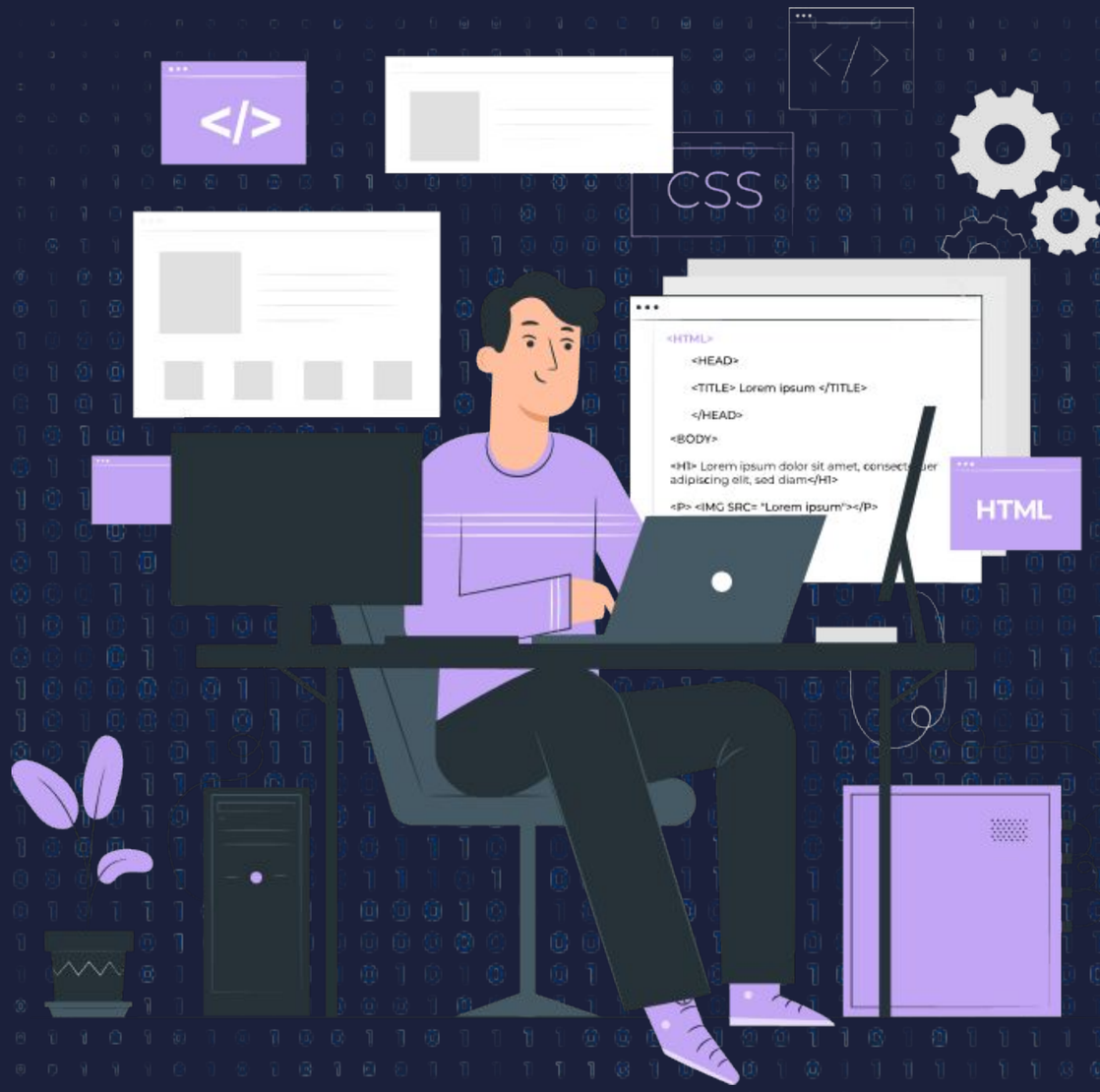
Why use HTML Entities

Named Entity	HTML5 Alias	Numbered Entity	Description	Intended Rendering
&amp;	&AMP;	&#38;	Ampersand	&
&copy;	&COPY;	&#169;	Copyright	©
&gt;	&GT;	&#62;	Greater than	>
&lt;	&LT;	&#60;	Less than	<
&quot;	&QUOT;	&#34;	Double quotes	“
&reg;	&REG;	&#174;	Registration mark	®
&trade;	&TRADE;	&#8482;	Trademark symbol	™


SKILLS
THANK YOU

PW SKILLS

Accessibility in HTML

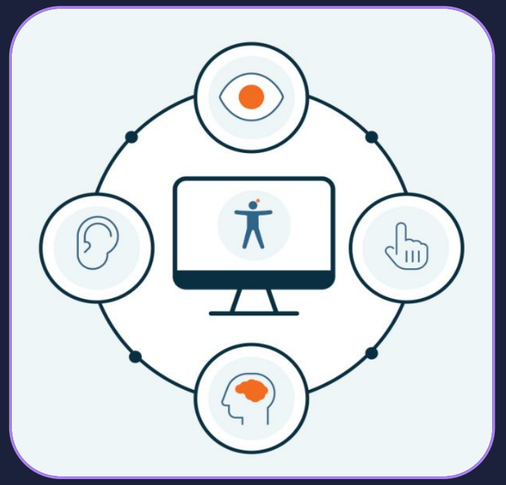


Topics

- What is Web Accessibility?
- Several ways to Improving accessibility in html.
- Keyboard accessibility

What is web accessibility?

Web accessibility refers to the practice of designing and developing websites, web applications, and other digital content that can be accessed and used by people with disabilities or different needs, without barriers or limitations.



Assistive Devices

There are some assistive devices which play a major role in providing accessibility.

1. **Screen Reader :** A screen reader is a software that reads out loud the content of a web page to individuals who are visually impaired. It can also interpret and communicate information about graphics, multimedia, and other elements on the page.

2. **Voice recognition software:** Voice recognition software enables users to navigate web pages and input text using voice commands. This technology is particularly useful for individuals with mobility impairments or those who have difficulty using a keyboard or mouse.

3. **Keyboard alternatives:** Keyboard alternatives such as sip-and-puff devices, head-tracking devices, and eye-tracking devices allow individuals with physical disabilities to navigate and interact with web pages without the use of a traditional keyboard or mouse.

Text content Accessibility

The **screen reader reads** each content out as you progress through the content, notifying you **what is heading** and **what is a paragraph**.

```
<h2>My subheading</h2>
<p>
  This is the first subsection of my document. I'd love people to be able to find
this content!
</p>
<h2>My 2nd subheading</h2>
<p>
  This is the second subsection of my content, which I think is more interesting
than the last one.
</p>
```

Page layouts Accessibility

Utilise proper sectioning elements to encapsulate your main navigation (<nav>), footer (<footer>), repeated content units (<article>), and other relevant content.

These elements offer additional semantics to screen readers and other assistive tools, providing users with more context and information about the content they are browsing.

```
<header>
  <h1>Header</h1>
</header>

<nav>
  <!— main navigation in here —>
</nav>

<!— Here is our page's main content —>
<main>
  <article>
    <h2>Article heading</h2>
    <!— article content in here —>
  </article>
</main>

<footer>
  <!— footer content in here —>
</footer>
```

UI Control Accessibility

Define links, buttons, form elements, appropriately with proper labels using **<label>** tag.

```
<— Link --!>
<p>This is a link to <a
href="https://www.mozilla.org">Mozilla</a>.</p>

<— buttons -!>
<p>
  <button data-message="First button">Click here!</button>
  <button data-message="Second button">Click right here!</button>
  <button data-message="Third button">Submit!</button>
</p>
```

Alt attribute

The **alt** attribute is an important attribute in HTML that is used to provide alternative text for an image if the image cannot be displayed or if the user is using a screen reader to access the page.

```
<img src="sample.jpg" alt="A sample image">
```

Title attribute

The title attribute is an important attribute in HTML that can be used to provide additional information about an element, such as a link or an image.

```
<h1 title="This is the h1 tag">Hover me</h1>
```

Keyboard Accessibility

Those who cannot use a mouse, are able to navigate and interact with web content using only a keyboard.

```
<div data-message="First button" tabindex="0" role="button">
  Click here!
</div>
<div data-message="Second button" tabindex="0" role="button">
  Click right here!
</div>
<div data-message="Third button" tabindex="0" role="button">
  Submit here
</div>
```

Keyboard Accessibility

Basically, the **tabindex** attribute is primarily intended to allow tabbable elements to have a custom tab order (specified in positive numerical order), instead of just being tabbed through in their default source order.

There are two additional options available for tabindex:

1. **tabindex="0"** - Allows elements that are not usually able to be focused via the keyboard to become focusable.

2. tabindex="-1" - Enables elements that are not typically focusable to receive focus programmatically, such as through JavaScript, or as the target of links.

ARIA

ARIA stand for The Accessible Rich Internet Applications (ARIA)

It comprises **roles** that establish methods for improving the accessibility of web content and web applications, for individuals with disabilities.

Example 1 - List roles **“list”, “listitem”**

```
<div role="list">
  <div role="listitem">List item 1</div>
  <div role="listitem">List item 2</div>
  <div role="listitem">List item 3</div>
</div>
```

For content that comprises an external container enclosing a group of items within it, assistive technologies can recognize the “**list**” and “**listitem**” containers, respectively.

ARIA

Example 2 - **img role**

```
<div role="img" aria-label="Description of the overall image">
  <img src="image1.png" alt="alt text"/>
  <img src="image2.png" alt="alt text" />
</div>
```

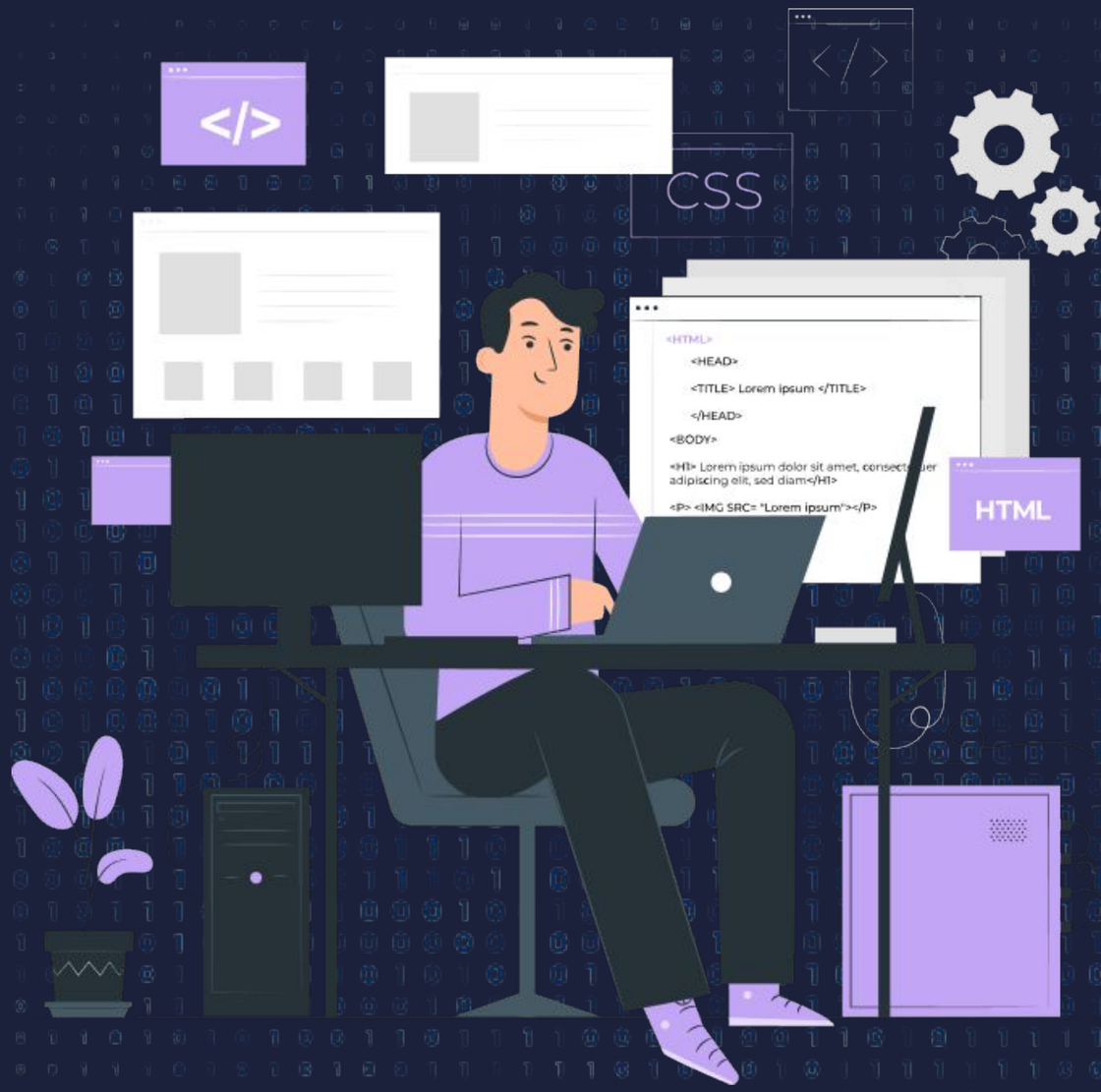
The ARIA **img role** can be used to identify multiple elements inside page content that should be considered as a single image.

Similarly, we have other ARIA roles like link roles, grid roles, form roles and many more, we use and study about them as per our requirement.


SKILLS
THANK YOU

PW SKILLS

Semantics and its Importance



Text content Accessibility

The **screen reader reads** each content out as you progress through the content, notifying you **what is heading** and **what is a paragraph**.

```
<h2>My subheading</h2>
<p>
  This is the first subsection of my document. I'd love people to be able to find
this content!
</p>
<h2>My 2nd subheading</h2>
<p>
  This is the second subsection of my content, which I think is more interesting
than the last one.
</p>
```

Topics

- Introduction to semantic tag
- Understanding semantic tags with example
- Benefits of semantic tags

Introduction to Semantic tags

Give meaning to the content of a web page.

Making it **Easier for Search Engines, Screen Readers** and others tools **to Understand the Page** and its content.

Example: <header> tag, <article> tag, <aside> tag and etc.

Understanding Semantic Elements

To understand semantics, we will build a web page using newly introduced structural semantic elements in HTML.

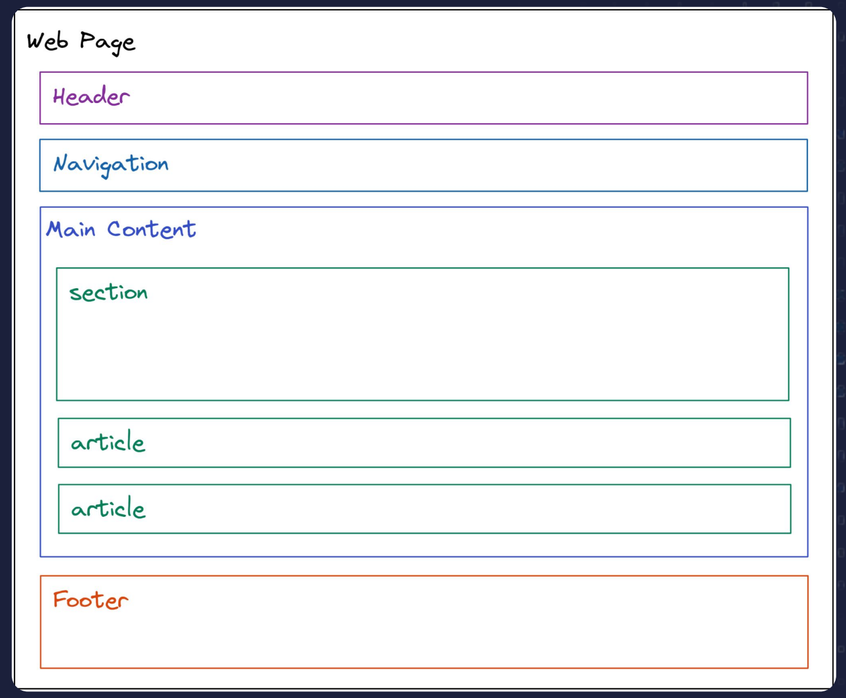
1. Header <header>
2. Navigation <nav>
3. Main Content <main>
4. Sections <section>
5. Articles <articles>
6. Footer <footer>
7. Heading <h1>..<h6>
8. Para <p>

Develop a web page using Semantic Elements

Unlike traditional HTML tags, semantic tags focus on the meaning of the content, rather than on the visual appearance.

Like in the right example, the **<header>** tag is a semantic tag indicating that the header section of a page begins.

Similarly other tags **<article>, <section>, <footer>** etc. have their semantic meaning.



Let’s build this web page part by part

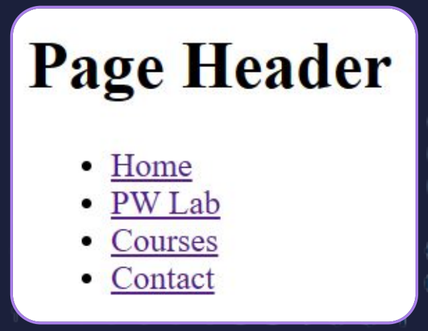
Create header

```
<header>
    <h1>Page Header</h1>
</header>
```

<header> specifies header for a document or any section

Adding Navigation

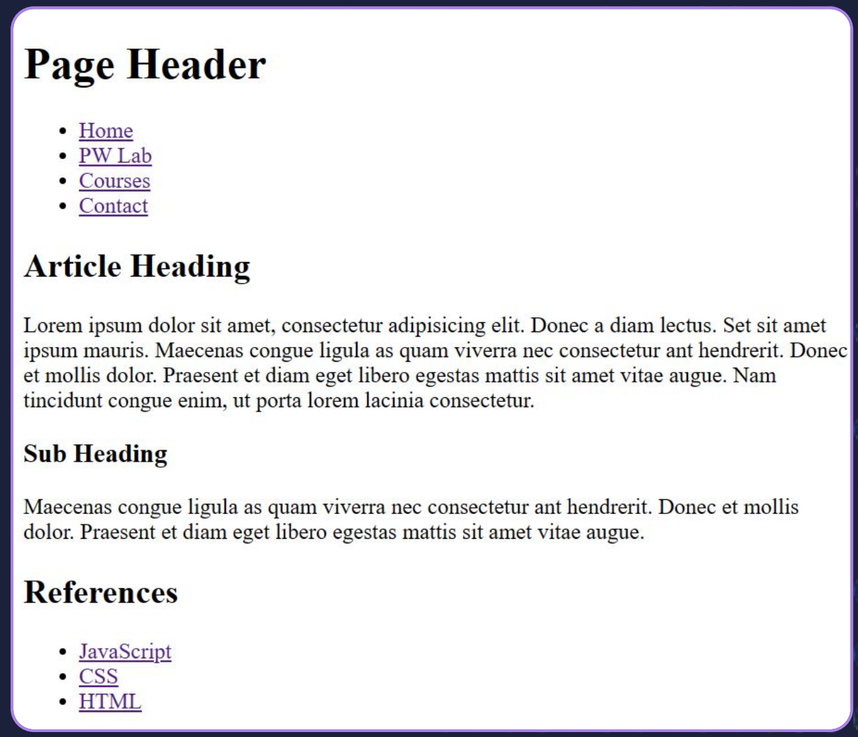
```
<header> ... </header>

<nav>
   <ul>
     <li><a href="#">Home</a></li>
     <li><a href="#">PW Lab</a></li>
     <li><a href="#">Courses</a></li>
     <li><a href="#">Contact</a></li>
   </ul>
</nav>
```



<nav> Defines navigation links

Adding Main content

```
<header> ... </header>
<nav> ... </nav>

<main>
   <article>
     <h2>Article Heading</h2>
     <p> ... </p>
       <h3>Sub Heading</h3>
     <p> ... </p>
   </article>

   <section>
     <h2>References</h2>
       <ul> ... </ul>
   </section>
</main>
```



<main> defines main content and **<article>, <section>** defines a part of main content with paragraph **<p>** and headings **<h1>**.

Adding Footer

```
<header> ... </header>
<nav> ... </nav>
<main> ... </main>

<footer>
    <p>
        ©Copyright 2089 by
        XYZ Company.
        All rights reversed.
      </p>
</footer>
```

Page Header

- Home
- PW Lab
- Courses
- Contact

Article Heading

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Donec a diam lectus. Set sit amet ipsum mauris. Maecenas congue ligula as quam viverra nec consectetur ant hendrerit. Donec et mollis dolor. Praesent et diam eget libero egestas mattis sit amet vitae augue. Nam tincidunt congue enim, ut porta lorem lacinia consectetur.

Sub Heading

Maecenas congue ligula as quam viverra nec consectetur ant hendrerit. Donec et mollis dolor. Praesent et diam eget libero egestas mattis sit amet vitae augue.

References

- JavaScript
- CSS
- HTML

©Copyright 2089 by XYZ Company. All rights reversed.

<footer> defines a footer for a document or section

Benefits of using Semantic tags

- Add additional information about the document.
- Helps screen readers to understand the content better.
- Provide **Search Engine Optimization**.


SKILLS
THANK YOU