"HTML DOCUMENTATION"

🔹 1. HTML Page Structure

1.<!DOCTYPE html>

- Defines the type of HTML document.
- Informs the browser that the document is written in **HTML5**.
- Must always be the **first line** in the document.

2. <html> ... </html>

- The **root element** of the HTML page.
- All other elements are contained inside this tag.
- It has two main parts:
    1. `<head>` – contains settings and metadata.
    2. `<body>` – contains the visible content of the page.

3. <head> ... </head>

Purpose: Stores meta information about the page.
Common elements inside `<head>`:

- `<title>` → Title shown in the browser tab.
- `<meta charset="UTF-8">` → Defines character encoding.
- **U**nicode **T**ransformation **F**ormat – **8-bit**
- `<meta name="viewport" ...>` → Makes the page mobile-responsive.
- `<meta name="description">` → Description for SEO.
- `<link>` → Links to an external CSS file.
- `<style>` → Internal CSS styling.
- `<script>` → JavaScript code or external JS file.

4. <body> ... </body>

Purpose: Contains the **visible content** of the web page.
Examples of elements inside `<body>`:

- Headings: `<h1>` to `<h6>`
- Paragraph: `<p>`
- Images: `<img>`
- Links: `<a>`
- Lists: `<ul>` / `<ol>`
- Tables: `<table>`

- Forms: `<form>`

HTML structure in vs code when (!+enter) called boiler code

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width,
initial-scale=1.0">

    <meta name="description" content="HTML Page Structure Notes">

    <title>HTML Structure</title>

</head>

<body>

    //

</body>

</html>
```

Note pad editor structure of HTML

```
<!DOCTYPE html>

<html>

<head>

    <title>My First Webpage</title>
```

```
</head>
<body>
    //
</body>
</html>
```

🔹 2. Text Formatting & Headings

Headings: `<h1>` to `<h6>`

- Define headings in decreasing order of importance.
- `<h1>` is the most important (usually the main title).
- `<h6>` is the least important.

Example:

```
<h1>Main Heading</h1>
<h2>Subheading</h2>
<h3>Smaller Subheading</h3>
```

Common Text Formatting Tags

Tag	Purpose	Notes
`<p>`	Defines a paragraph	Block element
`<b>`	Makes text bold (visual only)	No extra importance

`<strong>`	Makes text bold and adds semantic importance	Important text for SEO and accessibility
`<i>`	Italicizes text (visual only)	No extra importance
`<u>`	Underlines text	Visual effect, avoid misuse
`<span>`	Inline container for styling or scripting	No semantic meaning
`<div>`	Block container for grouping content	No semantic meaning

Special Text Elements

Tag	Purpose	Example
`<pre>`	Preformatted text (maintains whitespace and line breaks)	Useful for code blocks or ASCII art
`<blockquote>`	Block quotation (indented text)	For quoting other sources
`<code>`	Inline code snippet	Used to display code or commands

Example:

```
<pre>
  This is
  preformatted
  text.
</pre>

<blockquote>
  This is a quote from someone.
</blockquote>

<p>Use the <code>printf()</code> function in C.</p>
```

🔹 3. Links and Anchors

<a href="">

- The **anchor tag** creates a hyperlink.
- The `href` attribute defines the URL or link target.

Example:

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

`target="_blank"` & `rel="noopener"`

- `target="_blank"` opens the link in a **new tab/window**.
- Use `rel="noopener"` **for security and performance** when opening new tabs.

Example:

```
<a href="https://example.com" target="_blank" rel="noopener">Open in new tab</a>
```

Internal vs External Links

- **Internal links:** Point to pages **within the same website**.

Example:

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

- **External links:** Point to **different websites**.

Example:

```
<a href="https://google.com" target="_blank" rel="noopener">Google</a>
```

Anchor Links (Page Sections)

- Use `#sectionID` to link to a **specific part** of the same page.
- Requires an element with matching `id` attribute.

Example:

```
<a href="#contact">Go to Contact</a>

<!-- Somewhere else on the page -->
<h2 id="contact">Contact Us</h2>
```

🔹 4. Images

<img src="" alt="">

- Displays an image on the webpage.

- `src` attribute specifies the **image source path**.
- `alt` attribute provides **alternative text** if image fails to load (important for SEO & accessibility).

Example:

```
<img src="images/photo.jpg" alt="A beautiful sunset">
```

Image Paths

- **Relative path:** Path related to the current file location.

Example:

```
<img src="images/photo.jpg" alt="Sunset">
```

- **Absolute path:** Full URL or root-based path starting from domain root.

Example:

```
<img src="/assets/images/photo.jpg" alt="Sunset">
<img src="https://example.com/images/photo.jpg" alt="Sunset">
```

SEO-Friendly Image Usage

- Always use meaningful `alt` text describing the image.
- Use optimized image file sizes for faster loading.
- Use descriptive filenames for better SEO.

🔹 5. Lists

Ordered List `<ol>`

- Creates a **numbered** list.
- Items inside `<li>` tags.

Example:

```
<ol>
  <li>First item</li>
  <li>Second item</li>
</ol>
```

Unordered List `<ul>`

- Creates a **bulleted** list.
- Items inside `<li>` tags.

Example:

```
<ul>
  <li>First item</li>
  <li>Second item</li>
</ul>
```

Nested Lists

- Lists inside list items for sub-levels.

Example:

```
<ul>
  <li>Item 1
    <ol>
      <li>Sub-item 1</li>
      <li>Sub-item 2</li>
    </ol>
  </li>
  <li>Item 2</li>
</ul>
```

🔹 6. Tables

Basic Tags

- `<table>` — defines the table
- `<tr>` — table row
- `<td>` — table data (cell)
- `<th>` — table header (bold & centered)

Example:

```
<table>
  <tr>
    <th>Header 1</th>
    <th>Header 2</th>
  </tr>
  <tr>
    <td>Data 1</td>
    <td>Data 2</td>
  </tr>
</table>
```

`colspan` & `rowspan`

- `colspan="n"` — cell spans **n columns**
- `rowspan="n"` — cell spans **n rows**

Example:

```
<td colspan="2">Spans two columns</td>
```

Table Uses

- Used for **tabular data** (rows & columns)
- Avoid for page layout — use CSS instead

🔹 7. Forms

<form>

- Container for form elements.
- Attributes:
    - `action` → URL where form data is sent.
    - `method` → HTTP method (`GET` or `POST`).

Common `<input>` Types

- `text` — single-line text input.
- `email` — input for email addresses.
- `password` — input for passwords (hidden).
- `checkbox` — multiple choice options.
- `radio` — select one option from a group.
- `file` — file upload input.
- `submit` — submit button.

Other Form Elements

- `<textarea>` — multi-line text input.
- `<select>` — dropdown list.
- `<option>` — options inside `<select>`.
- `<label>` — labels for inputs; improves accessibility and usability.

Form Validation Attributes

- `required` — field must be filled.
- `min` / `max` — numeric or date limits.
- `pattern` — regex pattern for input format validation.

🔹 8. Media Elements

<audio>

- Embeds audio files on a webpage.
- Supports controls like play, pause, volume.
- Example:

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

<video>

- Embeds video files on a webpage.
- Supports controls like play, pause, volume, fullscreen.
- Example:

```
<video width="320" height="240" controls>
  <source src="video-file.mp4" type="video/mp4">
  Your browser does not support the video tag.
</video>
```

<source>

- Defines multiple media sources for `<audio>` and `<video>`.
- Browser picks the first supported format.

<iframe>

- Embeds external content like YouTube videos, Google Maps, other websites.
- Example:

```
<iframe width="560" height="315"
src="https://www.youtube.com/embed/VIDEO_ID"
title="YouTube video" frameborder="0" allowfullscreen></iframe>
```

🔹 9. Semantic HTML (HTML5)

Semantic tags give meaning to your HTML structure — improving accessibility, SEO, and readability.

Tag	Purpose
`<header>`	Defines introductory content or navigation at the top
`<footer>`	Defines footer content (copyright, contact info, etc.)
`<nav>`	Defines navigation links
`<main>`	Main content unique to the page (only one per page)
`<section>`	Thematic grouping of content (sections of a page)
`<article>`	Self-contained, independent content (blog post, news article)
`<aside>`	Sidebar or tangential content related to main content
`<figure>`	Container for media content (images, diagrams, code snippets)
`<figcaption>`	Caption or legend for `<figure>` content

🔹 10. Meta Tags

1. `<meta charset="UTF-8">`

- Sets the character encoding to UTF-8 (supports most characters worldwide).
- Always include to avoid character display issues.

---

2. `<meta name="viewport" content="width=device-width, initial-scale=1.0">`

- Makes your webpage **responsive** on different devices (especially mobiles).
- Sets the viewport width to device width and initial zoom level to 1.

---

3. SEO Meta Tags

- `<meta name="description" content="Your page description here">`
  Gives a summary shown in search engine results.

- `<meta name="keywords" content="HTML, CSS, web development">`
  Lists keywords related to your page content (less important now).

- `<meta name="author" content="Your Name">`
  Defines the page author.

🔹 11. HTML Entities

- HTML entities are **special codes** used to display reserved characters or symbols in HTML.

- They start with `&` and end with `;`.

Common HTML Entities:

Entity	Symbol	Description
` `	(space)	Non-breaking space (no line break)
`<`	<	Less than symbol
`>`	>	Greater than symbol
`&copy;`	©	Copyright symbol
`&`	&	Ampersand (&) symbol

Usage example:

```
<p>10 &lt; 20 means 10 is less than 20.</p>
<p>Use &copy; 2025 YourName.</p>
```

🔹 12. HTML Comments

- Comments are **notes in the code** that are not displayed on the webpage.

- Used to explain code or temporarily disable parts of code.
- Syntax:

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

- Everything between `<!--` and `-->` is ignored by the browser.

🔹 13. Input Attributes & HTML5 Form Controls

Common Attributes

- `placeholder` → hint text inside input
- `value` → sets default value
- `readonly` → visible but not editable
- `disabled` → cannot be used or submitted
- `checked` → pre-selected checkbox/radio
- `multiple` → allows multiple file uploads / selections

New HTML5 Input Types

- `date` → date picker
- `range` → slider control
- `color` → color picker
- `number` → numeric input with arrows

🔹 14. Responsive Design Support (with CSS)

Key Elements

- `<meta name="viewport" content="width=device-width, initial-scale=1.0">`
  → Makes page adjust to different screen sizes.

🔹 15. Accessibility (A11Y) Basics

1. `alt` attributes (for images)

- Provides text alternative for images.
- Helps screen readers describe visuals.
- Improves SEO & usability.

2. `<label for="">` (for inputs)

- Connects a label with a form input.
- Improves form accessibility & click usability.

3. Keyboard Navigable Structure

- Site should work via **Tab, Enter, Space, Arrow keys**.
- Ensure focus states (`:focus`) are visible.
- Avoid mouse-only interactions.

👉 Accessibility ensures your site is **inclusive for all users**

🔹 16. HTML Best Practices

1. Clean, Semantic Code

- Use meaningful tags (`<header>`, `<main>`, `<footer>` etc.).
- Improves SEO, accessibility & readability.

2. Proper Indentation & Comments

- Indent nested elements clearly.
- Use `<!-- comments -->` for explanation / reminders.
- Makes code easier to maintain.

3. Avoid Deprecated Tags

- Don't use old tags like `<font>`, `<center>`, `<marquee>`.
- Use **CSS** for styling & layout instead.

👉 Writing clean code = Professional & future-proof websites

🔹 17. HTML with CSS & JS Linking

1. Linking CSS

- **External:** `<link rel="stylesheet" href="style.css">` (best practice).
- **Internal:** `<style> ... </style>` inside `<head>`.
- **Inline:** `style="color:red;"` inside tag (avoid for maintainability).

2. Linking JavaScript

- **External:** `<script src="script.js"></script>` (preferred).
- **Internal:** `<script> ... </script>` inside HTML.
- **Placement:** Usually before `</body>` for faster loading.

👉 Use **external files** for clean, reusable & maintainable code.

🔹 18. Favicon & External Resources

1. Add Favicon (browser tab icon)

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

- Appears on browser tab.
- Use `.ico` or `.png` format.

2. Google Fonts

```
<link
href="https://fonts.googleapis.com/css2?family=Roboto&display=swap"
rel="stylesheet">
```

- Then use in CSS:

```
body { font-family: 'Roboto', sans-serif; }
```

3. External CSS/JS Libraries

- Example (Bootstrap CSS):

```
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap/dist/css/bootstrap.min.cs
s">
```

- Example (JS library like jQuery):

```
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
```

👉 Favicons + external resources = **branding + powerful design**

🔹 HTML Validation with W3C Validator

- **Validation** = checking your HTML code for errors.
- Tool: W3C Markup Validation Service
- You can **paste code, upload file, or enter website URL**.

🔹 Why use it?

- Finds mistakes (unclosed tags, wrong nesting, invalid attributes).
- Ensures **clean & standard HTML**.
- Improves **SEO, browser compatibility & accessibility**.

👉 In short: **Validation = Professional, error-free, reliable websites**