Basic HTML — Complete Notes ✦

~ a beginner's handwritten guide to building web pages, with diagrams ~

Who this is for: total beginners who have never written a line of HTML. No prior coding needed. By the end you'll be able to read HTML and build a real web page from scratch.

How to use these notes: read top to bottom, try each code example in a browser, and keep the cheat-sheet at the end handy. Every key idea has a diagram.

What's inside

1. What is HTML?
2. HTML, CSS & JavaScript — the trio
3. How the web & browsers work
4. Writing & running your first file
5. Tags vs Elements
6. Attributes — extra information
7. Four attributes you'll use everywhere
8. The HTML document structure
9. Inside the <head> section
10. Headings (<h1> to <h6>)
11. Paragraphs, line breaks & rules
12. Text formatting tags
13. HTML entities (special characters)
14. Comments & whitespace
15. Links (anchors)
16. File paths
17. Images
18. Audio, video & embedding
19. Lists
20. Tables
21. Forms & inputs
22. Div, Span & block vs inline
23. Semantic HTML
24. Adding CSS & JavaScript to a page
25. Colours in HTML
26. Best practices
27. Common beginner mistakes
28. Practice exercises
29. Mini project — a complete page
30. Cheat-sheet & glossary
31. What to learn next

1. What is HTML?

HTML stands for HyperText Markup Language. It is the standard language used to create the structure and content of every web page on the internet.

When you visit any website, your browser downloads an HTML file and turns it into the page you see. HTML describes what each piece of content is — "this is a heading", "this is a paragraph", "this is an image" — using little labels called tags.

Breaking down the name:

- HyperText — text that links to other pages. Clicking a hyperlink jumps you somewhere new.
- Markup — we "mark up" plain text with tags so the browser knows its meaning.
- Language — it has a fixed set of words and rules the browser understands.

In one line

HTML is the skeleton of a web page. It is not a programming language — it doesn't calculate or make decisions. It simply describes structure.

Good to know

HTML files end with `.html` (or `.htm`). The current version is HTML5 — that's what everyone uses today, and what these notes cover.

2. HTML, CSS & JavaScript — the trio

A modern web page is built from three languages that work together. HTML is always first — the other two build on top of it.

HTML	CSS	JavaScript
Structure	Style	Behaviour
headings, text, images (the skeleton)	colours, fonts, layout (the paint)	clicks, animation (the muscles)

Think of a house: HTML = bricks & rooms · CSS = paint & décor · JS = electricity & movement

You can build a complete web page with only HTML — it will just look plain. That's perfectly fine while you're learning. Focus on HTML first; add CSS and JavaScript later.

3. How the web & browsers work

Before writing HTML, it helps to know what happens when you open a website. Every page you view is really just a text file full of tags that a browser (Chrome, Firefox, Safari, Edge) downloads and draws.

Your browser asks a server for a page, gets back an HTML file, and paints it on screen

Key idea

HTML doesn't "run" on a server like some code — the browser on your own computer reads the tags and decides how to display them. That's why the same page can look slightly different in different browsers.

4. Writing & running your first file

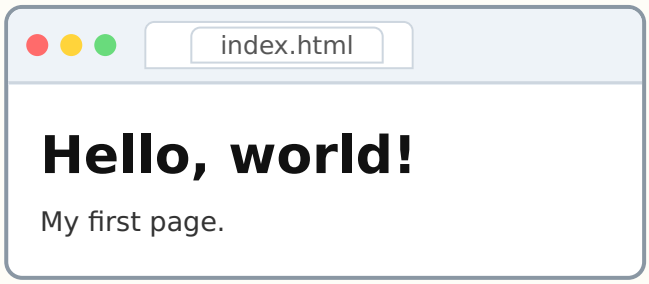
You don't need any special software to start — just a text editor and a browser. Here's the whole process:

1. Open a plain text editor. VS Code is free and popular, but even Notepad works.
2. Type some HTML (see the example on the right).
3. Save the file as `index.html` — the `.html` ending is what matters.
4. Find the file and double-click it — it opens in your browser, rendered.
5. Edit, save, and refresh the browser to see changes.

Tip

`index.html` is the traditional name for a site's home page. Servers look for it first.

```
<!-- save as index.html -->
<!DOCTYPE html>
<html>
  <body>
    <h1>Hello, world!</h1>
    <p>My first page.</p>
  </body>
</html>
```

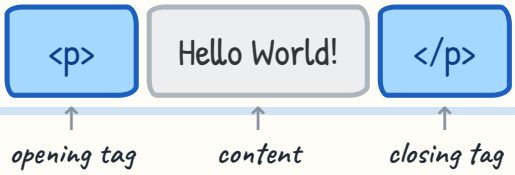


what the browser shows

5. Tags vs Elements

A tag is a keyword inside angle brackets, like `<p>`. Most tags come in pairs — an opening tag and a closing tag. The closing tag has a forward slash.

The opening tag + the content + the closing tag together make one element.



* Hello World! — this whole thing is one element (a paragraph)*

Remember

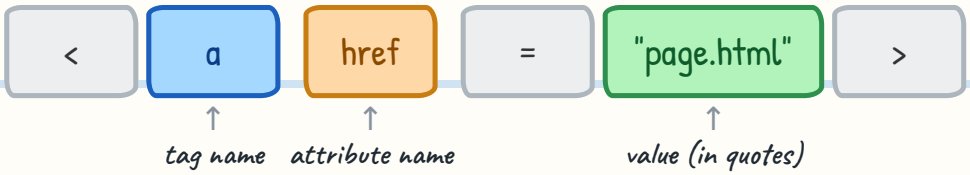
The closing tag is the same as the opening tag but with a / in front: `<h1>` closes with `</h1>`.

Empty (self-closing) tags

A few tags have no content and no closing tag, like `<br>` and `<img>`. They do their whole job in one tag.

6. Attributes — extra information

Sometimes a tag needs more detail. We add an attribute inside the opening tag. An attribute is a name paired with a "value" in quotes. For example, a link needs to know where it points:



the `href` attribute tells the <a> link where to go

Rules for attributes

- Always go inside the opening tag, never the closing tag.
- Values sit in "double quotes".
- A tag can have several attributes, separated by spaces: `<img src="x.jpg" alt="cat" width="200">`
- Names are not case-sensitive, but lowercase is standard.

7. Four attributes you'll use everywhere

A handful of attributes can be added to almost any tag. They're called **global attributes**. These four are the ones you'll meet first.

Attribute	What it does	Example
`id`	A unique name for one element (used by links & JavaScript)	`<h2 id="contact">`
`class`	A shared label for many elements (used by CSS to style groups)	`<p class="note">`
`style`	Quick inline CSS on one element	`<p style="color:red">`
`title`	Tooltip text shown on hover	`<abbr title="HyperText...">`

```
<p id="intro" class="lead"
  title="Hover me!">
 Welcome!
</p>
```

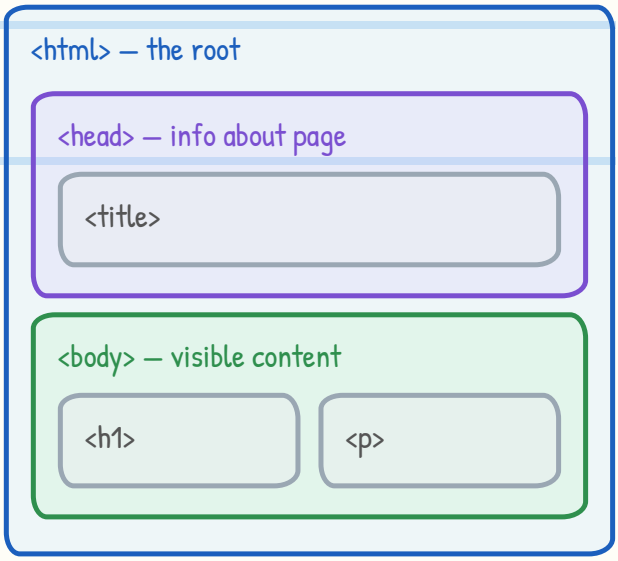
id vs class — the key difference

`id` must be unique (one per page). `class` can be reused on many elements. Rule of thumb: `class` for styling groups, `id` for a single jump-target or script hook.

8. The HTML document structure

Every HTML page follows the same basic skeleton. Memorise this — you'll type it at the start of every page you ever build.

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Page</title>
  </head>
  <body>
    <h1>Hello!</h1>
    <p>This is my page.</p>
  </body>
</html>
```



tags nest like boxes inside boxes

Part	What it does
`<!DOCTYPE html>`	Tells the browser "this is modern HTML5". Always the very first line.
`<html>`	The root element — everything else lives inside it.
`<head>`	Hidden info about the page (title, settings, links). Not shown in the page body.
`<body>`	Everything the visitor actually sees: text, images, links, buttons.

head vs body — the #1 beginner mix-up

Visible content (headings, paragraphs, images) goes in <body>. Behind-the-scenes info (page title, character set) goes in <head>. If your text won't appear, check it's inside `<body>`.

9. Inside the <head> section

The `<head>` holds information the browser and search engines need, but that visitors don't see directly on the page. The most common ones:

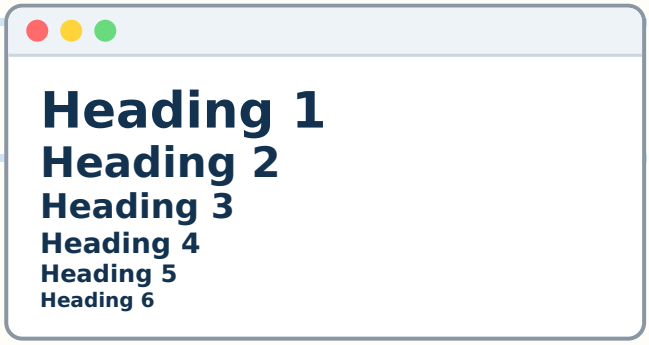
Tag	Purpose	Example
`<title>`	Text shown in the browser tab & in Google results	`<title>My Blog</title>`
`<meta charset>`	Sets the character set so symbols/ emojis show correctly	`<meta charset="UTF-8">`
`<meta name="viewport">`	Makes the page fit mobile screens	`<meta name="viewport" content="width=device-width">`
`<meta name="description">`	Short summary Google may show under your title	`<meta name="description" content="...">`
`<link>`	Attaches an external file, usually a CSS stylesheet	`<link rel="stylesheet" href="style.css">`

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

10. Headings (<h1> to <h6>)

Headings are titles and subtitles. There are six levels: `<h1>` is the biggest and most important, down to `<h6>` the smallest.

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```



bigger number = smaller heading

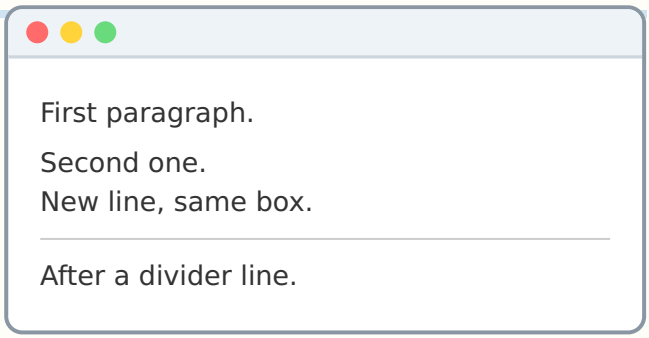
SEO tip

Use only one `<h1>` per page — it's the main title. Search engines use headings to understand your page, so use them in order (h1, then h2, then h3) and don't skip levels just to change size. Use CSS for sizing instead.

11. Paragraphs, line breaks & rules

Body text lives in paragraphs — the `<p>` tag. The browser adds space above and below each one automatically.

```
<p>First paragraph.</p>
<p>Second one.<br>New line, same box.</p>
<hr>
<p>After a divider line.</p>
```



Tag	Does	Closing tag?
`<p>`	A block of paragraph text	Yes — `</p>`
` `	A single line break (like pressing Enter)	No (empty tag)
`<hr>`	A horizontal divider line between sections	No (empty tag)
`<pre>`	Keeps your exact spacing & line breaks (good for code)	Yes — `</pre>`

Common mistake

Don't use lots of `<br>` tags to create space between paragraphs. Use separate `<p>` tags (or CSS) instead — it's cleaner and more correct.

12. Text formatting tags

These tags change how small pieces of text look or how important they are. They go inside a paragraph, around the words you want to affect.

Tag	Effect
`<strong>`	Bold + important
`<b>`	Bold (visual only)
`<em>`	Italic + emphasis
`<i>`	Italic (visual only)
`<u>`	Underline
`<mark>`	Highlighted text
`<small>`	Smaller text
`<del>`	Struck-through (deleted)
`<sub> <sup>`	Sub & super-script

```
<p>This is
<strong>bold</strong> and
<em>italic</em>.
Water is H<sub>2</sub>O.
5<sup>2</sup> = 25.
<mark>Highlighted!</mark></p>
```

This is **bold** and *italic*. Water is H_2O. 5^2 = 25. Highlighted!

vs <b> (and vs <i>)

They look the same, but `<strong>` and `<em>` also tell screen readers and search engines "this matters". Prefer them over the purely-visual `<b>` and `<i>`.

13. HTML entities (special characters)

Some characters are tricky because HTML already uses them (like `<` and `>`). To display them as text, we use an entity — a code that starts with `&` and ends with `;`.

You want	Type this	You want	Type this
< (less than)	`<`	© copyright	`&copy;`
> (greater than)	`>`	® registered	`&reg;`
& (ampersand)	`&`	™ trademark	`&trade;`
" (quote)	`&quot;`	→ arrow	`&rarr;`
a space (non-breaking)	` `	₹ / € / £	`&#8377; &euro; &pound;`

Why it matters

If you literally type `5 < 10` in HTML, the browser may think `< 10` is the start of a tag. Writing `5 < 10` safely shows "5 < 10".

14. Comments & whitespace

A comment is a note in your code that the browser ignores — it never shows on the page. Use it to explain things or temporarily "switch off" some HTML.

```
<!-- This is a comment -->
<p>Visible text</p>
<!-- <p>Hidden for now</p> -->
```

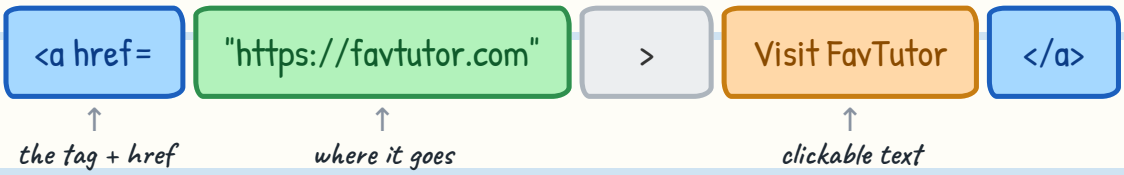
Whitespace collapses. HTML squashes multiple spaces, tabs and line breaks into a single space. So these look identical in the browser:

```
<p>Hello     World</p>
<p>Hello World</p>
```

Use this freedom to indent your code neatly — it won't affect the output.

15. Links (anchors)

Links are what make the web a "web". The anchor tag `<a>` uses the `href` attribute to say where to go. The text between the tags is what you click.



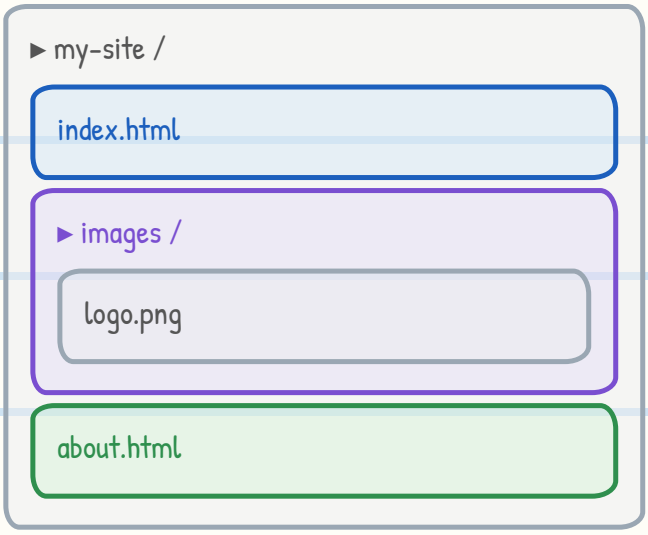
Types of links

Goal	Code
Another website (absolute URL)	`<a href="https://google.com">Google</a>`
Another page on your site (relative)	`<a href="about.html">About</a>`
Open in a new tab	`<a href="..." target="_blank">...</a>`
Send an email	`<a href="mailto:hi@site.com">Email</a>`
Call a phone number	`<a href="tel:+911234567890">Call</a>`
Jump to a section on the same page	`<a href="#contact">Contact</a>`

> **Tip**
>
> For jump links, give the target element an `id`: `<h2 id="contact">Contact</h2>`. The `#contact` in the link matches that id.

16. File paths

When you link to a page or an image on your own site, you tell the browser where the file lives relative to the current page. Picture your files in folders:



a typical little website folder

From `index.html`:

To reach	Path
about.html (same folder)	`about.html`
logo (inside images/)	`images/logo.png`
go up one folder	`../file.html`
another website	`https://...`

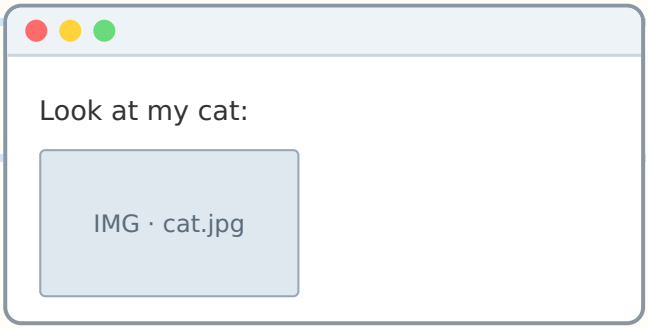
Two kinds of path

Relative = directions from the current file (`images/logo.png`). Absolute = a full web address (`https://...`).

17. Images

Images use the `<img>` tag. It's an empty tag — no closing tag. Two attributes matter most: src (which image file) and alt (a text description).

```
<img
  src="cat.jpg"
  alt="A sleepy cat"
  width="200">
```



the browser loads and shows the file

Attribute	Meaning
`src`	Path/URL of the image file (required)
`alt`	Text shown if image fails + read aloud by screen readers
`width` / `height`	Size in pixels
`title`	Tooltip on hover

Always add alt text

Good `alt` text helps blind users and boosts SEO — search engines read it to understand the image.

Grouping an image with a caption

Use `<figure>` and `<figcaption>` to tie an image to a caption:

```
<figure>
  <img src="chart.png" alt="Sales chart">
  <figcaption>Sales grew in 2026</figcaption>
</figure>
```

18. Audio, video & embedding

Beyond images, HTML5 can play sound and video, and embed other pages (like a YouTube video) — all without extra plugins.

Video & audio

```
<video src="clip.mp4" controls width="300">
</video>

<audio src="song.mp3" controls>
</audio>
```

the `controls` attribute adds play/pause

Embedding with <iframe>

An `<iframe>` ("inline frame") drops another web page inside yours — most commonly a YouTube video or a map.

```
<iframe src="https://www.youtube.com/embed/VIDEO_ID"
    width="560" height="315"></iframe>
```

> **Note**
>
> Video/audio need real media files, and browsers support different formats — `.mp4` for video and `.mp3` for audio are the safest choices.

19. Lists

HTML has three list types. Each item sits in its own tag.

Unordered list — bullets

```
<ul>
  <li>Apples</li>
  <li>Bananas</li>
</ul>
```

Ordered list — numbers

```
<ol>
  <li>Wake up</li>
  <li>Code</li>
</ol>
```

Fruits

- Apples
- Bananas

Morning

1. Wake up
2. Code

ul = bullets · ol = numbers

Nested lists

Put a `<ul>` or `<ol>` inside an `<li>` to make sub-items — great for menus and outlines.

Description list — term + definition

```
<dl>
  <dt>HTML</dt>
  <dd>Structure of a page</dd>
  <dt>CSS</dt>
  <dd>Styling of a page</dd>
</dl>
```

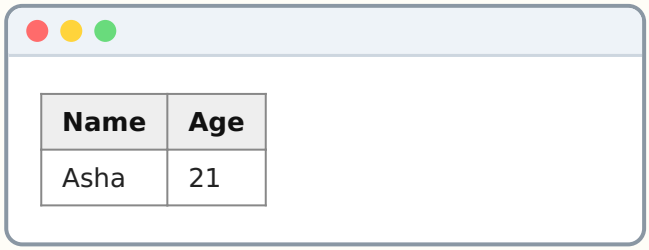
Tag	Means
`<dl>`	Description list wrapper
`<dt>`	The term being described
`<dd>`	The description of that term

20. Tables

Tables show data in rows and columns. They're built from a few nested tags. Learn these four:

Tag	Meaning
`<table>`	The whole table
`<tr>`	Table row
`<th>`	Header cell (bold, centred)
`<td>`	Data cell (normal)

```
<table>
  <tr>
    <th>Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Asha</td>
    <td>21</td>
  </tr>
</table>
```



rows go across, cells stack into columns

Merging cells

`colspan="2"` makes a cell span 2 columns; `rowspan="2"` spans 2 rows. Use them for headers that cover several columns.

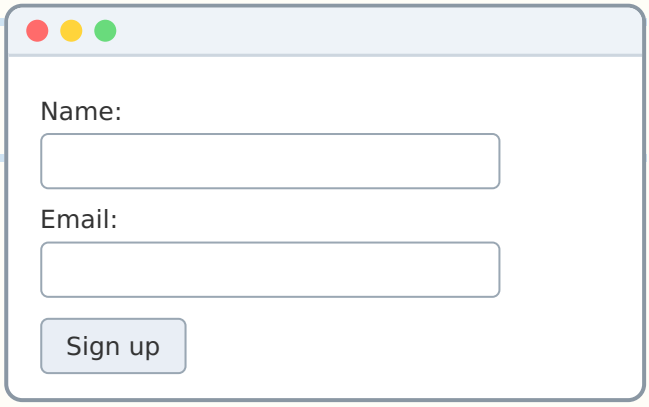
21. Forms & inputs

Forms collect information from visitors — a login, a search box, a sign-up. Everything sits inside a `<form>`. The star of forms is the flexible `<input>` tag.

```
<form>
  <label>Name:</label>
  <input type="text">

  <label>Email:</label>
  <input type="email">

  <input type="submit"
    value="Sign up">
</form>
```



a simple sign-up form

Useful <input> types (just change type="...")

type	Shows	type	Shows
`text`	A single-line text box	`checkbox`	A tick box
`email`	Text box that checks for @	`radio`	Pick-one circles
`password`	Hidden dots	`date`	A date picker
`number`	Numbers only	`submit`	The send button

Other form tags

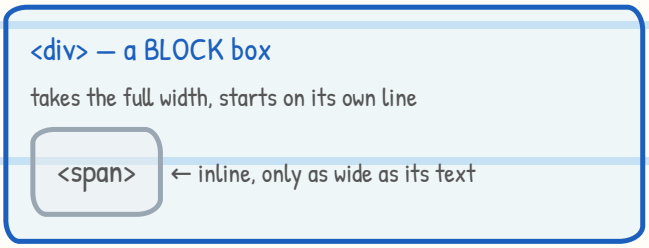
`<textarea>` = a big multi-line box · `<select>` + `<option>` = a dropdown · `<button>` = a clickable button · `<label>` = the caption for a field (helps accessibility).

Good habit

Always pair an `<input>` with a `<label>`. Add a `placeholder="..."` for hint text inside the box.

22. Div, Span & block vs inline

`<div>` and `<span>` are "container" tags with no look of their own — they exist to group content so you can style or position it later with CSS.



Two ways elements sit on a page:

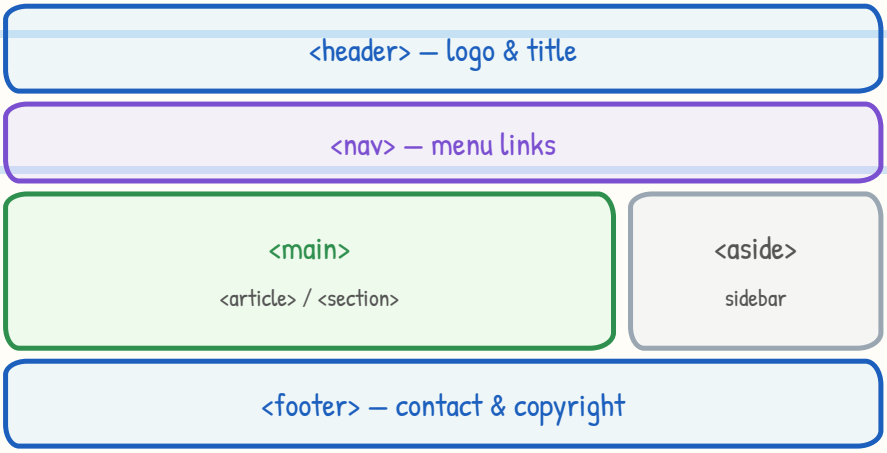
Type	Behaviour	Examples
Block	New line, full width	`div` , `p` , `h1` , `ul`
Inline	Stays in the line, hugs its content	`span` , `a` , `strong` , `img`

```
<div class="card">
  <p>Price: <span class="big">₹499</span></p>
</div>
```

The `class` attribute is just a label you'll later target with CSS. You'll use `<div>` and `class` constantly once you learn CSS.

23. Semantic HTML

Instead of using `<div>` for everything, HTML5 gives us meaningful tags that describe the role of each region. These are called semantic tags. They make your code readable and help SEO & screen readers.



a typical page layout built from semantic tags

Tag	Region it marks	Tag	Region it marks
`<header>`	Top: logo, title	`<article>`	A self-contained post
`<nav>`	Navigation menu	`<aside>`	Side content
`<main>`	The main content	`<footer>`	Bottom info
`<section>`	A themed group of content	`<figure>`	Image + caption

24. Adding CSS & JavaScript to a page

HTML gives structure; CSS adds style and JavaScript adds behaviour. Here's how you attach them — you'll use the external way most of the time.

Three ways to add CSS

Way	How	When
Inline	`<p style="color:red">`	A one-off quick tweak
Internal	`<style>` block inside `<head>`	Styles for a single page
External ✓	`<link rel="stylesheet" href="style.css">`	Best — one file styles the whole site

```
<!-- internal CSS -->
<head>
  <style>
    h1 { color: blue; }
  </style>
</head>
```

```
<!-- add JavaScript -->
<script>
  alert("Hi!");
</script>

<!-- or external -->
<script src="app.js"></script>
```

Where do they go?

`<link>` to CSS goes in the `<head>`. `<script>` tags are usually placed just before the closing `</body>` so the page loads first, then the script runs.

25. Colours in HTML

Colours are usually set with CSS, but it helps to know the three ways to name a colour. You'll meet all of them constantly.

Way	Looks like	Example
Name	140 preset names	`tomato`, `skyblue`
Hex code	# + 6 digits	`#1c5fbd`
RGB	red, green, blue 0–255	`rgb(28,95,189)`

Hex is the most common. Each pair is Red-Green-Blue, from `00` (none) to `ff` (full). `#ffffff` = white, `#000000` = black.

a few hex colours & their swatches

```
<!-- colours are applied with the style attribute (a taste of CSS) -->
<h1 style="color: #1c5fbd;">Blue heading</h1>
```

26. Best practices

Habits of clean HTML

- Write tag names in lowercase and always close your tags.
- Indent nested tags so the structure is easy to read.
- Use semantic tags (`header` , `nav` , `main` ...) instead of `div` for everything.
- Give every image a meaningful `alt` .
- Use one `<h1>` per page; keep headings in order.
- Add the `<meta charset="UTF-8">` and viewport tags to every page.
- Keep content (HTML) separate from styling (CSS) — don't over-use the `style` attribute.

27. Common beginner mistakes

Watch out for these

- Forgetting the closing tag, or the slash in it (`<p>` instead of `</p>`).
- Crossing tags — remember: last opened, first closed. `<b><i>hi</b></i>` is wrong.
- Missing quotes around attribute values.
- Putting attributes in the closing tag.
- Putting visible text in `<head>` instead of `<body>`.
- Using `<br>` many times instead of proper `<p>` paragraphs.
- Skipping `alt` on images.

28. Practice exercises

Reading isn't enough — the only way to learn HTML is to type it. Try these small challenges in a real `.html` file. Start each with the document skeleton.

Warm-up

1. Make a page with your name as an `<h1>` and a short intro paragraph.
2. Add three of your hobbies as an unordered list.
3. Add a link to your favourite website that opens in a new tab.
4. Insert an image (any `.jpg` from your computer) with proper `alt` text.

Level up

1. Build a 3-column table of your weekly timetable using `<th>` and `<td>`.
2. Create a contact form with name, email and a submit button.
3. Lay out a page using `<header>`, `<nav>`, `<main>` and `<footer>`.
4. Add a comment explaining what each section does.

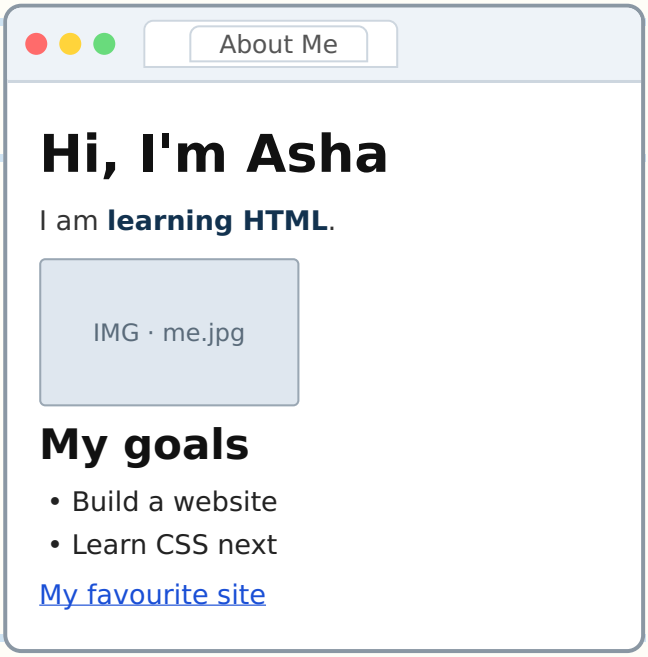
How to check your work

Open the file in your browser after each change and refresh. If something looks wrong, check for a missing closing tag or a missing quote — those cause most beginner bugs.

29. Mini project — a complete page

Here's everything so far, combined into one real page. Type it out, save as `index.html`, and open it in your browser.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>About Me</title>
</head>
<body>
  <h1>Hi, I'm Asha</h1>
  <p>I am <strong>learning HTML</strong>.</p>
  <img src="me.jpg" alt="My photo">
  <h2>My goals</h2>
  <ul>
    <li>Build a website</li>
    <li>Learn CSS next</li>
  </ul>
  <a href="https://favtutor.com">My favourite site</
a>
</body>
</html>
```



your finished page

You did it!

This one page uses the doctype, head, title, meta, headings, a paragraph, bold text, an image, a list, and a link — the core of HTML.

30. Cheat-sheet & glossary

Tag quick reference

Tag	Use	Tag	Use
`<h1>-<h6>`	Headings	`<img>`	Image
`<p>`	Paragraph	`<ul>/<ol>/<li>`	Lists
` `	Line break	`<table><tr><td>`	Tables
`<hr>`	Divider line	`<form><input>`	Forms
`<strong>/<em>`	Bold / italic	`<div>/<span>`	Containers
`<a>`	Link	`<header><main><footer>`	Semantic layout

Glossary

Word	Means
Tag	A keyword in angle brackets, e.g. `<p>`
Element	Opening tag + content + closing tag
Attribute	Extra info inside an opening tag (name="value")
Empty tag	A tag with no closing tag, e.g. ` `, `<img>`
Nesting	Putting tags inside other tags
Semantic	Tags that describe meaning (`<nav>`, `<footer>`)

31. What to learn next

Your roadmap

HTML (structure ✓ you're here) → CSS (colours, fonts, layout) → JavaScript (interactivity). Then explore CSS Flexbox & Grid, responsive design, and build small projects — a profile page, a to-do list, a landing page.

You can now read and write real HTML. Keep building — practice beats theory!