SkillNewton

HTML Cheat Sheet

Every everyday HTML tag and attribute on one page: structure, text, media, tables, forms, semantic layout, meta tags and accessibility.

> This HTML cheat sheet lists the tags and attributes you use on almost every page, with a one-line explanation for each. It is written for beginners building their first sites and for developers who want a quick reminder of form input types, Open Graph tags or ARIA attributes. Start with the document skeleton, then jump to the section you need. Every snippet is valid HTML you can paste into a file and open in a browser.

How to use it

1. **Copy the skeleton.** Start every new page from the document skeleton in the first section so the doctype, charset, viewport and language are always right.
2. **Pick semantic elements first.** Before reaching for a div, check the semantic layout and text sections for an element that already describes the content.
3. **Fill in the head.** Add a unique title, meta description and canonical link to each page, plus Open Graph tags if it will be shared on social media.
4. **Check accessibility.** Run through the accessibility section: alt text, labels on every input and visible focus for keyboard users.
5. **Validate.** Paste your page into the W3C Markup Validation Service to catch unclosed tags and invalid attributes.

Document skeleton

`<!DOCTYPE html>`	Switches the browser to standards mode. Always the very first line.
`<html lang="en-IN">`	Root element. `lang` tells screen readers and search engines the page language.
`<meta charset="utf-8">`	Character encoding. Put it first inside `<head>` so Hindi, emoji and symbols display correctly.
`<meta name="viewport" content="width=device-width, initial-scale=1">`	Makes the page use the device width on phones instead of a zoomed-out desktop view.
`<title>Page title | Brand</title>`	Shown in the browser tab and often as the search result headline. Unique per page.
`<link rel="stylesheet" href="/css/style.css">`	Loads an external stylesheet.
`<script src="/js/app.js" defer></script>`	Loads JavaScript without blocking parsing; deferred scripts run in order after the HTML is parsed.
`<!-- comment -->`	A comment. Visible in page source, so never put secrets here.

```
<!DOCTYPE html>
<html lang="en-IN">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
  <title>Home | Example Store</title>
  <meta name="description" content="Handmade notebooks shipped across India.">
  <link rel="stylesheet" href="/css/style.css">
  <script src="/js/app.js" defer></script>
</head>
<body>
  <h1>Example Store</h1>
</body>
</html>
```

Text

`<h1>...</h1>` to `<h6>...</h6>`	Headings. Use one `h1` for the page topic and do not skip levels for styling.
`<p>Paragraph</p>`	A paragraph of text.
`<strong>important</strong>`	Strong importance (bold by default).
`<em>stress</em>`	Stress emphasis (italic by default).
` `	Line break inside text, such as an address. Not for spacing.
`<hr>`	Thematic break between topics.
`<blockquote cite="https://example.com/source">...</blockquote>`	A quotation from another source.
`<code>let x = 1;</code>`	Inline code.
`<pre><code>...</code></pre>`	Pre-formatted block that keeps spaces and line breaks; used for code samples.
`<abbr title="Search Engine Optimisation">SEO</abbr>`	Abbreviation with its expansion.
`<time datetime="2026-11-10">10 Nov 2026</time>`	A machine-readable date or time.
`<mark>highlight</mark>`	Highlighted text, for example search matches.

Links and media

`<a href="/courses/">Courses</a>`	Link to another page. Write descriptive link text, not "click here".
`<a href="https://example.com" target="_blank" rel="noopener">Site</a>`	Opens in a new tab. `rel="noopener"` stops the new page controlling yours.
`<a href="#pricing">Jump to pricing</a>`	Jumps to the element with `id="pricing"` on the same page.
`<a href="mailto:hello@example.com">Email us</a>`	Opens the visitor's email app.
`<a href="tel:+911234567890">Call</a>`	Starts a phone call on mobiles. Use the full international format.
`<a href="/files/guide.pdf" download>Download</a>`	Asks the browser to download the file instead of opening it.
`<img src="cat.jpg" alt="Grey cat on a sofa" width="800" height="600">`	Image. Width and height reserve space and prevent layout shift.

`<img src="photo.jpg" alt="..." loading="lazy">`	Lazy-loads an image below the fold. Do not lazy-load the main hero image.
`<img srcset="s.jpg 480w, l.jpg 1200w" sizes="(max-width: 600px) 480px, 1200px" src="l.jpg" alt="...">`	Lets the browser pick the right image size for the screen.
`<picture><source srcset="a.avif" type="image/avif"><img src="a.jpg" alt="..."></picture>`	Serves a modern format with a fallback.
`<figure><img ...><figcaption>Caption</figcaption></figure>`	Image or diagram with a visible caption.
`<video src="demo.mp4" controls muted playsinline></video>`	Embedded video with player controls.

Lists and tables

`<ul><li>Item</li></ul>`	Unordered (bulleted) list.
`<ol><li>Step</li></ol>`	Ordered (numbered) list. Add `start="5"` or `reversed` if needed.
`<dl><dt>Term</dt><dd>Definition</dd></dl>`	Description list for terms and definitions or label and value pairs.
`<table>...</table>`	Table for tabular data only, never for page layout.
`<caption>Course fees</caption>`	Table title. First child of `<table>`.
`<thead>, <tbody>, <tfoot>`	Group header, body and footer rows.
`<tr><th scope="col">Name</th></tr>`	Header cell. `scope` tells assistive tech whether it heads a column or a row.
`<td colspan="2">...</td>`	Data cell spanning two columns. `rowspan` spans rows.

```
<table>
  <caption>Batch timings</caption>
  <thead>
    <tr><th scope="col">Batch</th><th scope="col">Time</th></tr>
  </thead>
  <tbody>
    <tr><th scope="row">Morning</th><td>9:00 to 11:00</td></tr>
    <tr><th scope="row">Evening</th><td>18:00 to 20:00</td></tr>
  </tbody>
</table>
```

Forms and input types

`<form action="/enquiry" method="post">`	Sends fields to `/enquiry`. Use `post` for anything that changes data.
`<form ... enctype="multipart/form-data">`	Required when the form uploads files.
`<label for="email">Email</label> <input id="email" name="email">`	Every input needs a label; `for` matches the input `id`.
`<input type="email" autocomplete="email" required>`	Email field with mobile keyboard, browser autofill and built-in validation.

`<input type="tel" | "url" | "number" | "date" | "search">`	Specialised inputs with suitable keyboards and pickers.
`<input type="password" autocomplete="current-password">`	Masked password. Use `new-password` on sign-up forms.
`<input type="checkbox" name="terms" value="yes">`	Tick box; only sent when ticked.
`<input type="radio" name="batch" value="morning">`	Pick one of several options sharing the same `name`.
`<input type="file" name="cv" accept=".pdf">`	File upload, limited to PDF in the picker.
`<input minlength="3" maxlength="50" pattern="[A-Za-z ]+">`	Client-side validation rules. Always validate again on the server.
`<select name="city"><option value="pune">Pune</option></select>`	Drop-down list.
`<textarea name="message" rows="5"></textarea>`	Multi-line text.

```
<form action="/enquiry" method="post">
  <fieldset>
    <legend>Course enquiry</legend>
    <label for="name">Full name</label>
    <input id="name" name="name" autocomplete="name" required>

    <label for="phone">Mobile number</label>
    <input id="phone" name="phone" type="tel" autocomplete="tel">

    <button type="submit">Send enquiry</button>
  </fieldset>
</form>
```

Semantic layout elements

`<header>`	Introductory content: logo, site title or the heading of an article.
`<nav aria-label="Main">`	A block of major navigation links.
`<main>`	The main content of the page. Only one visible per page.
`<article>`	Self-contained content that makes sense on its own: a blog post, product card or comment.
`<section>`	A thematic group of content, normally with its own heading.
`<aside>`	Related but secondary content, such as a sidebar or callout.
`<footer>`	Footer for the page or for its nearest article or section.
`<details><summary>Question</summary>Answer</details>`	Native expand and collapse, useful for FAQs. No JavaScript needed.
`<dialog id="d">...</dialog>`	A modal or non-modal dialog; open it with `showModal()` in JavaScript.

`<div> / <span>`	Generic block and inline containers with no meaning. Use when nothing else fits.

Head, SEO and social meta tags

`<meta name="description" content="...">`	Page summary that search engines may show as the snippet.
`<link rel="canonical" href="https://example.com/page/">`	Tells search engines the preferred URL when duplicates exist.
`<meta name="robots" content="noindex, follow">`	Keeps a page out of search results while still following its links.
`<link rel="alternate" hreflang="hi-IN" href="https://example.com/hi/">`	Points to a language or regional version of the page.
`<meta property="og:title" content="...">`	Open Graph title used when the page is shared on social platforms.
`<meta property="og:description" content="...">`	Open Graph description for share previews.
`<meta property="og:image" content="https://example.com/share.jpg">`	Share preview image. Use an absolute URL.
`<meta property="og:url" content="https://example.com/page/">`	The canonical URL of the shared object.
`<meta property="og:type" content="website">`	Object type, for example `website` or `article`.
`<meta name="twitter:card" content="summary_large_image">`	Asks X (Twitter) to show a large image preview.
`<link rel="icon" href="/favicon.ico" sizes="any">`	Site icon for tabs and search results.
`<link rel="preload" href="/fonts/inter.woff2" as="font" type="font/woff2" crossorigin>`	Fetches a critical file early.

Accessibility attributes

`alt="Student using a laptop"`	Describes an informative image. Use `alt=""` for purely decorative images.
`<a class="skip" href="#main">Skip to content</a>`	Skip link so keyboard users can bypass the navigation.
`aria-label="Close"`	Gives an accessible name to a control with no visible text, such as an icon button.
`aria-labelledby="heading-id"`	Names an element using the text of another element.
`aria-describedby="hint-id"`	Links extra help or error text to a form field.
`aria-expanded="false"`	States whether a menu or accordion controlled by this button is open.
`aria-current="page"`	Marks the current page link in navigation.
`aria-hidden="true"`	Hides decorative content such as icons from screen readers.

`aria-live="polite"`	Announces content that changes later, such as a form status message.
`tabindex="0" / tabindex="-1"`	`0` adds an element to the keyboard order; `-1` makes it focusable only by script.
`<button type="button">`	Use a real button for actions instead of a clickable div; it gets keyboard and screen reader support free.

Online version, with the matching free guide: skillnewton.com/resources/cheat-sheets/html-cheat-sheet/
Learn this hands-on: PHP Web Development: Foundation & Practical Implementation – skillnewton.com/courses/php-foundation-course/