

HTML Fundamentals

A Complete Study Guide for Students

From Basic Tags to Forms, Tables & Semantic HTML5

What you'll learn:

- Document structure & core syntax
- Text, links, images, lists
- Tables and forms
- Semantic HTML5 layout elements
- Best practices & accessibility basics
- Practice exercises after every chapter

Lecture Notes · Study Edition

Table of Contents

- 1. Introduction to HTML & How the Web Works
- 2. Document Structure & Basic Syntax
- 3. Text Elements & Formatting
- 4. Links and Images
- 5. Lists
- 6. Tables
- 7. Forms
- 8. Semantic HTML5 & Page Layout
- 9. Attributes, Comments & Best Practices
- 10. Quick Reference Cheat Sheet

1. Introduction to HTML & How the Web Works

1.1 What is HTML?

HTML (HyperText Markup Language) is the standard markup language used to create web pages. It is not a programming language — it does not contain logic, loops, or variables. Instead, it describes the *structure* and *content* of a page using elements called **tags**.

Think of HTML as the **skeleton** of a webpage. CSS is the skin and clothing (styling), and JavaScript is the muscles (behavior/interactivity).

1.2 How a Webpage is Built: The Three Layers

Layer	Purpose	Example
HTML	Structure and content	Headings, paragraphs, images
CSS	Presentation and styling	Colors, fonts, layout
JavaScript	Behavior and interactivity	Button clicks, animations

1.3 How Browsers Render HTML

When you type a URL into a browser, the following happens:

1. The browser sends a request to a web server.
2. The server responds with an HTML file (and linked CSS/JS files).
3. The browser parses the HTML and builds the **DOM** (Document Object Model) — a tree representation of the page.
4. CSS is applied, and the page is rendered visually on screen.

1.4 Tools You Need

- A text editor (VS Code, Sublime Text, or even Notepad)
- A web browser (Chrome, Firefox, Edge)
- No installation or compiler required — HTML files run directly in the browser!

➤ **Exercise 1.1**

Open a text editor, create a file called `index.html` , and open it using your web browser (double-click it). What do you see if the file is empty?

Key Takeaways

- HTML defines structure, not appearance or behavior.
- Browsers convert HTML into the DOM, then render it visually.
- You only need a text editor and a browser to start.

2. Document Structure & Basic Syntax

2.1 Anatomy of an HTML Tag

Most HTML elements consist of an **opening tag**, **content**, and a **closing tag**:

```
<p>This is a paragraph.</p>
```

Some elements are **self-closing** (void elements) and have no closing tag or content, such as `<br>` , `<img>` , and `<input>` .

2.2 The Minimum HTML Document

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>My First Page</title>
</head>
<body>
  <h1>Hello, World!</h1>
</body>
</html>
```

Breaking it down

Element	Purpose
<!DOCTYPE html>	Tells the browser this is an HTML5 document
<html>	The root element wrapping the entire page
<head>	Metadata — not visible on the page itself
<meta charset="UTF-8">	Sets the character encoding (supports all languages/symbols)
<title>	Text shown in the browser tab
<body>	All visible content goes here

Common mistake: Forgetting to close tags, or nesting them incorrectly (e.g. `<b><i>text</b></i>` instead of `<b><i>text</i></b>`). Tags must close in the reverse order they were opened — like stacking boxes.

2.3 Nesting Elements

Elements can contain other elements. This is called **nesting**, and it forms the tree-like structure of the DOM.

```
<div>
  <h2>Title</h2>
  <p>Some <strong>bold</strong> text.</p>
</div>
```

📌 Exercise 2.1

Write the minimum valid HTML document with the title "My Portfolio" and an `<h1>` that says "Welcome to my portfolio."

Key Takeaways

- Every HTML page starts with `<!DOCTYPE html>` .
- `<head>` holds metadata; `<body>` holds visible content.
- Tags must be properly nested and closed in reverse order.

3. Text Elements & Formatting

3.1 Headings

HTML provides six levels of headings, `<h1>` through `<h6>` , in decreasing order of importance.

```
<h1>Main Title</h1>
<h2>Section Heading</h2>
<h3>Subsection Heading</h3>
```

Use only **one** `<h1>` per page — it represents the page's main topic and matters for accessibility and SEO.

3.2 Paragraphs & Line Breaks

```
<p>This is a paragraph of text.</p>
<p>Line one<br>Line two (same paragraph)</p>
<hr> <!-- horizontal rule / divider -->
```

3.3 Text Formatting Tags

Tag	Effect	Semantic Meaning
<code></code>	Bold	Strong importance
<code></code>	Italic	Emphasis
<code></code>	Bold	Visual only, no semantic weight
<code><i></code>	Italic	Visual only, no semantic weight
<code><mark></code>	Highlighted	Relevant/marked text
<code><small></code>	Smaller text	Fine print
<code><sub></code> / <code><sup></code>	Subscript/superscript	H ₂ O, x ²
<code></code> / <code><ins></code>	Strikethrough / underline	Deleted/inserted content

Prefer `<strong>` / `<em>` **over** `<b>` / `<i>` when possible — screen readers announce semantic emphasis differently, improving accessibility.

3.4 Quotations & Preformatted Text

```
<blockquote>
  A great quote goes here.
</blockquote>

<p>She said <q>hello</q> to me.</p>

<pre>
Text    with    exact
spacing preserved
</pre>
```

⇒ **Exercise 3.1**

Create a short bio paragraph using at least: one `<h2>` , one `<strong>` , one `<em>` , and a `<blockquote>` with your favorite quote.

Key Takeaways

- Headings (`h1` – `h6`) create document hierarchy.
- Semantic tags (`strong` , `em`) carry meaning; presentational tags (`b` , `i`) do not.
- `<pre>` preserves whitespace exactly as written.

4. Links and Images

4.1 The Anchor Tag

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

<!-- open in a new tab -->
<a href="https://example.com" target="_blank" rel="noopener">Visit</a>

<!-- link to another page in same site -->
<a href="about.html">About Us</a>

<!-- link to a section on the same page -->
<a href="#contact">Jump to Contact</a>

<!-- email / phone links -->
<a href="mailto:hi@example.com">Email us</a>
<a href="tel:+123456789">Call us</a>
```

4.2 The Image Tag

```

```

Attribute	Purpose
src	Path or URL to the image file
alt	Alternative text — shown if image fails, read by screen readers
width / height	Sets the display dimensions in pixels

Always include alt text. It is essential for accessibility and is required for valid, semantic HTML.

4.3 Relative vs. Absolute Paths

Type	Example	Meaning
Absolute	https://site.com/img.png	Full URL, works anywhere
Relative	images/img.png	Path relative to the current file's location
Root-relative	/images/img.png	Path from the site's root folder

4.4 Figures with Captions

```
<figure>
  
  <figcaption>Figure 1: Quarterly sales</figcaption>
</figure>
```

Exercise 4.1

Create a small "favorite websites" page with three links (opening in new tabs) and one image with descriptive alt text.

Key Takeaways

- creates hyperlinks; target="_blank" opens a new tab.
- is a void element — always include alt .
- <figure> / <figcaption> semantically group an image with its caption.

5. Lists

5.1 Unordered Lists

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

5.2 Ordered Lists

```
<ol>
  <li>Preheat the oven</li>
  <li>Mix ingredients</li>
  <li>Bake for 20 minutes</li>
</ol>
```

5.3 Nested Lists

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

5.4 Description Lists

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

Used for term/definition pairs — like a glossary.

⇒ Exercise 5.1

Build a recipe page: an unordered list of ingredients and an ordered list of numbered steps.

Key Takeaways

- `<ul>` = unordered (bullets), `<ol>` = ordered (numbers).
- Lists can be nested inside a parent `<li>`.
- `<dl>` / `<dt>` / `<dd>` handle term-definition pairs.

6. Tables

6.1 Basic Table Structure

```
<table>
  <thead>
    <tr>
      <th>Name</th>
      <th>Grade</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Alice</td>
      <td>A</td>
    </tr>
    <tr>
      <td>Bob</td>
      <td>B+</td>
    </tr>
  </tbody>
</table>
```

Tag	Purpose
<table>	Wraps the entire table
<thead>	Groups header row(s)
<tbody>	Groups the main data rows
<tfoot>	Groups footer row(s), e.g. totals
<tr>	Table row
<th>	Header cell (bold, centered by default)
<td>	Standard data cell

6.2 Merging Cells

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

Tables should be used for **tabular data** only (schedules, statistics, comparisons) — never for page layout. Use CSS for layout instead.

⇒ **Exercise 6.1**

Create a weekly class schedule table with days as columns and time slots as rows.

Key Takeaways

- Tables are built from `table > (thead/tbody/tfoot) > tr > th/td`.
- `colspan` and `rowspan` merge cells.
- Never use tables for visual page layout.

7. Forms

7.1 The Form Element

```
<form action="/submit" method="POST">
  ...
</form>
```

Attribute	Purpose
action	URL where form data is sent
method	GET (data in URL) or POST (data in request body)

7.2 Text Inputs & Labels

```
<label for="name">Full Name:</label>
<input type="text" id="name" name="name" placeholder="Jane Doe" required>
```

Always pair `<label for="id">` with a matching input `id` . This links them for screen readers and lets clicking the label focus the input.

7.3 Common Input Types

Type	Example
text	Single-line text
email	Validates email format
password	Masks characters
number	Numeric input with steppers
date	Date picker
checkbox	Multiple selectable options
radio	Single choice among options (shared <code>name</code>)
file	File upload
submit	Submits the form

7.4 Checkboxes & Radio Buttons

```
<!-- checkboxes: multiple choices allowed -->
<input type="checkbox" id="pizza" name="topping" value="pizza">
<label for="pizza">Pizza</label>

<!-- radio buttons: only one choice, same "name" -->
<input type="radio" id="m" name="gender" value="male">
<label for="m">Male</label>
<input type="radio" id="f" name="gender" value="female">
<label for="f">Female</label>
```

7.5 Dropdowns & Text Areas

```
<select name="country">
  <option value="eg">Egypt</option>
  <option value="us">USA</option>
</select>

<textarea name="message" rows="4" cols="30"></textarea>
```

7.6 Buttons

```
<button type="submit">Send</button>
<button type="reset">Clear</button>
<button type="button">Just a click</button>
```

7.7 Grouping Fields with Fieldset

```
<fieldset>
  <legend>Contact Info</legend>
  ... inputs ...
</fieldset>
```

Exercise 7.1

Build a "Contact Us" form with: name, email, a dropdown for subject, a message textarea, and a submit button — all properly labeled.

Key Takeaways

- Forms collect user input and send it via `action / method`.
- Every input needs a matching `<label>` for accessibility.
- Radio buttons sharing a `name` become mutually exclusive.

8. Semantic HTML5 & Page Layout

8.1 Why Semantic HTML?

Semantic elements clearly describe their meaning to both the browser and the developer, improving **accessibility**, **SEO**, and code readability — compared to using generic `<div>` tags everywhere.

8.2 Key Semantic Elements

Element	Represents
<code><header></code>	Introductory content, often a logo/nav
<code><nav></code>	Navigation links
<code><main></code>	The primary content, unique to that page (only one per page)
<code><section></code>	A thematic grouping of content
<code><article></code>	Self-contained content (blog post, news story)
<code><aside></code>	Tangential content, like a sidebar
<code><footer></code>	Footer info: copyright, links, contact

8.3 A Semantic Page Skeleton

```
<body>
  <header>
    <h1>My Blog</h1>
    <nav>
      <a href="#">Home</a>
      <a href="#">About</a>
    </nav>
  </header>

  <main>
    <article>
      <h2>Post Title</h2>
      <p>Post content...</p>
    </article>

    <aside>
      <h3>Related Posts</h3>
    </aside>
  </main>

  <footer>
    <p>&copy; 2026 My Blog</p>
  </footer>
</body>
```

8.4 div and span: The Generic Containers

`<div>` (block-level) and `<span>` (inline) carry no semantic meaning — use them only when no semantic element fits, typically for styling/scripting hooks.

Rule of thumb: Ask "does this content have a clear role (navigation, article, sidebar)?" If yes, use a semantic tag. If it's purely a styling wrapper, `<div>` is fine.

⇒ **Exercise 8.1**

Rebuild the semantic skeleton above with real content: a header with a nav of 3 links, a main section with one article, an aside listing 2 "related" items, and a footer with a copyright line.

Key Takeaways

- Semantic tags describe meaning, improving SEO and accessibility.

- Use one `<main>` , and `<header>` / `<footer>` per page (or per article).
- `<div>` / `<span>` are meaning-free fallbacks for styling.

9. Attributes, Comments & Best Practices

9.1 Global Attributes

Attribute	Purpose
id	Unique identifier for one element
class	Reusable name for styling/scripting groups of elements
style	Inline CSS (avoid overusing — prefer external CSS)
title	Tooltip text shown on hover
data-*	Custom data attributes for JavaScript use

9.2 Comments

```
<!-- This is a comment; it is not rendered on the page -->
```

9.3 Best Practices Checklist

- Always declare `<!DOCTYPE html>` and set `lang` on `<html>`.
- Use lowercase tag and attribute names.
- Always quote attribute values: `class="box"`, not `class=box`.
- Close all elements, and close them in the correct (reverse) order.
- Use one `<h1>` per page; keep a logical heading hierarchy (don't skip from h2 to h4).
- Always add meaningful `alt` text to images.
- Prefer semantic tags over `<div>` soup.
- Separate content (HTML) from presentation (CSS) — avoid excessive inline styles.
- Validate your HTML using the [W3C Validator](#).

Key Takeaways

- `id` is unique; `class` can repeat across elements.
- Comments help document code but are invisible on the page.
- Good HTML is valid, semantic, and accessible.

10. Quick Reference Cheat Sheet

10.1 Structure

Tag	Use
<!DOCTYPE html>	Document type declaration
<html>	Root element
<head>	Metadata container
<body>	Visible content container

10.2 Text & Content

Tag	Use
<h1>--<h6>	Headings
<p>	Paragraph
 / 	Semantic bold / italic
/ /	Lists
	Link
	Image

10.3 Tables

Tag	Use
<table><tr><th><td>	Rows, header cells, data cells
colspan / rowspan	Merge cells

10.4 Forms

Tag	Use
<form>	Wraps inputs, defines submission
<input type="">	text / email / password / checkbox / radio / date...
<select><option>	Dropdown menu
<textarea>	Multi-line text
<button>	Clickable action
<label for="">	Accessible field label

10.5 Semantic Layout

Tag	Use
<header> / <footer>	Top / bottom page or section content
<nav>	Navigation links
<main>	Primary unique page content
<section> / <article>	Thematic grouping / self-contained content
<aside>	Sidebar / tangential content

Study tip: Rebuild each cheat-sheet section from memory on a blank page, then check it against this guide. Repetition is the fastest way to internalize HTML syntax.

