ABES Engineering College

Department of Computer Science and Engineering

HTML & CSS Tutorial

Comprehensive, Beginner-Friendly Guide (Windows/Any OS)

How to use this guide

Read top-to-bottom, or jump to any section. Code blocks are copy-paste ready. Open in a browser to see results.

Contents

1 What Are HTML and CSS?

HTML (HyperText Markup Language) gives *structure and meaning* to web pages using elements (tags). **CSS** (Cascading Style Sheets) controls *presentation*: layout, colors, fonts, responsiveness.

2 Your First Web Page

2.1 Project Structure

```
/my-site
  |-- index.html
  |-- styles.css
  |-- images/
```

2.2 Minimal HTML Skeleton

Listing 1: index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>My First Page</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <h1>Hello, world!</h1>
  <p>This is my first web page.</p>
</body>
</html>
```

2.3 Attach CSS

Listing 2: styles.css

```
/* Reset (lightweight) */
*{ box-sizing:border-box; margin:0; padding:0; }

body{
  font-family: system-ui, -apple-system, Segoe UI, Roboto, Arial, sans-serif;
  line-height: 1.6;
  padding: 24px;
}
h1{ margin-bottom: 12px; }
p{ color: #333; }
```

3 HTML Essentials

3.1 Elements, Tags, Attributes

Elements are written as ¡tag attribute=”value”¿content¡/tag¿. Some are void (self-closing) like ¡img¿, ¡br¿, ¡hr¿.

3.2 Common Text Elements

- Headings: `<h1> ... <h6>`
- Paragraph: `<p>`, emphasis: `<em>`, strong: `<strong>`
- Links: `<a href="https://example.com" target="_blank">Visit</a>`
- Lists: `<ul>`, `<ol>` with `<li>` items
- Images: `<img src="images/pic.jpg" alt="Description">` (alt is essential for accessibility)

3.3 Semantic HTML (Use Meaningful Tags)

`<header>`, `<nav>`, `<main>`, `<section>`, `<article>`, `<aside>`, `<footer>` improve structure, SEO, and accessibility.

```
<header>
  <nav>
    <a href="#home">Home</a>
    <a href="#about">About</a>
    <a href="#contact">Contact</a>
  </nav>
</header>
<main>
  <section id="home"><h1>Welcome</h1></section>
  <section id="about"><h2>About Us</h2></section>
</main>
<footer><small>&copy; 2025 My Site</small></footer>
```

3.4 Tables (When Data Truly Tabular)

```
<table>
  <caption>Scores</caption>
  <thead><tr><th>Name</th><th>Points</th></tr></thead>
  <tbody>
    <tr><td>Ada</td><td>95</td></tr>
    <tr><td>Linus</td><td>90</td></tr>
  </tbody>
</table>
```

3.5 Forms (Inputs and Labels)

```
<form action="/signup" method="post" autocomplete="on">
  <label for="name">Name</label>
  <input id="name" name="name" type="text" required>

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

  <label><input type="checkbox" name="agree" required> I agree</label>

  <button type="submit">Sign Up</button>
</form>
```

3.6 Media

```
<figure>
  <img src="images/hero.jpg" alt="Mountain view" width="600">
  <figcaption>Our campus trek</figcaption>
</figure>

<video controls width="480">
  <source src="media/intro.mp4" type="video/mp4">
  Sorry, your browser can't play this video.
</video>
```

4 CSS Fundamentals

4.1 The Cascade, Specificity, Inheritance

Order of application: Origin (user-agent, author), Specificity, Source order. Specificity: inline ¿ id ¿ class/attr/pseudo-class ¿ element.

```
/* Specificity demo */
p{ color: black; } /* low */
.message{ color: blue; } /* higher */
#urgent{ color: red; } /* highest */
```

4.2 Selectors

- Type: `p`, `h1`
- Class: `.card`, ID: `#main`
- Attribute: `a[target="_blank"]`
- Pseudo-class: `a:hover`, `:focus`, `:nth-child(2)`
- Pseudo-element: `p::first-line`, `::before`, `::after`
- Combinators: `.card p` (descendant), `.card > p` (child), `h1 + p` (adjacent)

4.3 The Box Model

Every element is a box: content + padding + border + margin.

```
.card{
  width: 300px;
  padding: 16px;
  border: 1px solid #ccc;
  margin: 12px auto;
  box-sizing: border-box; /* recommended */
}
```

4.4 Units (px, %, em, rem, vw/vh, fr)

- Absolute: `px`
- Relative to element: `em`
- Relative to root: `rem` (recommended for scalable typography)
- Viewport: `vw`, `vh`
- Grid fraction: `fr` (in CSS Grid)

4.5 Colors and Typography

Use hex, rgb/rgba, hsl/hsla. Web-safe stacks for fonts.

```
body{ color:#222; background:#f9fafb; }
h1{ color:hsl(210, 80%, 35%); }
p.muted{ color: rgba(0,0,0,0.6); }
```

5 Layout: Flexbox and Grid

5.1 Flexbox (1D Layout)

```
.nav{
  display:flex;
  gap: 12px;
  justify-content: space-between; /* main axis */
  align-items: center; /* cross axis */
}
.nav a{ padding:8px 12px; }
```

5.2 Grid (2D Layout)

```
.grid{
  display:grid;
  grid-template-columns: repeat(3, 1fr);
  gap: 16px;
}
.card{ background:white; padding:16px; }
```

6 Positioning and Stacking

```
.badge{
  position:absolute; top:8px; right:8px;
}
.header{
  position:sticky; top:0; z-index:100; /* sticks during scroll */
}
.modal{
  position:fixed; inset:0; display:grid; place-items:center;
}
```

7 Responsive Design

7.1 Viewport Meta

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

7.2 Media Queries

```
/* Mobile first */
.grid{ grid-template-columns: 1fr; }

@media (min-width: 768px){
  .grid{ grid-template-columns: repeat(3, 1fr); }
}
```

7.3 Fluid Typography and Clamp

```
:root{ --step: clamp(1rem, 2vw + 0.5rem, 2rem); }
h1{ font-size: var(--step); }
```

8 State, Effects, and Motion

8.1 Transitions

```
.button{
  background:#2563eb; color:white; padding:10px 16px; border-radius:6px;
  transition: background 200ms ease, transform 200ms ease;
}
.button:hover{ background:#1e40af; transform: translateY(-2px); }
```

8.2 Transforms and Animations

```
.box{ transform: rotate(2deg) scale(1.02); }
@keyframes pulse{
  0%{ transform: scale(1); } 50%{ transform: scale(1.05); } 100%{ transform:
      scale(1); }
}
.pulse{ animation: pulse 1.2s ease-in-out infinite; }
```

9 Best Practices

- Use semantic HTML; avoid div soup.
- Keep CSS modular; prefer classes over IDs for styling.
- Follow a naming convention (e.g., BEM: `.block__element--modifier`).
- Accessibility: labels for inputs; alt text for images; contrast and focus states.
- Performance: compress images; avoid large blocking CSS; use critical CSS if needed.
- Organization: `reset.css` or `normalize.css`, then `base.css`, `layout.css`, `components.css`.

10 Common Pitfalls

- Forgetting `<meta name="viewport">` (breaks mobile layout).
- Overusing IDs for styling (specificity battles).
- Not using `box-sizing: border-box;` (unexpected sizes).
- Mixing absolute units everywhere; prefer rem/em for scalable UI.
- Images without `alt` attributes.

11 Debugging with DevTools

Right-click → Inspect. Watch layout boxes, applied CSS, computed values, toggle states (`:hover`, `:focus`), and mobile device simulation.

12 Mini Project: Portfolio Section

12.1 HTML

```
<section class="portfolio">
  <h2>Projects</h2>
  <div class="grid">
    <article class="card">
      <h3>Sentinox AI</h3>
      <p>Real-time threat detection app.</p>
      <a class="button" href="#">View</a>
    </article>
    <article class="card">
      <h3>COA Notes</h3>
      <p>Concise study notes with diagrams.</p>
      <a class="button" href="#">View</a>
    </article>
    <article class="card">
      <h3>Java MCQs</h3>
      <p>BTL-aligned question bank.</p>
      <a class="button" href="#">View</a>
    </article>
  </div>
</section>
```

12.2 CSS

```
.portfolio{ max-width:1000px; margin:0 auto; padding:24px; }
.portfolio h2{ margin-bottom:16px; }
.grid{
  display:grid; gap:16px;
  grid-template-columns: repeat(auto-fit, minmax(220px, 1fr));
}
.card{
  background:white; border:1px solid #e5e7eb; border-radius:10px;
  padding:16px; box-shadow: 0 2px 8px rgba(0,0,0,0.04);
}
.card h3{ margin-bottom:8px; }
.button{
  display:inline-block; margin-top:8px;
  background:#111827; color:white; padding:8px 12px; border-radius:6px;
  text-decoration:none; transition: background 200ms ease;
}
.button:hover{ background:#374151; }
```

13 Exercises (Practice and Verify for Beginners)

1. **Create your first HTML page** Make a new file `index.html`. Add a title, a main heading (`<h1>`), and a short paragraph about yourself.

2. **Add more text elements** Use different heading levels (`<h2>`, `<h3>`), add a list of your hobbies (`<ul><li>` ... `</li></ul>`), and make one word bold (`<strong>`) and one italic (`<em>`).

3. **Insert a link and an image** Add a hyperlink to any website (e.g., `<a href="https://example.` Insert a picture of your choice using `<img src="..." alt="...">`.

4. **Apply basic CSS styling** Create a `styles.css` file. Change the page background color, set a font family, and give your heading a different text color.

5. **Build a simple 2-column layout** Use a `div` with two child `div`s side by side. Apply `display: flex;` in CSS to align them horizontally. In one column put text, in the other put an image.

6. **Create a basic contact form** Add inputs for name, email, and a message text area. Include a "Submit" button. Don't worry about making it functional yet.

Further Reading

- MDN Web Docs
- web.dev/learn
- HTML One Shot (YouTube Video)