

Clase Básica de HTML y CSS

Diego Muñoz

09 de noviembre de 2025

Introducción

HTML y CSS para construir páginas web estáticas y responsivas. La meta es que puedas estructurar contenido semántico y aplicar estilos limpios y mantenibles.

¿Qué es HTML? ¿Qué es CSS?

- **HTML:** Lenguaje de marcado para estructurar contenido.
- **CSS:** Lenguaje de estilos para presentar ese contenido.
- **Separación de responsabilidades:** HTML = estructura y significado; CSS = apariencia.

Estructura mínima de un documento

```
<!doctype html>
<html lang="es">
  <head>
    <meta charset="utf-8">
    <title>Mi primera página</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
  </head>
  <body>
    Hola mundo
  </body>
</html>
```

Etiquetas y atributos

- Elementos: `<p>`, `<h1>`, `<a>`, ``, ``, etc.
- Atributos: `href`, `src`, `alt`, `id`, `class`, `type`, `placeholder`, etc.
- Anidación correcta: los elementos se deben abrir y cerrar ordenadamente.

- Estructura base: header, nav, main, section, article, aside, footer.
- Mejora accesibilidad, SEO y mantenibilidad.

```
<header>
  <nav>
    <a href="/">Inicio</a>
    <a href="/acerca">Acerca</a>
  </nav>
</header>
```

```
<main>
  <article>
    <h1>Bienvenidos</h1>
    <p>Texto de introducción</p>
  </article>
</main>

<footer>
  <small>© 2025</small>
</footer>
```

<h1>Fundamentos de HTML</h1>

<p>Construir páginas web con estructura y significado.</p>

 Títulos

 Párrafos

 Listas

Aprender más

```

```

- alt describe la imagen para lectores de pantalla y cuando no carga.

Formularios básicos

```
<form action="/suscribir" method="post">
  <label for="correo">Correo</label>
  <input
    id="correo"
    name="correo"
    type="email"
    placeholder="e@mail.cl"
    required
  >
  <button type="submit">Suscribirse</button>
</form>
```

Tablas básicas

```
<table>
  <thead>
    <tr>
      <th>Plan</th>
      <th>Precio</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Básico</td>
      <td>$9</td>
    </tr>
  </tbody>
</table>
```

¿Qué es CSS?

- Define colores, tipografías, espacios, tamaños y layouts.
- Se aplica a elementos seleccionados por **selectores**.

Formas de incluir CSS

- 1) En línea (no recomendado a gran escala)
- 2) Interno (en `<style>`)
- 3) Externo (archivo `.css` enlazado) ← preferido

```
<link rel="stylesheet" href="estilos.css">
```

```
<main>
  <h1>Producto</h1>
  <p>Nuevas funciones disponibles ahora.</p>
  <a href="/comprar">Comprar ahora</a>
</main>
```

Con CSS (archivo externo)

```
main {  
    max-width: 720px;  
    margin: 0 auto;  
    padding: 16px;  
    font-family: system-ui, sans-serif;  
}  
  
a {  
    display: inline-block;  
    padding: 10px 16px;  
    text-decoration: none;  
    border: 1px solid #222;  
    border-radius: 8px;  
}
```

Selectores esenciales

- Por tipo: p, h1, img
- Por clase: .tarjeta, .boton
- Por id: #principal
- Atributo: input[type="email"]
- Pseudoclases: a:hover, button:disabled
- Pseudoelementos: p::first-line, ::selection

```
.tarjeta {}  
#principal {}  
input[type="email"] {}  
a:hover {}
```

Cascada y especificidad

- Orden de aplicación: origen + especificidad + orden.
- Especificidad: id > .clase > etiqueta.
- Evitar !important salvo casos puntuales.

```
h1 { color: #444; }  
.titulo { color: #222; }  
#tituloPrincipal { color: #000; }
```

Unidades y colores

- Unidades relativas: rem, %, vw, vh.
- Unidades absolutas: px.
- Colores: #112233, rgb(17,34,51), hsl(210, 30%, 20%).

```
:root { font-size: 16px; }
```

```
h1 { font-size: 2rem; }
```

```
section { padding: 4vw; }
```

Tipografía y espaciado

```
body {  
  font-family: Inter, system-ui, sans-serif;  
  line-height: 1.6;  
  color: #111;  
}  
  
h1, h2, h3 {  
  line-height: 1.2;  
  margin: 0 0 0.5em;  
}  
  
p { margin: 0 0 1em; }
```

Modelo de caja (Box Model)

- Cada elemento: contenido, padding, borde, margen.
- `box-sizing: border-box` simplifica el cálculo.

```
* { box-sizing: border-box; }
```

```
.tarjeta {  
    width: 320px;  
    padding: 16px;  
    border: 1px solid #ddd;  
    margin: 12px;  
    border-radius: 10px;  
}
```

Display y posicionamiento

- display: block, inline, inline-block, flex, grid.
- position: static, relative, absolute, fixed, sticky.

```
.etiqueta {  
    position: absolute;  
    top: 8px;  
    right: 8px;  
}
```

Flexbox (layout unidimensional)

```
.fila {  
    display: flex;  
    gap: 12px;  
    align-items: center;  
    justify-content: space-between;  
}  
  
.columna {  
    flex: 1;  
    padding: 12px;  
    border: 1px solid #eee;  
}
```

Grid (layout bidimensional)

```
.grid {  
  display: grid;  
  grid-template-columns: repeat(3, 1fr);  
  gap: 16px;  
}  
  
.item {  
  padding: 16px;  
  border: 1px solid #eee;  
}
```

Responsive design

- Ajustar layout y tipografías según ancho de pantalla con **media queries**.
- Mobile-first: estilos base para móviles, luego ampliar.

```
h1 { font-size: 1.5rem; }
```

```
@media (min-width: 768px) {  
    h1 { font-size: 2rem; }  
}
```

- Texto alternativo en imágenes (`alt`).
- Labels asociados a inputs (`for/id`).
- Orden lógico del contenido y uso de semántica.
- Contraste suficiente.

```
<label for="buscar">Buscar</label>
```

```
<input id="buscar" name="buscar" type="search" placeholder="Escribe aquí">
```

Componentes simples con HTML + CSS

```
<button class="boton">Comenzar</button>
```

```
.boton {  
    padding: 12px 18px;  
    border: 0;  
    border-radius: 10px;  
    background: #111;  
    color: #fff;  
    font-weight: 600;  
    cursor: pointer;  
}
```

```
.boton:hover { opacity: 0.9; }
```

Patrones comunes de layout

- Barra de navegación fija
- Grillas de información
- Pie de página con links

```
<header class="nav">
  <a href="/">Marca</a>
  <nav> <a href="#caracteristicas">Características</a> </nav>
</header>

.nav {
  display: flex; align-items: center;
  justify-content: space-between; padding: 12px 16px;
  border-bottom: 1px solid #eee; position: sticky;
  top: 0; background: #fff;
}
```

Buenas prácticas

- HTML semántico y accesible.
- CSS modular por componentes y utilidades.
- Reutilizar clases, evitar duplicación.
- Mantener consistencia de escalas (espacios, fuentes, colores).
- Medir en dispositivos reales y usar DevTools.

Proyecto de ejemplo

Referir a ejemplo en [GitHub](#).

- **HTML** define la estructura y semántica.
- **CSS** define la presentación y el layout.
- Practica con componentes simples y layouts comunes.
- Prioriza accesibilidad y responsive.
- Mantén estilos modulares y consistentes.