CSS

Cascading Style Sheets

🖍️ 2026-01 — ⏱️ 60 min.

Que es CSS?

CSS - Cascading Style Sheets

CSS es el lenguaje que define el **estilo visual** de las páginas web.

Controla colores, fuentes, espaciado, layout, animaciones, y más.

Separa el contenido (HTML) de la presentación (CSS).

Por qué usar CSS?

- ✅ **Separación de responsabilidades**: Contenido vs Presentación
- ✅ **Reutilización**: Un archivo CSS para múltiples páginas
- ✅ **Mantenimiento**: Cambiar estilos en un solo lugar
- ✅ **Responsividad**: Adaptar diseño a diferentes dispositivos

Sintaxis y Selectores

Sintaxis Básica de CSS

```
selector {
    propiedad: valor;
    otra-propiedad: otro-valor;
}
```

Ejemplo:

```
p {
    color: blue;
    font-size: 16px;
    margin: 10px;
}
```

Selectores Básicos

Selector de elemento:

```
p { color: blue; }
```

Selector de clase:

```
.mi-clase { color: red; }
```

Selector de ID:

```
#mi-id { color: green; }
```

Selector universal:

```
* { margin: 0; }
```

Pseudo-clases Comunes

```css
/* Estados de interacción */
a:hover { color: red; }
button:hover { background: lightblue; }

/* Elementos de formulario */
input:focus { border-color: blue; }
input:disabled { opacity: 0.5; }
```

Propiedades CSS Comunes

Colores

```css
/* Nombres de color */
color: red;

/* Hexadecimal */
color: #ff0000;
color: #f00; /* Forma corta */

/* RGB */
color: rgb(255, 0, 0);

/* RGBA (con transparencia) */
color: rgba(255, 0, 0, 0.5);

/* HSL */
color: hsl(0, 100%, 50%);
```

Tipografía

```css
/* Familia de fuente */
font-family: Arial, Helvetica, sans-serif;

/* Tamaño */
font-size: 16px;
font-size: 1.2rem;

/* Peso */
font-weight: normal; /* 400 */
font-weight: bold;   /* 700 */

/* Estilo */
font-style: italic;

/* Altura de línea */
line-height: 1.5;
```

Texto

```css
/* Alineación */
text-align: left | center | right | justify;

/* Decoración */
text-decoration: none | underline | line-through;

/* Transformación */
text-transform: uppercase | lowercase | capitalize;

/* Espaciado */
letter-spacing: 2px;
word-spacing: 5px;

/* Sombra */
text-shadow: 2px 2px 4px rgba(0,0,0,0.5);
```

Fondo

```css
/* Color de fondo */
background-color: #f0f0f0;

/* Imagen de fondo */
background-image: url('imagen.jpg');

/* Repetición */
background-repeat: no-repeat | repeat | repeat-x | repeat-y;

/* Posición */
background-position: center;

/* Tamaño */
background-size: cover | contain | 100px 200px;

/* Forma corta */
background: url('img.jpg') no-repeat center/cover;
```

Cascada y Especificidad

Qué es la Cascada?

La **cascada** es el mecanismo que determina qué estilos se aplican cuando hay conflictos.

Orden de prioridad:

1. **Importancia**: `!important` (evitar)
2. **Especificidad**: IDs > Clases > Elementos
3. **Orden de código**: El último estilo gana

```
/* Archivo styles.css */
p { color: blue; }
p { color: red; }  /* ✅ Este gana (último) */

.texto { color: green; }  /* ✅ Este gana (más específico) */
```

La cascada permite **heredar** y **sobreescribir** estilos de manera predecible.

La Cascada y Especificidad

Los estilos se aplican en orden de especificidad:

IDs (`#id`) > **Clases** (`.class`) > **Elementos** (`div`)

```
/* Especificidad baja */
p { color: blue; }

/* Especificidad media */
.texto { color: red; }

/* Especificidad alta */
#principal { color: green; }
```

⚠️ **Evitar** `!important` - rompe la cascada natural.

Combinando Selectores: Mayor Especificidad

Combinar selectores aumenta la especificidad:

```
/* Especificidad: 0,0,1 (1 elemento) */
p { color: blue; }

/* Especificidad: 0,1,0 (1 clase) */
.texto { color: red; }

/* Especificidad: 0,1,1 (1 clase + 1 elemento) */
p.texto { color: green; }

/* Especificidad: 0,2,1 (2 clases + 1 elemento) */
p.texto.destacado { color: purple; }

/* Especificidad: 1,0,0 (1 ID) */
#principal { color: orange; }

/* Especificidad: 1,1,1 (1 ID + 1 clase + 1 elemento) */
div#principal .texto { color: brown; }
```

Animaciones y Transiciones

Transiciones

Animan cambios de propiedades CSS:

```css
.button {
    background-color: blue;
    transition: background-color 0.3s ease;
}

.button:hover {
    background-color: red;
}

/* Múltiples propiedades */
.box {
    transition: all 0.3s ease-in-out;
}

/* Propiedades individuales */
.element {
    transition:
        opacity 0.3s ease,
        transform 0.5s cubic-bezier(0.68, -0.55, 0.265, 1.55);
}
```

Animaciones (Keyframes)

```css
/* Definir animación */
@keyframes slide-in {
    0% {
        transform: translateX(-100%);
        opacity: 0;
    }
    100% {
        transform: translateX(0);
        opacity: 1;
    }
}

/* Usar animación */
.element {
    animation: slide-in 0.5s ease-out;
}

/* Con todas las propiedades */
.element {
    animation: slide-in 1s ease-in-out 0.5s infinite alternate;
    /*       nombre duración timing delay iteraciones dirección */
}
```

Transform

```css
/* Trasladar */
transform: translate(50px, 100px);
transform: translateX(50px);

/* Escalar */
transform: scale(1.5);
transform: scale(2, 0.5); /* x, y */

/* Rotar */
transform: rotate(45deg);

/* Sesgar */
transform: skew(10deg, 5deg);

/* Combinar */
transform: translate(50px) rotate(45deg) scale(1.2);
```

El Modelo de Caja (Box Model)

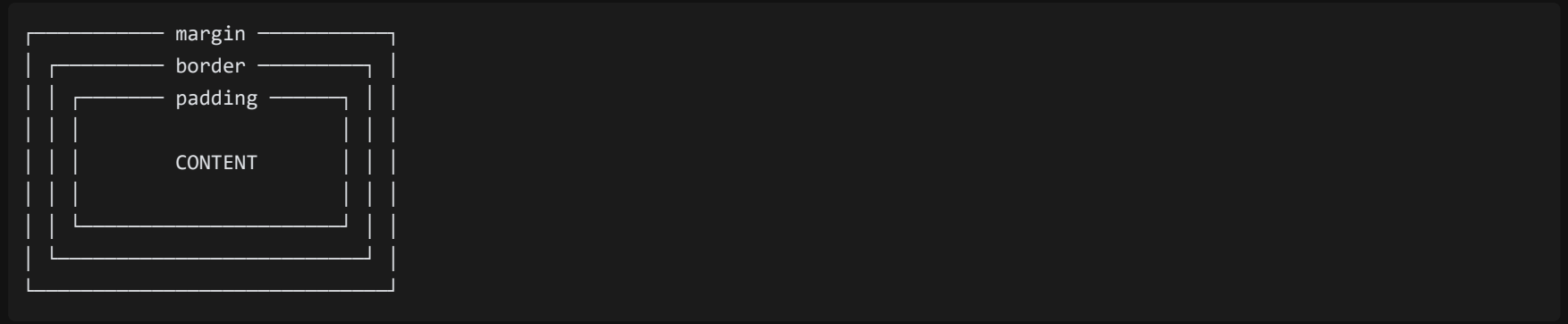
Box Model

Cada elemento HTML es una caja rectangular con:

1. **Content**: El contenido (texto, imagen, etc.)
2. **Padding**: Espacio interno entre contenido y borde
3. **Border**: El borde de la caja
4. **Margin**: Espacio externo entre la caja y otros elementos

```
div {
    width: 300px;
    padding: 20px;
    border: 2px solid black;
    margin: 10px;
}
```

Box Model - Visualización



Ancho total = width + padding + border + margin

Box Sizing

```css
/* Por defecto (content-box) */
div {
    box-sizing: content-box;
    width: 300px; /* Solo el contenido */
    padding: 20px; /* Se suma al width */
}

/* Border-box (recomendado) */
div {
    box-sizing: border-box;
    width: 300px; /* Incluye padding y border */
    padding: 20px; /* No se suma al width */
}
```

Mejor práctica: Usar `border-box` globalmente

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

Padding y Margin

```css
/* Individual */
padding-top: 10px;
padding-right: 20px;
padding-bottom: 10px;
padding-left: 20px;

/* Forma corta - 4 valores (arriba, derecha, abajo, izquierda) */
padding: 10px 20px 10px 20px;

/* Forma corta - 2 valores (vertical, horizontal) */
padding: 10px 20px;

/* Forma corta - 1 valor (todos los lados) */
padding: 10px;
```

Lo mismo aplica para `margin`.

Bordes

```css
/* Forma individual */
border-width: 2px;
border-style: solid;
border-color: black;

/* Forma corta */
border: 2px solid black;

/* Bordes específicos */
border-top: 1px solid red;
border-right: 2px dashed blue;

/* Bordes redondeados */
border-radius: 5px;
border-radius: 50%; /* Círculo */
```

Display y Position

Display

```css
/* Bloque (ocupa toda la línea) */
display: block;

/* En línea (solo el espacio necesario) */
display: inline;

/* En línea-bloque (híbrido) */
display: inline-block;

/* Ocultar elemento */
display: none;

/* Flexbox */
display: flex;

/* Grid */
display: grid;
```

Position

```css
/* Estático (por defecto) */
position: static;

/* Relativo (relativo a su posición original) */
position: relative;
top: 10px;
left: 20px;

/* Absoluto (relativo al ancestro posicionado) */
position: absolute;

/* Fijo (relativo al viewport) */
position: fixed;

/* Sticky (híbrido relativo-fijo) */
position: sticky;
top: 0;
```

Z-Index

Controla el orden de apilamiento de elementos posicionados:

```css
.modal {
    position: fixed;
    z-index: 1000; /* Encima */
}

.overlay {
    position: fixed;
    z-index: 999; /* Debajo del modal */
}

.content {
    position: relative;
    z-index: 1; /* Nivel normal */
}
```

Valores más altos = más cerca del usuario.

CSS Variables (Custom Properties)

CSS Variables

```css
/* Definir variables en :root */
:root {
    --primary-color: #007bff;
    --secondary-color: #6c757d;
    --spacing-unit: 8px;
    --font-main: 'Arial', sans-serif;
}

/* Usar variables */
.button {
    background-color: var(--primary-color);
    padding: var(--spacing-unit);
    font-family: var(--font-main);
}

/* Con valor por defecto */
.element {
    color: var(--undefined-color, black);
}
```

Variables - Ámbito y Herencia

```css
:root {
    --spacing: 10px;
}

.container {
    /* Redefinir para este ámbito */
    --spacing: 20px;
    padding: var(--spacing); /* 20px */
}

.container .child {
    padding: var(--spacing); /* 20px (heredado) */
}

.other {
    padding: var(--spacing); /* 10px (del :root) */
}
```

Variables - Uso Común: Temas

```css
:root {
    --bg-color: white;
    --text-color: black;
}

[data-theme="dark"] {
    --bg-color: #1a1a1a;
    --text-color: white;
}

body {
    background-color: var(--bg-color);
    color: var(--text-color);
}
```

```javascript
// Cambiar tema con JavaScript
document.body.setAttribute('data-theme', 'dark');
```

Mejores Prácticas

Mejores Prácticas

- ✅ **Usar nombres de clase descriptivos**
- ✅ **Usar variables CSS** para valores reutilizables
- ✅ **Evitar** `!important`
- ✅ **Mobile First** approach con media queries
- ✅ **Preferir Flexbox y Grid** para layouts

CSS Moderno: Características Útiles

```css
/* Clamp - tamaño responsive */
font-size: clamp(1rem, 2.5vw, 2rem);

/* Aspect Ratio */
.video {
    aspect-ratio: 16 / 9;
}

/* Gap en Flexbox */
.flex-container {
    display: flex;
    gap: 20px;
}

/* Logical Properties */
margin-inline: 20px; /* margin-left + margin-right */
padding-block: 10px; /* padding-top + padding-bottom */
```

Recursos de Aprendizaje

Documentación:

- MDN Web Docs: https://developer.mozilla.org/es/docs/Web/CSS
- CSS Tricks: https://css-tricks.com

Práctica Interactiva:

- Flexbox Froggy: https://flexboxfroggy.com
- Grid Garden: https://cssgridgarden.com

Herramientas:

- Can I Use: https://caniuse.com