








# CSS: Selectores y Modelos de Caja

El diseño de la web

Fundamentos de la Web - Clase 02

# Objetivos de Hoy

## ¿Qué aprenderemos?

-  ¿Qué es CSS y por qué es importante?
-  Configurar proyectos HTML + CSS
-  Selectores CSS básicos y avanzados
-  Propiedades de color y texto
-  Tamaños y bordes
-  El modelo de caja (margin, padding, border)
-  Buenas prácticas de CSS

# ¿Qué es CSS?






# CSS: El Diseño de la Web

## Cascading Style Sheets (Hojas de Estilo en Cascada)

CSS es el lenguaje que le da vida visual a tus páginas HTML

Piensa en una casa:







-  **HTML** es la estructura (paredes, techo, puertas)
-  **CSS** es la decoración (pintura, muebles, cortinas)
-  **JavaScript** es la funcionalidad (luz, agua, calefacción)

Sin CSS, todas las páginas web se verían iguales y aburridas 

# ¿Qué Podemos Hacer con CSS?

¡Un montón de cosas increíbles!

Con CSS podemos cambiar:

-  **Colores** - Texto, fondos, bordes
-  **Tamaños** - Ancho, alto, fuentes
-  **Fuentes** - Tipo, peso, estilo
-  **Espaciado** - Márgenes, rellenos
-  **Posición** - Dónde se ubican los elementos
-  **Efectos** - Sombras, transiciones, animaciones

¡Y mucho más!

## Configurar Proyectos

# Estructura de un Proyecto Web

A medida que añadimos CSS a nuestros proyectos, necesitamos mantener todo organizado:

```

MiProyecto/
├── index.html
└── style.css

```

¿Por qué esta estructura?

- `index.html` → El punto de entrada (el navegador la busca primero)
- `style.css` → Todos los estilos en un solo lugar
- **Organización** → Fácil de encontrar y mantener

Es como tener tu closet ordenado 

# Conectando HTML y CSS

La etiqueta `<link>` es la clave

index.html:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Mi Sitio Estilizado</title>
  <link rel="stylesheet" href="style.css" />
</head>
<body>
  <h1>¡Hola Mundo con Estilo!</h1>
</body>
</html>
```



## Conectando HTML y CSS

La etiqueta `<link>` es la clave

Atributos de `<link>`:

- `rel="stylesheet"` → Le dice que es una hoja de estilos
- `href="style.css"` → La ruta al archivo CSS



# Tu Primer CSS

## Cambiamos el color de un título

style.css:

```
h1 {  
  color: rebeccapurple;  
  font-size: 50px;  
}
```

¿Qué hace este código?

- `h1` → Selecciona todos los títulos h1
- `color` → Cambia el color del texto
- `font-size` → Cambia el tamaño de la fuente

¡Así de simple! ✨

# Sintaxis de CSS

## Anatomía de una regla CSS

```
selector {  
  propiedad: valor;  
  propiedad2: valor2;  
}
```

- **Selector** → ¿A qué elemento aplicar los estilos?
- **Propiedad** → ¿Qué quieres cambiar?
- **Valor** → ¿Cómo quieres que se vea?
- **{ }** → Las llaves agrupan todas las propiedades
- **;** → Punto y coma al final de cada propiedad

## ? Pregunta Rápida

¿Qué crees que hace este código CSS?

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

Piensa unos segundos... 🤔



## Respuesta

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

Este código:

- Selecciona todos los párrafos ( `<p>` )
- Les da color azul al texto
- Les pone un tamaño de fuente de 16 píxeles

Muy bien si lo adivinaste 🙌

# Selectores CSS

# ¿Qué son los selectores?




Los selectores son patrones que indican a CSS qué elementos HTML debe estilizar. Funcionan como "buscadores" que identifican elementos específicos en tu página para aplicarles reglas de estilo.

## Características principales:

- Permiten apuntar a elementos específicos sin modificar el HTML
- Son la base para crear estilos reutilizables
- Pueden ser simples (un elemento) o complejos (combinaciones)

# Los 3 Selectores Principales

## Tres formas de seleccionar elementos

1.  **Por etiqueta** → Selecciona todos los elementos del mismo tipo
2.  **Por clase** → Selecciona elementos con una clase específica
3.  **Por ID** → Selecciona un elemento único

Cada uno tiene su propósito específico



# Repaso de Selectores

## Etiqueta HTML:

- Son los elementos básicos como `<p>` , `<h1>` , `<div>`
- Ejemplo: `<p>Este es un párrafo</p>`

## Clase:

- Atributo que puede aplicarse a cualquier elemento
- Se define con `class="nombre-clase"` en HTML
- Puede usarse en múltiples elementos

## ID:

- Identificador único que solo debe usarse una vez por página
- Se define con `id="identificador"` en HTML

# ? Ejercicio: Identifica los Selectores

Observa este código HTML y encuentra:

```
<div class="card" id="profile">
  <h2 class="title">Juan Pérez</h2>
  <p>Desarrollador Web Full Stack</p>
  
  <button class="btn">Contactar</button>
</div>
```

1. ¿Cuántas **etiquetas** diferentes hay?
2. ¿Cuántas **clases** se usan?
3. ¿Cuántos **IDs** hay?

Piensa unos segundos... 🤔

## ✓ Respuesta: Identificando Selectores

```
<div class="card" id="profile">  
  <h2 class="title">Juan Pérez</h2>  
  <p>Desarrollador Web Full Stack</p>  
    
  <button class="btn">Contactar</button>  
</div>
```

🏷️ Etiquetas (5): `div`, `h2`, `p`, `img`, `button`

🎨 Clases (4): `card`, `title`, `avatar`, `btn`

ID `profile`

¿Las identificaste todas? 🎉

# Selector por Etiqueta

Selecciona todos los elementos de ese tipo

```
h1 {  
  color: blue;  
}  
  
p {  
  text-align: right;  
}
```

¿Qué hace este código?

- Todos los `<h1>` serán azules
- Todos los `<p>` estarán alineados a la derecha

Perfecto cuando quieres un estilo consistente para todo ✨

## Ejemplo de Selector por Etiqueta

```
<!DOCTYPE html>
<html lang="es">
<head>
  <style>
    h1 { color: blue; }
    p { text-align: right; }
  </style>
</head>
<body>
  <h1>Título 1</h1>           <!-- Será azul -->
  <h1>Título 2</h1>           <!-- También será azul -->
  <p>Párrafo 1</p>            <!-- Alineado a la derecha -->
  <p>Párrafo 2</p>            <!-- También alineado a la derecha -->
</body>
</html>
```

¿Ves el patrón? Todos los elementos del mismo tipo tienen el mismo estilo

# ? Pregunta

Si tengo este CSS:

```
h2 {  
  color: green;  
  font-size: 24px;  
}
```

¿Qué elementos se verán afectados?

- A) Solo el primer `<h2>` de la página
- B) Todos los `<h2>` de la página
- C) Solo los `<h2>` con clase especial

Piensa la respuesta... 🤔



## Respuesta

```
h2 {  
  color: green;  
  font-size: 24px;  
}
```

Respuesta correcta: B) Todos los `<h2>` de la página

El selector de etiqueta afecta a **todos** los elementos de ese tipo en toda la página.

**Recuerda:** Si quieres selectividad, usa clases o IDs

# Selector por Clase

## Para estilos reutilizables

### HTML:

```
<p class="is-centered">Este texto está centrado</p>
<p>Este texto está normal</p>
<p class="is-orange">Este texto es naranja</p>
```

### CSS:

```
.is-centered {
  text-align: center;
}

.is-orange {
  color: orange;
}
```

 Las clases se seleccionan con un punto ( . )



# ¿Por Qué Usar Clases?

## Ventajas de las clases

- ✓ **Reutilizables** - Puedes usar la misma clase en múltiples elementos
- ✓ **Selectivas** - Solo afectan a elementos específicos
- ✓ **Flexibles** - Un elemento puede tener varias clases

```
<p class="importante destacado">Este párrafo tiene 2 clases</p>
```

Las clases son tu mejor amiga en CSS 

# Selector por ID: Para elementos únicos

HTML:

```
<div id="container">  
    
</div>
```

CSS:

```
#container {  
  width: 200px;  
  height: 200px;  
  padding: 10px;  
}  
#avatar {  
  width: 150px;  
  height: 150px;  
  border-radius: 50%;  
}
```

 Los IDs se seleccionan con almohadilla ( # )

# Diferencia: Clase vs ID

¿Cuándo usar cada uno?

Característica	Clase ( . )	ID ( # )
Unicidad	Múltiples elementos	Solo un elemento
Reutilizable	✓ Sí	✗ No

**Regla de oro:** Usa clases para estilos, IDs para elementos únicos

**Especificidad:** Es el peso o importancia que tiene un selector en CSS. Determina qué estilos se aplican cuando hay conflictos. Cuanto mayor sea la especificidad, más prioridad tendrá el estilo.

# ? Pregunta Rápida

¿Cuál selector usarías para...?

- A) Cambiar el color de **todos** los párrafos
- B) Dar estilo a un grupo de botones similares
- C) Estilizar el logo único de tu sitio

Piensa qué selector usarías en cada caso... 🤔

## Respuesta

A) Cambiar el color de **todos** los párrafos

```
p { color: blue; } /* Selector de etiqueta */
```

B) Dar estilo a un grupo de botones similares

```
.btn { /* ... */ } /* Selector de clase */
```

C) Estilizar el logo único de tu sitio

```
#logo { /* ... */ } /* Selector de ID */
```

¿Las acertaste todas? 🎉



# Ejercicio: Selectores Básicos

¡Hora de practicar!

Crea un archivo `selectores.html` con:

1. 3 títulos diferentes ( `<h1>` , `<h2>` , `<h3>` )
2. 3 párrafos, cada uno con una clase diferente
3. Un div con un ID único
4. Aplica estilos usando los 3 tipos de selectores

Pistas:

- Haz que todos los h2 sean rojos
- Una clase para texto centrado
- Un ID para un contenedor con fondo gris



**Tiempo:** 10 minutos

# Selectores Avanzados

# 🌟 Selector Comodín (\*)

## El selector universal

```
* {  
  margin: 0;  
  padding: 0;  
  box-sizing: border-box;  
}
```

¿Qué hace el `*`?

- Selecciona **absolutamente todos** los elementos
- Útil para resetear estilos por defecto
- Común al inicio de un CSS

⚠ **Advertencia:** Tiene la especificidad más baja (0.1)



# Especificidad de Selectores

## ¿Quién gana cuando hay conflictos?

La especificidad es como un sistema de puntos:

Selector	Puntos	Ejemplo
* (comodín)	0.1	* { }
elemento	1	p { }
.clase	10	.destacado { }
#id	100	#principal { }

Regla de oro: El selector con más puntos gana 🏆

## Ejemplo de Especificidad

```
<h1 id="titulo" class="rojo">¿De qué color seré?</h1>
```

```
h1 { color: blue; }           /* Especificidad: 1 */
.rojo { color: red; }         /* Especificidad: 10 */
#titulo { color: green; }     /* Especificidad: 100 */
```

¿Qué color tendrá el h1?

Piensa antes de pasar a la siguiente slide... 🤔

## ✓ Respuesta: Especificidad

```
h1 { color: blue; }           /* Puntos: 1 */  
.rojo { color: red; }         /* Puntos: 10 */  
#titulo { color: green; }     /* Puntos: 100 ← GANADOR */
```

El h1 será VERDE ●

¿Por qué?

- El ID ( #titulo ) tiene la mayor especificidad (100 puntos)
- Gana sobre la clase (10 puntos) y el elemento (1 punto)

**Importante:** Aunque escribas el ID primero, igual gana por especificidad



# Selector Descendente

## Elementos dentro de otros

```
.quote p {  
  color: darkslategrey;  
  font-style: italic;  
}
```

```
<div class="quote">  
  <p>"CSS es como el ajedrez. Aprendes los conceptos básicos  
    en un día y pasas toda la vida intentando dominarlo."</p>  
</div>
```

## ¿Cómo funciona?

- Selecciona todos los `<p>` que estén **dentro** de `.quote`
- No importa qué tan profundo estén (hijos, nietos, bisnietos...)



# Diferencia: Descendente vs Directo

## Entendiendo la jerarquía

Descendente (con espacio):

```
.container p { } /* Todos los p dentro, sin importar profundidad */
```

Descendente Directo (con `>`):

```
.container > p { } /* Solo p que son hijos directos */
```

Como una familia: Descendente incluye hijos, nietos, bisnietos...

Directo solo incluye a los hijos



## Ejemplo: Descendente Directo

```
.header > img {  
  height: 50px;  
}
```

```
<div class="header">  
    
  <div>  
      
  </div>  
</div>
```



Hijo directo - SE SELECCIONA



Nieto - NO se selecciona

### ¿Cuándo usar directo?

- Cuando solo quieres afectar el primer nivel
- Para evitar afectar elementos anidados profundamente

# Combinar Selectores

Puedes combinar selectores para más especificidad:

```
#container p {  
  font-size: 16px;  
  background-color: #333;  
}
```

```
#container p.highlight {  
  font-weight: bold;  
  color: #ff0000;  
}
```

```
<div id="container">  
  <p class="highlight">Párrafo resaltado</p>  
  <p>Otro párrafo</p>  
</div>
```

# Selector de Atributo

Selecciona elementos por sus atributos:

```
input[type="text"] {  
    border: none;  
    border-bottom: 2px solid rebeccapurple;  
}  
  
input[type="submit"] {  
    color: white;  
    background-color: rebeccapurple;  
}
```

Útil para formularios y elementos específicos



## Práctica: Selectores Avanzados

### Ejercicio:

Crea un formulario con varios campos:

- Inputs de texto
- Un botón de submit
- Organiza los campos dentro de un div contenedor

Aplica estilos usando:

- Selectores descendentes
- Selectores de atributo




 **Tiempo:** 15 minutos

# Propiedades CSS

# Colores en CSS

## Tres formas de definir colores

CSS nos da flexibilidad para definir colores:

1.  **Nombres** - Fácil de recordar
2.  **RGB** - Mezcla de rojo, verde y azul
3.  **Hexadecimal** - Código de 6 caracteres




Cada método tiene sus ventajas ✨



# Colores con Nombre: 140+ colores predefinidos

```
p {  
  color: slategray;  
}  
h1 {  
  color: tomato;  
}  
div {  
  background-color: lightblue;  
}
```

## Ventajas:

-  Fácil de leer y recordar
-  No necesitas códigos
-  Opciones limitadas (solo ~140 colores)

## 12 RGB - Red, Green, Blue

```
a {  
  color: rgb(218, 112, 214); /* Lavanda */  
}  
div {  
  background-color: rgb(255, 0, 0); /* Rojo puro */  
}
```

- Cada valor va de 0 a 255
- `rgb(255, 0, 0)` → Rojo al máximo, verde y azul en 0
- `rgb(0, 255, 0)` → Verde al máximo
- `rgb(0, 0, 255)` → Azul al máximo
- `rgb(255, 255, 255)` → Blanco (todos al máximo)
- `rgb(0, 0, 0)` → Negro (todos en 0)

# # Hexadecimal: Códigos de 6 dígitos

```
button {  
    background-color: #ff6347; /* Rojo coral */  
}  
header {  
    background-color: #2196f3; /* Azul material */  
}
```

**Formato:** #RRGGBB

- Cada par representa: Rojo, Verde, Azul / Valores: 00 (mínimo) a FF (máximo)
- #FF0000 → Rojo puro
- #00FF00 → Verde puro
- #0000FF → Azul puro

Es el formato más usado por diseñadores 🎨

# ? Pregunta Rápida

¿Qué color hace este código?

```
div {  
  background-color: rgb(255, 255, 0);  
}
```

- A) Rojo
- B) Amarillo
- C) Celeste
- D) Blanco

Piensa... 🤔

## ✓ Respuesta

```
div {  
  background-color: rgb(255, 255, 0);  
}
```

Respuesta: B) Amarillo ●

¿Por qué?

- Rojo al máximo (255) ✓
- Verde al máximo (255) ✓
- Azul en cero (0) ✗

**Rojo + Verde = Amarillo** (teoría del color)





# Color vs Background-Color

No son lo mismo

```
h1 {  
  color: white;           /* Color del TEXTO */  
  background-color: black; /* Color del FONDO */  
}
```

Resultado:

-  Texto blanco
-  Fondo negro

¡No los confundas!

- `color` → Texto
- `background-color` → Fondo

# Recursos de Colores

## Herramientas útiles

 Aprende más:

- [Nombres de colores](#) - Todos los nombres disponibles
- [Colores en HTML](#) - Guía completa
- [Color Picker](#) - Elige colores visualmente






 **Tip:** Google "color picker" para encontrar códigos de colores fácilmente



# Propiedades de Texto

## Dale estilo a tus palabras

Las propiedades de texto controlan cómo se ve el contenido escrito:

-  Alineación
-  Decoración
-  Peso (grosor)
-  Estilo
-  Familia de fuente

# Alineación de Texto

## text-align

```
.izquierda {  
    text-align: left;    /* Default - izquierda */  
}  
  
.centrado {  
    text-align: center; /* Centro */  
}  
  
.derecha {  
    text-align: right;   /* Derecha */  
}  
  
.justificado {  
    text-align: justify; /* Justificado */  
}
```

Como en Word: Alinea el texto horizontalmente

# Decoración y Estilo de Texto

## Subrayado, tachado y más

```
a {  
    text-decoration: underline;    /* Subrayado */  
    text-decoration: line-through; /* Tachado */  
    text-decoration: none;        /* Sin decoración */  
}  
  
em {  
    font-style: italic;    /* Cursiva */  
    font-style: normal;    /* Normal */  
}
```

**Tip:** Los enlaces ( `<a>` ) vienen subrayados por defecto.

Usa `text-decoration: none` para quitarlo.



# Peso de la Fuente

## font-weight - Grosor de la fuente

```
h1 {  
  font-weight: normal; /* 400 - Normal */  
  font-weight: bold; /* 700 - Negrita */  
  font-weight: 100; /* Muy delgado */  
  font-weight: 500; /* Medio */  
  font-weight: 900; /* Muy grueso */  
}
```

### Valores:

- **Palabras:** `normal`, `bold`, `bolder`, `lighter`
- **Números:** 100, 200, 300, 400, 500, 600, 700, 800, 900
  - 400 = normal
  - 700 = bold



# Familia de Fuentes

## font-family - El tipo de letra

```
body {  
    font-family: sans-serif; /* Sin serifas (moderna) */  
}  
  
.titulo {  
    font-family: serif;      /* Con serifas (clásica) */  
}  
  
code {  
    font-family: monospace;  /* Monoespacio (código) */  
}  
  
.fancy {  
    font-family: cursive;    /* Cursiva (decorativa) */  
}
```

Aplica al **body** para que toda la página use la misma fuente



## Ejemplo de Fuentes

```
body {  
  font-family: Arial, Helvetica, sans-serif;  
}
```

### ¿Por qué tres fuentes?

- Funciona como un **fallback** (respaldo)
- Si Arial no está disponible, usa Helvetica
- Si ninguna está, usa cualquier sans-serif del sistema

Siempre termina con una familia genérica:

serif, sans-serif, monospace, etc.



## Ejercicio Práctico: Estilo Box Model

### Box Model

¡Hola! Soy el primer div de color light sea green.

¡Hola! Soy el segundo div fucsia.

¡Hola! Soy el tercer div azul marino.

## Ejercicio Práctico: Estilo Box Model

Pistas para resolver:

- Piensa en cómo centrar elementos en la página
- Considera qué propiedad CSS permite que elementos de bloque aparezcan uno al lado del otro
- Recuerda definir dimensiones para los elementos
- Usa márgenes para crear espacio entre elementos
- Aplica colores de fondo diferentes a cada elemento
- Considera cómo dar un estilo especial al título principal





 **Tiempo:** 15 minutos

## Tamaños en CSS

# Controlando Tamaños

## Ancho, Alto y Más

En CSS podemos controlar el tamaño de los elementos de varias formas:

-  **width** - El ancho del elemento
-  **height** - La altura del elemento
-  **font-size** - El tamaño del texto
-  **Unidades** - Píxeles, porcentajes, em, rem...

Controlar tamaños es fundamental para diseñar interfaces 



# Ancho y Alto: width y height

## Valores absolutos con píxeles

```
button {  
  width: 100px; /* Ancho de 100 píxeles */  
  height: 50px; /* Alto de 50 píxeles */  
}
```

### ¿Qué son los píxeles (px)?

- Son puntos en la pantalla
- **Valor fijo** - Siempre mide lo mismo
- Perfecto cuando quieres un tamaño exacto

Como medir con una regla 



# Tamaños Relativos: Porcentajes

## Adaptándose al contenedor

```
div {  
  width: 80%;      /* 80% del contenedor padre */  
  height: 200px;   /* Alto fijo de 200px */  
}
```

## ¿Cómo funcionan los porcentajes?

- Se calculan respecto al **elemento padre**
- Si el padre mide 1000px, 80% = 800px
- **Flexibles** - Se adaptan si cambia el tamaño del padre

Útil para diseños responsivos  

# ? Pregunta Rápida

¿Qué ancho tendrá este div?

```
<div class="contenedor" style="width: 500px;">  
  <div class="hijo" style="width: 50%;">  
    Contenido  
  </div>  
</div>
```

- A) 50 píxeles
- B) 250 píxeles
- C) 500 píxeles

Piensa unos segundos... 🤔

## ✓ Respuesta

```
<div class="contenedor" style="width: 500px;">  
  <div class="hijo" style="width: 50%;">  
    Contenido  
  </div>  
</div>
```

Respuesta correcta: B) 250 píxeles

¿Por qué?

- El contenedor padre mide 500px
- El hijo tiene width: 50%
- 50% de 500px = 250px

¡El porcentaje siempre se calcula del padre! 👨👦



# Tamaño de Fuente: font-size

## Tres formas de definir tamaños de texto

CSS nos ofrece diferentes unidades para el texto:

1. **px** - Píxeles (fijo)
2. **%** - Porcentaje (relativo al padre)
3. **em** - Relativo al tamaño heredado




Cada una tiene su propósito ✨



# Font-Size con Píxeles: Tamaño fijo y predecible

```
p {  
    font-size: 16px; /* Tamaño estándar de lectura */  
}  
h1 {  
    font-size: 32px; /* El doble de grande */  
}  
small {  
    font-size: 12px; /* Texto pequeño */  
}
```

## Ventajas:

-  Fácil de entender
-  Siempre mide lo mismo
-  No se adapta bien en diferentes dispositivos



# Font-Size con Porcentaje: Relativo al elemento padre

```
body {  
    font-size: 16px; /* Base */  
}  
p {  
    font-size: 100%; /* = 16px (mismo que el body) */  
}  
h1 {  
    font-size: 200%; /* = 32px (el doble del body) */  
}  
small {  
    font-size: 75%; /* = 12px (75% de 16px) */  
}
```

Más flexible que píxeles fijos 🌟



# Font-Size con Em: Relativo al tamaño heredado

```
body {  
    font-size: 16px; /* Tamaño base */  
}  
h1 {  
    font-size: 2em; /* 2 veces el tamaño base = 32px */  
}  
p {  
    font-size: 1em; /* 1 vez el tamaño base = 16px */  
}  
small {  
    font-size: 0.75em; /* 0.75 veces = 12px */  
}
```

1em = el tamaño de fuente heredado del padre



Recomendado: Usa `em` o `%` para mejor adaptabilidad

# ? Pregunta: ¿Qué Tamaño Tendrá?

Analiza este código:

```
body {  
    font-size: 20px;  
}  
  
.texto {  
    font-size: 1.5em;  
}
```

¿Qué tamaño tendrá el texto con clase `.texto` ?

- A) 15px
- B) 20px
- C) 30px

Piensa... 🤔



## Respuesta

```
body {  
    font-size: 20px;  
}  
.texto {  
    font-size: 1.5em; /* 1.5 veces el tamaño del padre */  
}
```

Respuesta correcta: C) 30px

¿Por qué?

- El body tiene 20px
- $1.5\text{em} = 1.5 \times 20\text{px}$
- $1.5 \times 20 = 30\text{px}$

¡El em se multiplica por el tamaño heredado! 🎯

# Overflow: Cuando el contenido es muy grande

¿Qué pasa si el contenido no cabe?

Imagina una caja de 100px × 100px con un texto muy largo...  

CSS te permite controlar qué hacer:

```
div {  
  overflow: visible; /* Dejar que se salga (default) */  
  overflow: hidden; /* Ocultar lo que sobra */  
  overflow: scroll; /* Siempre mostrar scrollbar */  
  overflow: auto; /* Scrollbar solo si es necesario */  
}
```

# Overflow: Ejemplos Visuales

## visible (default)

El contenido se sale de la caja  Puede verse desordenado

## hidden

Recorta el contenido que no cabe  Se pierde lo que sobra

## scroll

Agrega barras de desplazamiento siempre  Incluso si no es necesario

## auto

Solo agrega scrollbar cuando hace falta  La más usada





# Border: Dale Bordes a tus Elementos

## La anatomía de un borde

Un borde tiene 3 partes:

```
border: 2px solid black;
/*      ▲      ▲      ▲
      grosor estilo color */
```

Orden:

1. **Grosor** - ¿Qué tan grueso? ( 1px , 2px , 5px ...)
2. **Estilo** - ¿Qué apariencia? ( solid , dashed , dotted ...)
3. **Color** - ¿De qué color? (cualquier color CSS)



# Estilos de Borde

## Diferentes apariencias

```
.solido {  
    border: 2px solid black;    /* Línea sólida — */  
}  
  
.punteado {  
    border: 2px dotted blue;    /* Puntos ..... */  
}  
  
.guiones {  
    border: 2px dashed red;     /* Guiones - - - */  
}  
  
.doble {  
    border: 3px double green;    /* Línea doble == */  
}
```

¡Experimenta con diferentes estilos! 🎨



# Bordes por Lados

## Controla cada lado individualmente

```
div {  
  border-top: 2px solid red;      /* Solo arriba */  
  border-right: 1px dashed blue; /* Solo derecha */  
  border-bottom: 3px dotted green; /* Solo abajo */  
  border-left: 2px solid orange; /* Solo izquierda */  
}
```

## O todos a la vez:

```
div {  
  border: 2px solid black; /* Los 4 lados iguales */  
}
```

Útil para efectos creativos ✨

# ○ Border-Radius: Esquinas Redondeadas

## Suaviza las esquinas

```
/* Bordes ligeramente redondeados */  
button {  
    border-radius: 5px;  
}
```

Sin border-radius: □ Esquinas puntiagudas

Con border-radius: □ Esquinas suaves



Hace que los elementos se vean más modernos 🎨

# ○ Crear Círculos Perfectos

## El truco del 50%

```
.circulo {  
  width: 100px;  
  height: 100px;  
  border-radius: 50%; /* ¡Círculo perfecto! */  
  background-color: tomato;  
}
```

### Requisitos para círculos:

-  El width y height deben ser **iguales**
-  Usa `border-radius: 50%`

Perfecto para avatares, botones circulares, etc 

# ? Pregunta Rápida

¿Qué forma tendrá este elemento?

```
div {  
  width: 150px;  
  height: 150px;  
  border-radius: 50%;  
  background-color: blue;  
}
```

- A) Un cuadrado azul
- B) Un rectángulo azul con esquinas redondeadas
- C) Un círculo azul perfecto

Piensa la respuesta... 🤔

## ✓ Respuesta

```
div {  
  width: 150px;  
  height: 150px;  
  border-radius: 50%;  
  background-color: blue;  
}
```

Respuesta correcta: C) Un círculo azul perfecto ●

¿Por qué?

- Width y height son iguales (150px)
- border-radius: 50% convierte el cuadrado en círculo

¡El 50% es la clave! 🔑






## El Modelo de Caja



# El Concepto Más Importante de CSS

## Todo es una caja

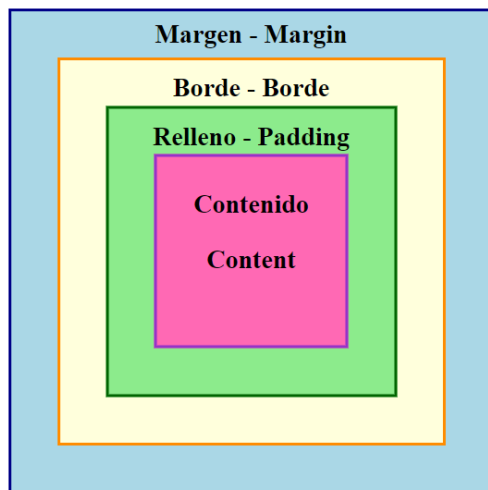
En CSS, **absolutamente todo** es una caja rectangular:





- Los títulos `<h1>` son cajas 
- Los párrafos `<p>` son cajas 
- Los divs `<div>` son cajas 
- Las imágenes `<img>` son cajas 
- ¡Los botones `<button>` son cajas! 

Entender esto es la clave para dominar CSS 

# Anatomía de una Caja

## Las 4 capas del modelo de caja



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 (separación con otros elementos)

Cada capa tiene su propósito ✨



# Content: El Contenido

## El centro de la caja

```
div {  
  width: 200px; /* Ancho del contenido */  
  height: 100px; /* Alto del contenido */  
}
```

### El content es:

- El texto que escribes
- Las imágenes que insertas
- Cualquier elemento hijo

Todo lo demás se agrega alrededor del content 🎯

# Padding: El Espacio Interno

## Aire entre el contenido y el borde

```
div {  
  padding: 20px; /* 20px de espacio interno en todos los lados */  
}
```

### ¿Para qué sirve el padding?

- Evita que el texto toque el borde
- Da "respiro" al contenido
- Hace que el elemento se vea más espacioso

Como el acolchado de un sofá 

## Padding: Ejemplo Visual

### Sin padding vs con padding

Sin padding:


```
div {  
  border: 2px solid black;  
  padding: 0; /* El texto toca el borde ✗ */  
}
```

[Texto pegado al borde]

# Padding: Ejemplo Visual

Sin padding vs con padding

Con padding:

```
div {  
  border: 2px solid black;  
  padding: 20px; /* Espacio cómodo  */  
}
```

[      Texto      ]



# Border: El Borde

## El marco de la caja

```
div {  
  border: 2px solid black; /* Borde de 2px */  
}
```

### El border:

- Rodea el contenido y el padding
- Es visible (a menos que sea transparente)
- Tiene grosor, estilo y color

Como el marco de un cuadro 

# Margin: El Espacio Externo

## Separación con otros elementos

```
div {  
  margin: 30px; /* 30px de separación externa */  
}
```

### ¿Para qué sirve el margin?

- Separa elementos entre sí
- Crea espacio en blanco
- Evita que todo se vea apretado

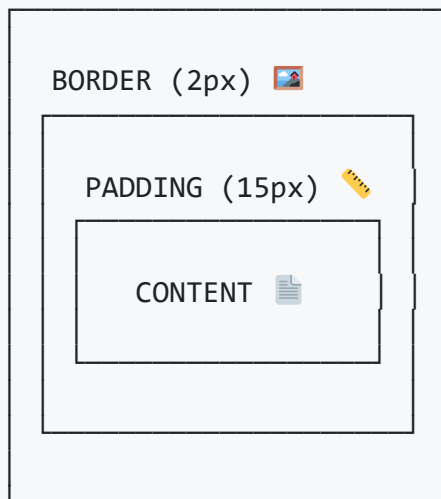
Como la distancia de seguridad entre autos 🚗 ➡️  ➡️ 🚗



# La caja completa

```
div {  
  margin: 20px;      /* Espacio externo */  
  border: 2px solid black; /* El borde */  
  padding: 15px;     /* Espacio interno */  
}
```

MARGIN (20px) - espacio externo 🌌



---

# 📏 Padding y Margin por Lados

### \*\*Forma larga - Cada lado individual\*\*

```css

```
div {  
  padding-top: 25px;  
  padding-right: 50px;
```

# Padding y Margin: Forma Corta

## El truco del reloj

```
div {  
  padding: 25px 50px 75px 100px;  
  /*      ▲      ▲      ▲      ▲  
          arriba derecha abajo izquierda */  
}
```

Recuerda el orden:

 Arriba   Derecha   Abajo   Izquierda

Como las manecillas del reloj en sentido horario 

 Lo mismo aplica para `margin`

# Valores Abreviados: Atajos Útiles

## Menos código, mismo resultado

```
/* 4 valores: arriba, derecha, abajo, izquierda */  
padding: 25px 50px 75px 100px;  
  
/* 3 valores: arriba, lados, abajo */  
padding: 25px 50px 75px; /* derecha e izquierda = 50px */  
  
/* 2 valores: arriba-abajo, lados */  
padding: 25px 50px; /* arriba y abajo = 25px, lados = 50px */  
  
/* 1 valor: todos los lados */  
padding: 25px; /* los 4 lados = 25px */
```

¡Memoriza estos atajos! 💪

# ? Pregunta: ¿Cuánto Padding en Cada Lado?

Analiza este código:

```
div {  
  padding: 10px 20px;  
}
```

¿Cuánto padding tiene cada lado?

- A) Todos los lados: 10px
- B) Arriba/Abajo: 10px, Izquierda/Derecha: 20px
- C) Arriba: 10px, Derecha: 20px, Abajo y Izquierda: 0

Piensa... 🤔



## Respuesta

```
div {  
  padding: 10px 20px;  
}
```

Respuesta correcta: B) Arriba/Abajo: 10px, Izquierda/Derecha: 20px

¿Por qué?

- Con 2 valores: primero es arriba-abajo, segundo es lados
- padding-top: 10px
- padding-bottom: 10px
- padding-left: 20px
- padding-right: 20px

¡Los 2 valores son super útiles! 👍



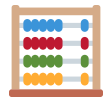
# Calculando el Tamaño Total

¿Cuánto espacio ocupa realmente?

```
div {  
  width: 200px;  
  padding: 10px;  
  border: 2px solid black;  
  margin: 15px;  
}
```

**Fórmula del ancho total:**

```
Ancho Total = margin-left + border-left + padding-left  
              + width  
              + padding-right + border-right + margin-right
```



# Ejemplo de Cálculo

## Paso a paso

```
div {  
  width: 200px;      /* Contenido */  
  padding: 10px;     /* 10px a cada lado */  
  border: 2px;       /* 2px a cada lado */  
  margin: 15px;      /* 15px a cada lado */  
}
```

## Cálculo:

$$\begin{aligned}\text{Ancho Total} &= 15 + 2 + 10 + 200 + 10 + 2 + 15 \\ &= 254\text{px}\end{aligned}$$

¡La caja ocupa 254px en total! 

# ? Pregunta: Calcula el Tamaño

¿Cuánto mide en total?

```
div {  
  width: 100px;  
  padding: 5px;  
  border: 1px solid black;  
  margin: 10px;  
}
```

¿Cuál es el ancho total?

- A) 100px
- B) 116px
- C) 132px

Intenta calcularlo... 🤔



## ✓ Respuesta

```
div {  
  width: 100px;  
  padding: 5px;  
  border: 1px solid black;  
  margin: 10px;  
}
```

Respuesta correcta: C) 132px

Cálculo:

```
margin-left: 10px  
border-left: 1px  
padding-left: 5px  
width: 100px  
padding-right: 5px  
border-right: 1px  
margin-right: 10px
```

---

TOTAL: 132px





# Colapso de Márgenes Verticales

## Un comportamiento curioso de CSS

Cuando dos cajas están una sobre otra, sus márgenes **no se suman**, ¡se toma solo el mayor!

```
.caja1 {  
  margin-bottom: 30px;  
}  
  
.caja2 {  
  margin-top: 20px;  
}
```

¿Cuánta separación habrá?

-  NO son 50px (30 + 20)
-  Sí son 30px (el mayor de los dos)



# ¿Por Qué Colapsan los Márgenes?

## El diseño de CSS

### Razón histórica:

CSS se diseñó así para que los párrafos se vean bien sin doble espacio.

```
<p style="margin-bottom: 20px;">Primer párrafo</p>  
<p style="margin-top: 20px;">Segundo párrafo</p>
```

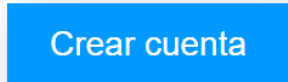
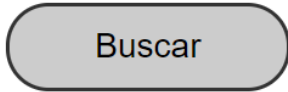
Si se sumaran: 40px de separación ❌ Demasiado espacio

Con colapso: 20px de separación ✅ Perfecto

**Importante:** 💡 Solo ocurre con márgenes **verticales** (top/bottom), no con horizontales (left/right)

# ¡Hora de practicar el modelo de caja!

Recrea estos 3 botones:



## Requisitos:





1. Botón "Search" - Con border, padding y esquinas redondeadas
2. Botón "+" - Circular usando `border-radius: 50%` y padding
3. Botón "Crear Cuenta" - Con sombra usando `box-shadow`

## Pistas:

- Usa `padding` generoso para botones cómodos (10px-15px)
- Para sombras: `box-shadow: 2px 2px 5px rgba(0, 0, 0, 0.3);`

## Recursos Adicionales

### Referencias útiles:

-  [W3Schools CSS](#)
-  [CSS Diner](#) - Juego de selectores
-  [Colores HTML](#)
-  [Unidades CSS](#)

## Resumen de CSS



# Conceptos Clave

## Lo que aprendimos hoy:

- ✓ **CSS** - El diseño y estilo de la web
- ✓ **Selectores** - Cómo elegir elementos (etiqueta, clase, ID)
- ✓ **Especificidad** - Qué estilo gana cuando hay conflictos
- ✓ **Colores** - Nombres, RGB y hexadecimal
- ✓ **Texto** - Alinear, decorar, peso y fuentes
- ✓ **Tamaños** - width, height, font-size, border
- ✓ **Modelo de caja** - margin, padding, border
- ✓ **Bordes** - border, border-radius para esquinas

# Lo Más Importante


## Recuerda:

 CSS da vida a tu HTML

 Selectores son precisos - elemento, clase, ID

 Especificidad importa - ID > Clase > Elemento

 Todo es una caja - content, padding, border, margin

 Colores tienen opciones - nombres, rgb(), #hexadecimal






 Las fuentes marcan la diferencia - font-family, font-weight

 La práctica hace al maestro



# Recursos para Seguir Aprendiendo

## Documentación y herramientas:

-  [MDN Web Docs - CSS](#) - La mejor documentación
-  [W3Schools CSS](#) - Tutoriales y ejemplos
-  [CSS Diner](#) - Juego de selectores
-  [Color Hunt](#) - Paletas de colores
-  [CSS-Tricks](#) - Trucos y guías

# Próximos Pasos

## ¿Qué sigue?

Ahora que dominas selectores y el modelo de caja, el siguiente paso es:

### **Layouts con Flexbox** - Diseños flexibles y responsivos

Aprenderás a:

- Crear diseños de columnas
- Alinear elementos horizontal y verticalmente
- Hacer diseños que se adapten a diferentes pantallas
- Centrar elementos fácilmente
- ¡Crear layouts profesionales!

## Refuerza lo aprendido

### Proyecto sugerido:

Crea una página "Sobre Mí" ( `sobre-mi.html` ) con:

- 🎨 Encabezado con tu nombre y fondo de color
- 👤 Foto o avatar con `border-radius`
- 📝 Sección "Sobre mí" con texto estilizado
- 📁 Lista de habilidades con diferentes colores
- 🎯 Botones de redes sociales personalizados
- 📦 Usa el modelo de caja en todos los elementos

### Aplica:

- Al menos 3 tipos diferentes de selectores
- 3 métodos de color (nombre, RGB, hex)

# Consejos Finales

## Buenas prácticas de CSS

💡 Organiza tu código - Comenta secciones importantes

💡 Usa nombres descriptivos - `.btn-primary` mejor que `.azul`

💡 Prefiere clases - Más reutilizables que IDs

💡 Empieza simple - No compliques desde el inicio

💡 Inspecciona sitios - F12 es tu mejor maestro

💡 Practica diario - 30 minutos hacen la diferencia

💡 No te frustres - Todos empezamos aquí ❤️

 ¡Felicidades!

Ya sabes CSS

Ahora puedes dar estilo a cualquier página web 

# ? Preguntas

¿Alguna duda sobre CSS?

 ¡Excelente Trabajo!

¡Nos vemos en la siguiente clase con Flexbox!

No olvides completar todos los ejercicios 