

Maestría en Ingeniería Electrónica con Énfasis en Tecnologías de la Información

ALUMNO

Ing. César Arguello

CURSO

Ingeniería de Software

DOCENTE

Dr. Julio César Mello Román

TEMA DEL TRABAJO

Explorando CSS

San Lorenzo, Paraguay

2024

Contenido

[1. Técnicas para combinar selectores de CSS y la utilización de herencias. 3](#_Toc193026876)

[1.1. Combinación de selectores 3](#_Toc193026877)

[1.2. Utilización de Herencias en CSS 3](#_Toc193026878)

[2. Prevalencias entre selectores CSS 4](#_Toc193026879)

[2.1. Especificidad 4](#_Toc193026880)

[2.2. Cálculo de la especificidad 5](#_Toc193026881)

[2.3. Precedencia de selectores 5](#_Toc193026882)

[2.4. Orden de prevalencia 5](#_Toc193026883)

[2.5. Reglas de comportamiento 5](#_Toc193026884)

[3. Framework de CSS 5](#_Toc193026885)

[3.1. Características principales de los frameworks de CSS: 6](#_Toc193026886)

[3.2. Ejemplos populares de frameworks de CSS: 6](#_Toc193026887)

[Referencias 7](#_Toc193026888)

# Técnicas para combinar selectores de CSS y la utilización de herencias.

En CSS, la combinación de selectores y la utilización de la herencia son técnicas fundamentales para escribir hojas de estilo eficientes y mantener un diseño coherente:

## 1.1. Combinación de selectores

Existen varias formas de combinar selectores en CSS para ser más específicos o aplicar estilos a elementos relacionados entre sí. Algunas de las combinaciones más comunes son:

a. Selectores de descendientes (espacio)

Un selector de descendientes selecciona todos los elementos que son descendientes de un elemento específico.

b. Selectores de hijos directos (>)

Un selector de hijos directos selecciona solo los elementos que son hijos directos de un elemento específico.

c. Selectores de hermanos adyacentes (+)

Un selector de hermanos adyacentes selecciona un elemento que es inmediatamente precedido por otro.

d. Selectores de hermanos generales (~)

Un selector de hermanos generales selecciona todos los elementos que son hermanos (es decir, que comparten el mismo elemento padre) y que siguen a un elemento específico.

e. Selectores múltiples (,)

Puedes agrupar varios selectores para aplicar los mismos estilos a varios elementos.

## 1.2. Utilización de Herencias en CSS

La herencia en CSS se refiere a la capacidad de un elemento para heredar ciertas propiedades de su elemento padre. Por ejemplo, propiedades como color, font-family, font-size, y line-height suelen ser heredadas por los elementos hijos.

a. Propiedades heredadas por defecto

Algunas propiedades se heredan automáticamente, como:

color

font-family

font-size

line-height

text-align

b. Evitar la herencia (uso de inherit, initial, y unset)

inherit: Fuerza la herencia de una propiedad.

initial: Establece la propiedad con el valor inicial (el valor por defecto de la propiedad).

unset: Restaura el comportamiento predeterminado, es decir, hereda si es heredable, o establece el valor inicial si no lo es.

c. Herencia controlada con all

La propiedad all permite restablecer o aplicar todos los valores heredados o no heredados de un solo golpe.

# Prevalencias entre selectores CSS

En CSS, la prevalencia (o especificidad) de los selectores determina qué estilo se aplicará a un elemento cuando varios selectores coincidan con el mismo elemento. La especificidad se calcula con base en los tipos de selectores utilizados, y si hay conflictos entre reglas, se elige la que tiene mayor especificidad.

## Especificidad

La especificidad es un sistema que CSS utiliza para decidir qué regla de estilo aplicar cuando múltiples reglas coinciden con el mismo elemento. Se calcula en función de los diferentes tipos de selectores empleados en una regla.

Fórmula de especificidad

La especificidad se calcula con un sistema basado en 4 partes:

A: Cuenta de selectores de ID (ej. #id)

B: Cuenta de selectores de clases, pseudoclases y atributos (ej. clase, :hover, [type="text"])

C: Cuenta de selectores de elementos (ej. div, p, h1)

D: Selectores !important (que tienen la mayor prioridad, pero no afectan al cálculo de la especificidad de la regla, solo si hay un conflicto entre reglas).

## Cálculo de la especificidad

La especificidad se calcula con base en el número de selectores de cada tipo en una regla

Si una propiedad CSS tiene la declaración !important, tiene mayor prioridad que las reglas con una especificidad más baja, independientemente de la especificidad de los selectores. Sin embargo, !important no afecta al cálculo de especificidad, solo a la prioridad de la regla.

## Precedencia de selectores

Cuando dos o más reglas tienen la misma especificidad, la última regla en el archivo CSS tendrá prioridad.

## Orden de prevalencia

La prevalencia o la prioridad de los selectores sigue este orden (de menor a mayor especificidad):

Selectores de elementos: Seleccionan todas las instancias de un tipo de elemento.

Selectores de clases, atributos y pseudoclases: Seleccionan elementos según una clase, atributo o pseudoclase.

Selectores de ID: Seleccionan elementos según un identificador único.

!important: Si se usa en una regla, da prioridad sobre reglas con la misma especificidad.

Combinaciones: Las combinaciones de selectores pueden aumentar la especificidad.

## Reglas de comportamiento

Si dos reglas tienen la misma especificidad, se aplica la última regla definida en el código CSS (esto se llama "cascada").

Los selectores más específicos tienen prioridad, por lo que siempre que se use un selector con mayor especificidad (como un ID frente a una clase o un elemento), su estilo será el que se aplique.

# Framework de CSS

Un framework de CSS es una colección de herramientas predefinidas (clases, reglas, funciones y componentes) que ayudan a los desarrolladores web a crear sitios web y aplicaciones de manera más rápida y eficiente, proporcionando una base consistente y estructurada para el diseño de interfaces. Estos frameworks se utilizan para evitar escribir código CSS desde cero, optimizando el desarrollo y asegurando una mayor coherencia visual.

## Características principales de los frameworks de CSS:

Estructura de diseño predefinida: Muchos frameworks ofrecen un sistema de rejilla o grid system, que permite crear diseños responsivos sin tener que escribir mucho código. Estos sistemas dividen la página en columnas, facilitando el diseño flexible y adaptativo a diferentes tamaños de pantalla.

Componentes listos para usar: Los frameworks incluyen una serie de componentes como botones, formularios, menús, tarjetas, modales y otros elementos de UI listos para usar, lo que ahorra tiempo en la creación de estos elementos básicos.

Estilos consistentes: Proporcionan una apariencia visual coherente que ayuda a mantener el diseño uniforme a lo largo del proyecto, evitando la necesidad de decidir colores, fuentes y otros aspectos de estilo repetidamente.

Personalización: La mayoría de los frameworks permiten cierto grado de personalización a través de variables, opciones de configuración y sobrecarga de estilos, lo que permite adaptar el diseño al branding o las necesidades específicas del proyecto.

Soporte para diseño responsivo: Los frameworks suelen estar diseñados para ser mobile-first, garantizando que los sitios sean compatibles con dispositivos móviles y de escritorio sin necesidad de realizar ajustes manuales para diferentes tamaños de pantalla.

## Ejemplos populares de frameworks de CSS:

Bootstrap:

Uno de los frameworks más populares, creado por Twitter.

Incluye un sistema de rejilla flexible, componentes de interfaz, y es ampliamente utilizado para desarrollo de sitios web responsivos.

Facilita la creación de interfaces atractivas con muy poco esfuerzo.

Tailwind CSS:

Un framework de CSS basado en utilidades que permite a los desarrolladores escribir estilos directamente en el HTML mediante clases.

Ofrece un enfoque de "estilos atómicos", lo que significa que puedes combinar clases pequeñas y específicas para crear componentes sin necesidad de escribir CSS adicional.

Foundation:

Similar a Bootstrap, es un framework robusto que también incluye un sistema de rejilla y componentes listos para usar.

Proporciona mayor flexibilidad y personalización para diseñadores y desarrolladores avanzados.

Bulma:

Un framework CSS moderno basado en flexbox que proporciona una estructura sencilla pero poderosa para crear interfaces responsivas y limpias.

Es muy fácil de usar y más ligero que otros frameworks como Bootstrap.

Materialize:

Basado en Material Design de Google, este framework proporciona una serie de componentes y estilos que siguen las guías de diseño de Google.

Ideal para quienes quieren aplicar el diseño visual de Material Design en sus sitios web rápidamente.

Ventajas de usar un framework de CSS:

Desarrollo más rápido: Los frameworks proporcionan una base de código sólida que elimina la necesidad de escribir CSS desde cero, lo que acelera el proceso de desarrollo.

Consistencia: Utilizando un framework, se garantiza que los elementos y la interfaz tengan un diseño uniforme y bien estructurado.

Compatibilidad entre navegadores: Muchos frameworks están optimizados para funcionar bien en todos los navegadores modernos, lo que reduce problemas de compatibilidad.

Soporte y comunidad: Los frameworks populares como Bootstrap tienen una gran comunidad de desarrolladores que pueden ofrecer soporte, tutoriales y actualizaciones.

Desventajas de usar un framework de CSS:

Peso extra: Los frameworks pueden incluir código innecesario para tu proyecto específico, lo que puede aumentar el tamaño de los archivos CSS y afectar el rendimiento de la página.

Falta de personalización total: Aunque muchos frameworks permiten personalización, pueden limitar la flexibilidad en ciertos aspectos del diseño si se requiere algo completamente único.

Dependencia del framework: El uso de un framework puede hacer que el proyecto dependa de una tecnología específica, lo que puede dificultar la transición a otro framework o solución en el futuro.

# Referencias

MDN Web Docs - Selectores CSS

MDN Web Docs - Herencia en CSS

W3Schools - CSS Selectors

CSS-Tricks – Specificity

*CSS: The Definitive Guide* de Eric A. Meyer

*HTML and CSS: Design and Build Websites* de Jon Duckett