14 CSS: La Cascada (The Cascade)

La Cascada es el algoritmo fundamental que usan los navegadores para resolver los conflictos cuando múltiples reglas CSS intentan aplicar diferentes valores a la misma propiedad de un mismo elemento. Es el corazón del sistema que determina qué estilo "gana" y se aplica finalmente.

El nombre "Cascada" sugiere un flujo descendente, donde los estilos de diferentes orígenes y niveles de importancia interactúan y potencialmente se sobrescriben unos a otros hasta que queda un valor final para cada propiedad.

El proceso de la cascada sigue un orden estricto de criterios para decidir qué declaración de estilo tiene prioridad:

Origen de la Hoja de Estilos e Importancia (`!important`)

Primero, la cascada ordena las reglas según su origen y si están marcadas como `!important`. Este es el factor más decisivo. El orden de prioridad, de **mayor a menor**, es el siguiente:

1. **Transiciones y Animaciones CSS:** Los estilos aplicados activamente por transiciones o animaciones CSS tienen una prioridad muy alta mientras están en ejecución, sobrescribiendo otros valores (excepto `!important`).
2. **Reglas del Usuario con `!important`:** Estilos definidos por el usuario final en su navegador (si lo permite y configura) y marcados con `!important`. Esto permite al usuario forzar sus preferencias de accesibilidad o visualización sobre el diseño del sitio.
3. **Reglas del Autor con `!important`:** Estilos definidos por el desarrollador web (en los archivos `.css` del sitio) y marcados con `!important`. Se usa para asegurar que una regla específica *siempre* se aplique, sobrescribiendo otras reglas del autor, del usuario (normales) y del navegador. **Debe usarse con moderación**, ya que puede dificultar la sobrescritura y el mantenimiento posteriores.
4. **Reglas del Autor (Normales):** Los estilos normales definidos por el desarrollador web en sus hojas de estilo. Esta es la categoría donde residen la mayoría de tus reglas CSS.
5. **Reglas del Usuario (Normales):** Estilos normales definidos por el usuario final en su navegador.
6. **Reglas del Navegador (User Agent Stylesheets):** Los estilos predeterminados que cada navegador aplica a los elementos HTML antes de que cualquier otro CSS entre en juego (por ejemplo, los márgenes por defecto de los párrafos, el tamaño de los encabezados, el subrayado de los enlaces). Tienen la menor prioridad.

•

Punto Clave sobre `!important`: Observa cómo `!important` invierte el orden de prioridad habitual entre el Autor y el Usuario. Normalmente, los estilos del autor (tu CSS) tienen más peso que los del usuario. Pero si ambos usan `!important`, la regla del usuario `!important` gana, garantizando que el usuario tenga la última palabra para forzar sus necesidades.

Especificidad del Selector

Si dos o más reglas compiten *dentro de la misma capa de origen e importancia* (por ejemplo, dos reglas normales del autor o dos reglas del autor con `!important`), la cascada recurre al siguiente

criterio: la **Especificidad** del selector.

- **Concepto:** La especificidad es una "puntuación" o "peso" que el navegador calcula para cada selector. Cuanto más específico sea un selector (por ejemplo, un ID es más específico que una clase, que a su vez es más específica que una etiqueta), mayor será su puntuación.
- **Regla:** La regla con el **selector de mayor especificidad** gana el conflicto.
- *Nota:* Detallaremos cómo se calcula exactamente la especificidad en la siguiente sección. Por ahora, solo necesitamos saber que es el segundo criterio en la cascada.

3. Orden en el Código Fuente (Source Order)

Si dos reglas tienen exactamente el mismo **origen**, la misma **importancia** (`!important` o no) y la misma **especificidad**, la cascada utiliza el criterio final de desempate: el **orden en el código fuente**.

- **Regla:** La regla que aparece **más tarde** en las hojas de estilo (o la última en el mismo archivo CSS) es la que gana. Si los estilos están en diferentes archivos enlazados en el HTML, el archivo enlazado más tarde tiene prioridad (asumiendo misma especificidad, etc.).

  ```
  /* archivo estilos.css */
  p {
    color: blue; /* Regla 1 */
  }

  /* ...mucho código después... */

  p {
    color: green; /* Regla 2 */
  }
  ```

  En este caso, aunque ambas reglas tienen la misma especificidad (selector de etiqueta), la segunda regla (`color: green;`) gana porque aparece después en el código. Todos los párrafos serán verdes (a menos que otra regla más específica o con `!important` diga lo contrario).

Resumen del Proceso de Cascada (por Propiedad):

Para cada elemento y cada propiedad CSS (como `color`, `font-size`, `margin-left`):

1. Recopila todas las declaraciones que intentan establecer esa propiedad para ese elemento.
2. Filtra y ordena esas declaraciones según **Origen e Importancia**. Quédate con las de mayor prioridad.
3. Si quedan varias declaraciones empatadas, calcula la **Especificidad** de sus selectores. Quédate con la(s) de mayor especificidad.
4. Si todavía hay empate, elige la declaración que aparece **última en el Orden del Código Fuente**.

¡Esa declaración "ganadora" es la que se aplica al elemento para esa propiedad!

¿Por qué es Importante Entender la Cascada?

- **Predecir Estilos:** Te permite entender por qué tus elementos se ven como se ven.
- **Depurar CSS:** Es la clave para diagnosticar por qué un estilo no se aplica como esperas (probablemente está siendo sobrescrito por una regla con mayor prioridad en la cascada).

- **Escribir CSS Eficiente:** Te ayuda a estructurar tu CSS y usar selectores de manera que evites conflictos innecesarios o el abuso de `!important`.
- **Trabajar con Frameworks:** Los frameworks CSS (como Bootstrap, Tailwind) dependen fuertemente de la cascada y la especificidad. Entenderla es vital para personalizarlos correctamente.

Ahora que comprendemos el flujo general de la cascada y cómo prioriza las reglas, estamos listos para sumergirnos en el segundo criterio clave: la **Especificidad**.