16 CSS: Practicando Cascada y Especificidad

La teoría de la Cascada y la Especificidad es crucial, pero ver cómo interactúan en ejemplos concretos es la clave para dominarlas. Analicemos algunos escenarios y luego pongamos a prueba tus conocimientos.

Recordatorio Rápido:

- **Cascada:** Resuelve conflictos basándose en: 1. Origen e Importancia (`!important`), 2. Especificidad, 3. Orden en el Código.
- **Especificidad:** Puntuación (A, B, C) donde A=IDs, B=Clases/Atributos/Pseudoclases, C=Elementos/Pseudoelementos. Se compara A, luego B, luego C.

Ejemplo 1: Conflicto Básico de Especificidad

- **HTML:**

```
<button id="boton-enviar" class="btn btn-primario">Enviar Datos</button>
```

- **CSS:**

```
/* Regla 1 */
button {
  background-color: lightgrey; /* Especificidad: (0, 0, 1) */
  padding: 10px;
}

/* Regla 2 */
.btn {
  background-color: blue;      /* Especificidad: (0, 1, 0) */
  color: white;
}

/* Regla 3 */
#boton-enviar {
  background-color: green;     /* Especificidad: (1, 0, 0) */
  font-weight: bold;
}
```

- **Pregunta:** ¿De qué color será el fondo (`background-color`) del botón? ¿Qué otros estilos se aplicarán?
- **Análisis:**
  1. **Origen e Importancia:** Todas las reglas son del autor y ninguna usa `!important`. Pasamos al siguiente criterio.
  2. **Especificidad (para `background-color`):**

    - Regla 1 (`button`): `(0, 0, 1)`
    - Regla 2 (`.btn`): `(0, 1, 0)`
    - Regla 3 (`#boton-enviar`): `(1, 0, 0)`
  3. **Comparación:** Comparamos la columna A (IDs). `1` es mayor que `0`. La Regla 3 tiene la mayor especificidad.
  4. **Resultado (`background-color`):** El fondo será **verde** (definido por `#boton-enviar`).
  5. **Otros Estilos:**
    - `padding: 10px;` (de `button`): Se aplica, ya que no hay conflicto con otras reglas más específicas para `padding`.
    - `color: white;` (de `.btn`): Se aplica, ya que no hay conflicto.
    - `font-weight: bold;` (de `#boton-enviar`): Se aplica.
- **Conclusión:** El botón tendrá fondo verde, texto blanco, padding de 10px y fuente en negrita.

Ejemplo 2: Empate de Especificidad y Orden en el Código

- **HTML:**

```
<p class="info">Este es un párrafo informativo.</p>
```

- **CSS:**

```
/* archivo: base.css */
.info {
  color: navy;      /* Especificidad: (0, 1, 0) */
  margin-bottom: 10px;
}

/* archivo: components.css (enlazado DESPUÉS de base.css en el HTML) */
.info {
  color: darkblue;  /* Especificidad: (0, 1, 0) */
  font-style: italic;
}
```

- **Pregunta:** ¿De qué color será el texto del párrafo y qué otros estilos tendrá?
- **Análisis:**
  1. **Origen e Importancia:** Ambas reglas son del autor, sin `!important`.
  2. **Especificidad (para `color`):** Ambas reglas tienen el mismo selector `.info`, por lo tanto, tienen la misma especificidad: `(0, 1, 0)`. Hay un empate.
  3. **Orden en el Código:** Como `components.css` se enlaza después de `base.css` (o si estuvieran en el mismo archivo, la segunda regla aparece después), la regla de `components.css` gana el desempate.
  4. **Resultado (`color`):** El color del texto será **darkblue**.
  5. **Otros Estilos:**

    - `margin-bottom: 10px;` (de `base.css`): Se aplica, no hay conflicto.
    - `font-style: italic;` (de `components.css`): Se aplica.
- **Conclusión:** El párrafo será `darkblue`, itálico y tendrá un margen inferior de 10px.

Ejemplo 3: Estilos en Línea vs. Hoja de Estilos

- **HTML:**

```
<h1 id="titulo" style="color: purple; text-decoration: underline;">Título Principal</h1>
```

- **CSS:**

```
#titulo {
  color: teal;           /* Especificidad: (1, 0, 0) */
  text-decoration: none;   /* Especificidad: (1, 0, 0) */
  font-size: 3em;
}
```

- **Pregunta:** ¿De qué color será el H1 y tendrá subrayado?
- **Análisis:**
    1. **Origen e Importancia:** Tenemos estilos del autor en la hoja de estilos y estilos en línea (que también son del autor pero tienen una prioridad especial en la cascada). No hay `!important`.
    2. **Precedencia:** Los estilos en línea (`style="..."`) tienen mayor precedencia que cualquier selector en las hojas de estilo (IDs, clases, etc.), *excepto* si se usa `!important` en la hoja de estilos.
    3. **Resultado:**
        - `color`: Gana el estilo en línea. Será **purple**.
        - `text-decoration`: Gana el estilo en línea. Será **underline**.
        - `font-size`: No hay conflicto, se aplica el `3em` de la hoja de estilos.
- **Conclusión:** El H1 será púrpura, subrayado y con tamaño de fuente 3em.

Ejemplo 4: El Poder de `!important`

- **HTML:**

```
<p id="aviso" class="mensaje error">¡Alerta Importante!</p>
```

- **CSS:**

```
/* Regla 1 */
#aviso {
  background-color: yellow;  /* Especificidad: (1, 0, 0) */
  padding: 15px;
}

/* Regla 2 */
```

```css
.error {
  background-color: red !important; /* Especificidad: (0, 1, 0), pero con !important */
  color: white;
}
```

- **Pregunta:** ¿Cuál será el color de fondo del párrafo?
- **Análisis:**
  1. **Origen e Importancia:** Ambas reglas son del autor. Sin embargo, la Regla 2 usa `!important`.
  2. **Cascada:** Las reglas del autor con `!important` tienen mayor prioridad que las reglas normales del autor, **independientemente de la especificidad**.
  3. **Resultado (`background-color`):** La Regla 2 (`background-color: red !important;`) gana debido al `!important`, aunque su selector (`.error`) sea menos específico que `#aviso`. El fondo será **rojo**.
  4. **Otros Estilos:**
     - `padding: 15px;` (de `#aviso`): Se aplica, no hay conflicto.
     - `color: white;` (de `.error`): Se aplica.
- **Conclusión:** El párrafo tendrá fondo rojo, texto blanco y padding de 15px. **¡Usa `!important` con moderación!** Hace que el CSS sea más difícil de sobrescribir y depurar.

Ejercicio Práctico: Predice el Resultado

Considera el siguiente HTML y CSS. Determina el valor final para `color`, `font-weight` y `text-decoration` del enlace (`<a>`).

- **HTML:**

```html
<body id="pagina-principal">
  <div class="contenido">
    <p>
      Visita nuestro <a href="#" class="enlace destacado" id="enlace-oferta">sitio de ofertas</a>.
    </p>
  </div>
</body>
```

- **CSS:**

```css
/* Hoja 1: reset.css */
* {
  text-decoration: none; /* Especificidad: (0,0,0) */
}

/* Hoja 2: styles.css (enlazada después de reset.css) */
body {
  color: #333; /* Heredable */
}
```

```css
a {
  color: blue; /* Especificidad: (0,0,1) */
  font-weight: normal; /* Especificidad: (0,0,1) */
}

.contenido a {
  color: navy; /* Especificidad: (0,1,1) */
  text-decoration: underline; /* Especificidad: (0,1,1) */
}

a.destacado {
  font-weight: bold; /* Especificidad: (0,1,1) */
}

#pagina-principal .contenido a#enlace-oferta {
  color: green; /* Especificidad: (2,1,1) = #pagina-principal, #enlace-oferta(A=2), .contenido (B=1), a(C=1) */
}

a:hover {
  color: red !important; /* Especificidad: (0,1,1) + !important */
  text-decoration: none; /* Especificidad: (0,1,1) */
}
```

- **Tu Turno:** ¿Qué valores tendrán `color`, `font-weight` y `text-decoration` para el enlace **en su estado normal (sin hover)**? (Piensa en la especificidad y el orden).

(Pausa para pensar...)

Solución del Ejercicio:

1. `color`:
   - Reglas que aplican `color`: `body` (heredado), `a`, `.contenido a`, `#pagina-principal .contenido a#enlace-oferta`, `a:hover`.
   - Ignoramos `body` (herencia se aplica solo si no hay valor directo) y `a:hover` (es otro estado).
   - Comparamos especificidades:
     - `a`: `(0, 0, 1)`
     - `.contenido a`: `(0, 1, 1)`
     - `#pagina-principal .contenido a#enlace-oferta`: `(2, 1, 1)`
   - Gana `#pagina-principal .contenido a#enlace-oferta`.
   - `color` **final:** `green`
2. `font-weight`:
   - Reglas que aplican `font-weight`: `a`, `a.destacado`.
   - Comparamos especificidades:
     - `a`: `(0, 0, 1)`

    - `a.destacado`: `(0, 1, 1)` (un elemento `a`, una clase `.destacado`)
  - Gana `a.destacado`.
  - `font-weight` **final:** `bold`
3. `text-decoration`:
  - Reglas que aplican `text-decoration`: , `.contenido a`, `a:hover`.
  - Ignoramos `a:hover`.
  - Comparamos especificidades:
    - : `(0, 0, 0)`
    - `.contenido a`: `(0, 1, 1)`
  - Gana `.contenido a`.
  - `text-decoration` **final:** `underline`

Conclusión Final del Ejercicio: En estado normal, el enlace tendrá `color: green`, `font-weight: bold` y `text-decoration: underline`. (Nota: Si el usuario hiciera hover, el `color` cambiaría a `red` debido al `!important`, y el `text-decoration` cambiaría a `none` porque la especificidad de `a:hover` (0,1,1) es igual a la de `.contenido a` (0,1,1) pero la regla `:hover` aparece después en el código).
