17 CSS: Herencia (Inheritance)

La Herencia es otro mecanismo clave en CSS que determina cómo se aplican los estilos a los elementos. A diferencia de la Cascada y la Especificidad, que se centran en resolver conflictos cuando múltiples reglas *directamente* apuntan a un elemento, la Herencia describe cómo los valores de ciertas propiedades CSS aplicadas a un elemento **padre** pueden ser **transmitidos automáticamente a sus elementos hijos**.

¿Cómo Funciona?

Cuando el navegador calcula los estilos finales para un elemento, después de haber aplicado la Cascada (considerando origen, importancia, especificidad y orden) para cada propiedad, puede ocurrir que una propiedad específica **no tenga ningún valor asignado directamente** a ese elemento.

Si esa propiedad es **heredable**, el elemento buscará el valor *calculado* de esa misma propiedad en su elemento padre directo y lo adoptará (lo "heredará"). Este proceso continúa hacia arriba en el árbol DOM si es necesario, hasta que se encuentre un valor o se llegue al elemento raíz (`<html>`).

Importante: La Herencia solo entra en juego si **no se ha definido un valor directo** para la propiedad en el elemento a través de la Cascada. Una regla CSS que se aplica directamente a un elemento (incluso una con baja especificidad como un selector de tipo) siempre tendrá prioridad sobre un valor heredado para esa misma propiedad.

¿Qué Propiedades se Heredan y Cuáles No?

No todas las propiedades CSS son heredables por defecto. La decisión de si una propiedad es heredable o no se basa generalmente en el sentido común y la utilidad:

- **Propiedades Típicamente Heredables:**
  - Propiedades relacionadas con el **texto:** `color`, `font-family`, `font-size`, `font-style`, `font-weight`, `letter-spacing`, `line-height`, `text-align`, `text-indent`, `text-transform`, `white-space`, `word-spacing`.
  - Propiedades de **listas:** `list-style`, `list-style-type`, `list-style-position`, `list-style-image`.
  - **Visibilidad:** `visibility`.
  - **Cursor:** `cursor`.
  - Otras: `border-collapse`, `border-spacing` (para tablas), `orphans`, `widows`, etc.
  - *Lógica:* Tiene sentido que el texto dentro de un `<div>` herede el color y la fuente del `<div>` por defecto, para mantener la consistencia.
- **Propiedades Típicamente NO Heredables:**
  - La mayoría de las propiedades del **Modelo de Caja:** `margin`, `padding`, `border` (y sus sub-propiedades), `width`, `height`, `min-width`, `min-height`, `max-width`, `max-height`.
  - Propiedades de **fondo:** `background`, `background-color`, `background-image`, etc.
  - Propiedades de **posicionamiento:** `position`, `top`, `right`, `bottom`, `left`, `z-index`.
  - Propiedades de **display y layout:** `display`, `float`, `clear`, `overflow`, `flex`, `grid`, etc.
  - Otras: `outline`, `box-shadow`, `text-decoration` (aunque afecta al texto, no se hereda directamente, sino que se aplica al elemento).

- *Lógica:* No querrías que cada `<span>` dentro de un `<div>` con borde herede automáticamente ese mismo borde, o que cada párrafo herede el `padding` de su contenedor.

- **¿Cómo Saberlo con Certeza?** La documentación de cada propiedad CSS (por ejemplo, en MDN Web Docs) indica explícitamente si es heredable ("Inherited: yes") o no ("Inherited: no").

Ejemplo de Herencia en Acción:

```
<div style="color: blue; border: 1px solid red; padding: 10px;">
  <p>Este párrafo hereda el <span>color azul</span> del div.</p>
  <p>El borde y el padding del div NO se heredan.</p>
</div>
```

- El texto de los párrafos `<p>` y del `<span>` será azul porque la propiedad `color` es heredable y no se definió un color diferente directamente en ellos.
- Los párrafos y el span no tendrán borde ni padding, porque esas propiedades no son heredables por defecto.

Controlando la Herencia: Palabras Clave Especiales

CSS proporciona palabras clave específicas que permiten controlar explícitamente la herencia y los valores por defecto:

1. `inherit`
   - **Qué hace:** Fuerza a un elemento a heredar el valor *calculado* de una propiedad de su elemento padre, **incluso si la propiedad no es heredable por defecto**.
   - **Uso:** Útil cuando quieres que un elemento hijo coincida explícitamente con una propiedad de su padre que normalmente no heredaría.
   - **Ejemplo:** Hacer que el color de borde de un input coincida con el color de texto de su contenedor.

     ```
     .contenedor {
       color: green;
     }
     .contenedor input {
       border: 2px solid inherit; /* Hereda 'green' del contenedor para el borde */
       color: inherit; /* Hereda 'green' para el texto del input */
     }
     ```

2. `initial`
   - **Qué hace:** Restablece una propiedad en un elemento a su **valor inicial**, tal como se define en la especificación CSS para esa propiedad. Este valor inicial es universal y no depende del tipo de elemento ni de los estilos del navegador.
   - **Uso:** Para deshacer cualquier estilo aplicado (directo o heredado) y volver al valor base absoluto definido por el estándar CSS.
   - **Ejemplo:**

```css
p {
  color: red;
}
.resetear-color {
  color: initial; /* Vuelve al valor inicial de 'color' (suele ser negro) */
}
```

3. `unset`
   - **Qué hace:** Actúa de forma diferente según si la propiedad es heredable o no:
     - Si la propiedad **es heredable** por defecto (como `color`), `unset` actúa como `inherit`.
     - Si la propiedad **no es heredable** por defecto (como `border` o `background-color`), `unset` actúa como `initial`.
   - **Uso:** Muy útil en reseteos de CSS para establecer un comportamiento predecible: heredar lo que tiene sentido heredar, y volver al valor inicial para lo demás.
   - **Ejemplo:**

```css
/* Dentro de un reseteo */
div {
  color: unset; /* Actúa como inherit → hereda color del padre */
  background-color: unset; /* Actúa como initial → fondo transparente */
  border: unset; /* Actúa como initial → sin borde */
}
```

4. `revert` (Más avanzado)
   - **Qué hace:** Revierte el estilo de la propiedad al valor establecido por la capa anterior en el origen de la cascada (Navegador → Usuario → Autor). Si se usa en los estilos del autor, revertirá al estilo del usuario (si existe) o, más comúnmente, al estilo predeterminado del navegador (User Agent).
   - **Uso:** Para deshacer específicamente los estilos del autor y volver a los predeterminados del navegador o del usuario, sin usar `initial` (que podría ser diferente del predeterminado del navegador para un elemento específico).