28 CSS: Tamaño de Caja (`box-sizing`)

Hemos visto que el modelo de caja estándar de CSS se compone de contenido, padding, borde y margen. Por defecto, cuando estableces las propiedades `width` y `height` de un elemento, estas dimensiones se aplican **únicamente al área de contenido** del elemento. El padding y el borde se *añaden* a ese ancho y alto para determinar el tamaño total que ocupa el elemento en la pantalla. Esto puede llevar a cálculos complicados y a resultados inesperados en el diseño.

El Problema del Modelo de Caja por Defecto (`content-box`)

Imagina que quieres dos cajas, cada una ocupando exactamente el 50% del ancho de su contenedor, con un borde y algo de padding.

- **HTML:**

```
<div class="contenedor">
  <div class="caja caja1">Caja 1</div>
  <div class="caja caja2">Caja 2</div>
</div>
```

- **CSS (con el modelo por defecto):**

```
.contenedor {
  width: 400px;
  border: 1px solid grey;
}
.caja {
  width: 50%; /* Esperarías 200px */
  padding: 10px;
  border: 5px solid blue;
  /* box-sizing: content-box;  ← Este es el valor por defecto */
  float: left; /* Para ponerlas lado a lado (veremos float más adelante) */
}
```

- **Cálculo del Ancho Real (por defecto):**
  - `width`: 50% de 400px = 200px (área de contenido)
  - `padding`: 10px a la izquierda + 10px a la derecha = 20px
  - `border`: 5px a la izquierda + 5px a la derecha = 10px
  - **Ancho Total Ocupado = 200px + 20px + 10px = 230px**
- **Resultado:** Cada caja intenta ocupar 230px. Como 230px + 230px = 460px, que es más que los 400px del contenedor, la segunda caja no cabrá al lado de la primera y saltará a la línea siguiente. ¡Esto no es lo que queríamos al poner `width: 50%` !

La Solución: `box-sizing: border-box;`

La propiedad `box-sizing` nos permite cambiar cómo se calcula el tamaño total de la caja.

Valores de `box-sizing` :

1. `content-box`:
   - Este es el valor **predeterminado**.
   - `width` y `height` definen las dimensiones del **área de contenido únicamente**.
   - El padding y el borde se suman *fuera* de estas dimensiones.
   - Ancho total = `width` + `padding-left` + `padding-right` + `border-left-width` + `border-right-width`.
   - Alto total = `height` + `padding-top` + `padding-bottom` + `border-top-width` + `border-bottom-width`.
2. `border-box`:
   - `width` y `height` definen las dimensiones **totales de la caja, incluyendo el padding y el borde**.
   - El área de contenido se encoge automáticamente para hacer espacio al padding y al borde dentro del `width` y `height` especificados.
   - Ancho total = `width` (¡y este `width` ya incluye padding y borde horizontal!).
   - Alto total = `height` (¡y este `height` ya incluye padding y borde vertical!).
   - El margen sigue estando fuera de este cálculo, como siempre.

Aplicando `border-box` al Ejemplo Anterior:

- **CSS (con `border-box`):**

```css
.contenedor {
  width: 400px;
  border: 1px solid grey;
  overflow: hidden; /* Para contener los floats */
}
.caja {
  box-sizing: border-box; /* ¡La clave! */
  width: 50%; /* Ahora sí significa 200px de ancho TOTAL */
  padding: 10px;
  border: 5px solid blue;
  float: left;
  background-color: lightblue; /* Para ver el área */
}
```

- **Cálculo del Ancho (con `border-box`):**
  - `width`: 50% de 400px = **200px (Ancho Total Ocupado)**
  - Dentro de estos 200px, el navegador calcula el espacio para el contenido:
    - Padding: 10px (izq) + 10px (der) = 20px
    - Border: 5px (izq) + 5px (der) = 10px
    - Espacio para contenido = 200px - 20px - 10px = 170px
- **Resultado:** Cada caja ocupa exactamente 200px de ancho total. Como 200px + 200px = 400px, las dos cajas caben perfectamente una al lado de la otra dentro del contenedor. ¡Mucho más intuitivo!

Ventajas de `border-box`:

- **Diseño Predecible:** Hace que trabajar con porcentajes y unidades fijas sea mucho más sencillo, ya que el `width` y `height` definen el espacio final que ocupará el elemento.
- **Menos Cálculos:** Evita tener que restar manualmente el padding y el borde al definir anchos o altos.
- **Consistencia:** Facilita la creación de rejillas (grids) y layouts flexibles.

Práctica Recomendada:

Muchos desarrolladores consideran `border-box` un modelo de caja más intuitivo y prefieren usarlo globalmente en sus proyectos. Una técnica muy común es aplicar `border-box` a todos los elementos y pseudoelementos al inicio del archivo CSS:

```
/* Aplicar un modelo de caja más intuitivo a todos los elementos */
*,
*::before,
*::after {
  box-sizing: border-box;
}

/* Opcionalmente, puedes resetear márgenes y paddings también aquí */
/* body, h1, h2, p, ul, li {
  margin: 0;
  padding: 0;
} */
```
