21 CSS: width & height

Las propiedades `width` (ancho) y `height` (alto) se utilizan para establecer las dimensiones del **área de contenido** de la caja de un elemento. Como vimos en la introducción, en el modelo de caja estándar (`box-sizing: content-box`), estas dimensiones *no* incluyen el padding, el borde ni el margen.

`width`

- **Propósito:** Define el ancho del área de contenido de un elemento.
- **Valores Comunes:**
  - `auto` **(Valor por defecto):** El navegador calcula el ancho automáticamente. El comportamiento exacto depende del tipo de elemento (que veremos en detalle más adelante en "Cajas de Línea VS Cajas de Bloque"):
    - Para **elementos de bloque** (como `<div>`, `<p>`, `<h1>`), `auto` generalmente significa que el elemento ocupará todo el ancho disponible de su contenedor padre.
    - Para **elementos en línea** (como `<span>`, `<a>`, `<strong>`), `width` por defecto es `auto` y el ancho se ajusta al contenido que contienen. ¡Importante: La propiedad `width` **no tiene efecto** en elementos en línea que no hayan sido modificados (por ejemplo, con `display: inline-block` o `display: block`)!
    - Para imágenes y otros elementos reemplazados, `auto` suele significar el ancho intrínseco del elemento (el tamaño original de la imagen).
  - `<length>`: Un valor de longitud fija. Las unidades más comunes son:
    - `px` **(píxeles):** Unidad absoluta, fija. Define un ancho exacto.

      ```
      .caja-fija { width: 300px; }
      ```

    - `em`: Relativo al tamaño de fuente del *propio elemento*.
    - `rem`: Relativo al tamaño de fuente del *elemento raíz* (`<html>`). (Menos común para `width` que `px` o `%`).
  - `<percentage>`: Un valor porcentual (`%`). El ancho se calcula como un porcentaje del **ancho del bloque contenedor** del elemento (su elemento padre, si es de bloque).

    ```
    .columna-mitad { width: 50%; } /* Ocupa la mitad del ancho de su contenedor */
    ```

  - `max-content`: El ancho intrínseco preferido (el necesario para mostrar todo el contenido sin saltos de línea si es posible).
  - `min-content`: El ancho intrínseco mínimo (el necesario si se permiten todos los saltos de línea posibles).
  - `fit-content(<length-percentage>)`: Una combinación, usando el espacio disponible pero sin superar el `max-content` ni ser menor que `min-content`.

`height`

- **Propósito:** Define la altura del área de contenido de un elemento.

- **Valores Comunes:**
  - `auto` **(Valor por defecto):** El navegador calcula la altura automáticamente para que quepa el contenido del elemento. Este es el comportamiento más común para elementos como párrafos, divs, etc., cuya altura se expande naturalmente con el contenido.
  - `<length>`**:** Un valor de longitud fija (`px`, `em`, `rem`). Define una altura exacta para el área de contenido.

    ```
    .cabecera-fija { height: 80px; }
    ```

  - `<percentage>`**:** Un valor porcentual (`%`). **¡Cuidado aquí!** Para que un porcentaje en `height` funcione, el **bloque contenedor (padre) debe tener una altura definida explícitamente** (no puede ser `auto`). Si el padre tiene `height: auto;`, el porcentaje del hijo no tendrá efecto (o se interpretará como `auto`). Esto es una fuente común de confusión.

    ```
    <div style="height: 400px; border: 1px solid blue;"> <!-- Padre con altura definida -->
      <div style="height: 50%; background-color: lightgreen;"> <!-- Hijo con altura porcent
    ual -->
        Ocupo el 50% de la altura del padre (200px).
      </div>
    </div>
    ```

    ```
    <div style="border: 1px solid red;"> <!-- Padre con altura auto -->
      <div style="height: 50%; background-color: lightcoral;"> <!-- Altura porcentual NO FU
    NCIONA -->
        Mi altura es 'auto' porque mi padre no tiene altura definida.
      </div>
    </div>
    ```

  - `max-content`, `min-content`, `fit-content()`**:** Funcionan de manera similar a `width`.

Propiedades Relacionadas: Límites de Tamaño

A menudo, no queremos un tamaño fijo, sino establecer límites para permitir que el contenido fluya pero sin exceder ciertas dimensiones. Para esto sirven:

- `min-width`**:** Establece el ancho *mínimo* que puede tener el área de contenido. El elemento puede ser más ancho si el contenido o un `width` explícito lo requieren, pero nunca más estrecho.
- `max-width`**:** Establece el ancho *máximo* que puede tener el área de contenido. El elemento puede ser más estrecho, pero nunca más ancho. Muy útil para la responsividad (ej., `max-width: 100%` para imágenes, asegurando que no desborden su contenedor).
- `min-height`**:** Establece la altura *mínima* del área de contenido. Puede crecer más si el contenido lo necesita.
- `max-height`**:** Establece la altura *máxima* del área de contenido. Si el contenido excede esta altura, puede desbordarse (controlado por la propiedad `overflow`, que veremos más adelante).

Ejemplo Combinado:

```css
.articulo {
  width: 80%;      /* Ocupa el 80% del ancho del padre */
  max-width: 900px;  /* Pero nunca más de 900px */
  min-width: 300px;  /* Y nunca menos de 300px */
  margin: 20px auto; /* Centrado (veremos 'margin: auto' más adelante) */
  padding: 25px;
  border: 1px solid #ccc;
  background-color: white;
  min-height: 150px; /* Asegura una altura mínima incluso si hay poco contenido */
}

img {
  max-width: 100%; /* Las imágenes se encogen para caber en su contenedor */
  height: auto;   /* Mantiene la proporción al cambiar el ancho */
}
```

Recordatorio sobre `box-sizing`

Es fundamental recordar que, por defecto (`box-sizing: content-box;`), `width` y `height` solo definen el área de contenido. Si usas el más común `box-sizing: border-box;`, entonces `width` y `height` definirán el tamaño total incluyendo contenido, padding y borde.

```css
.caja-border-box {
  box-sizing: border-box;
  width: 200px; /* Ancho TOTAL (contenido + padding + borde) es 200px */
  padding: 10px;
  border: 2px solid blue;
  /* El área de contenido será 200 - (10*2) - (2*2) = 176px */
}
```