29 CSS: Propiedades `float` & `clear`

Históricamente, las propiedades `float` y `clear` fueron pilares fundamentales para crear diseños de múltiples columnas y controlar la disposición de los elementos en la página, mucho antes de que existieran herramientas más modernas como Flexbox y CSS Grid. Aunque hoy en día su uso para layouts complejos ha disminuido, siguen siendo importantes para ciertos casos específicos, especialmente para el comportamiento clásico de texto fluyendo alrededor de imágenes.

La Propiedad `float`

La propiedad `float` se diseñó originalmente para permitir que el texto "flotara" alrededor de las imágenes, similar a lo que se ve en periódicos y revistas.

- **¿Qué hace?**
  Cuando aplicas
  `float` a un elemento, este es **sacado del flujo normal del documento** y desplazado hacia la izquierda (`float: left;`) o hacia la derecha (`float: right;`) de su contenedor, tanto como sea posible. El contenido que sigue al elemento flotante (especialmente texto y elementos en línea) **fluirá alrededor** de él.
  Un elemento flotante se convierte en un elemento de nivel de bloque (similar a `display: block;`), independientemente de su tipo original, aunque su tamaño puede ajustarse a su contenido si no se especifica un `width`.
- **Valores Principales:**
  - `left`: El elemento flota hacia la izquierda de su contenedor. El contenido circundante fluye por su lado derecho.
  - `right`: El elemento flota hacia la derecha de su contenedor. El contenido circundante fluye por su lado izquierdo.
  - `none`: (Valor predeterminado) El elemento no flota y permanece en el flujo normal del documento.
- **Ejemplo (Texto alrededor de imagen):**
  - **HTML:**

```
<div class="articulo">
  <img src="imagen.jpg" alt="Descripción de imagen" class="imagen-flotante">
  <p>Este es un párrafo de texto que describirá cómo el contenido fluye alrededor de la imagen flotante. Si el texto es lo suficientemente largo, veremos cómo ocupa el espacio disponible al lado de la imagen y luego continúa debajo de ella una vez que la imagen t ermina.</p>
  <p>Otro párrafo que también respeta el espacio ocupado por la imagen flotante hasta que esta termine.</p>
</div>
```

  - **CSS:**

```
.imagen-flotante {
  float: left; /* La imagen flota a la izquierda */
```

```
  width: 150px; /* Damos un ancho a la imagen */
  height: auto; /* Mantenemos la proporción */
  margin-right: 15px; /* Espacio entre la imagen y el texto */
  margin-bottom: 5px; /* Espacio debajo de la imagen */
}
.articulo {
  border: 1px solid #ccc;
  padding: 10px;
  width: 500px; /* Ancho del contenedor */
}
```

- **Resultado:** La imagen se colocará a la izquierda dentro del `div.articulo`. El texto de los párrafos comenzará a la derecha de la imagen y continuará hacia abajo, ocupando el ancho completo del contenedor una vez que se sobrepase la altura de la imagen.

- **El Problema del Contenedor Colapsado:**
  Un efecto secundario crucial de `float` es que los elementos flotantes **son eliminados del flujo normal**. Esto significa que el **contenedor padre deja de "verlos"** para calcular su propia altura. Si un contenedor solo contiene elementos flotantes (o si los elementos flotantes son más altos que el contenido no flotante), la altura del contenedor puede colapsar a cero (o a la altura del contenido no flotante), haciendo que los elementos flotantes parezcan "salirse" del contenedor.
  - **Ejemplo (Contenedor Colapsado):**

```
<div class="contenedor-colapsado">
  <div class="columna-flotante">Col 1</div>
  <div class="columna-flotante">Col 2</div>
  <!-- Sin contenido no flotante dentro -->
</div>
```

```
.contenedor-colapsado {
  border: 2px solid red; /* Borde para ver el tamaño del contenedor */
}
.columna-flotante {
  float: left;
  width: 100px;
  height: 100px;
  background-color: lightblue;
  margin: 5px;
}
```

  - **Resultado:** Verás las dos cajas azules flotando, pero el borde rojo del contenedor probablemente estará solo en la parte superior, con altura cero o muy pequeña, porque no contiene a las cajas flotantes.

La Propiedad `clear`

La propiedad `clear` se utiliza en un elemento para especificar en qué lados **no debe permitir elementos flotantes adyacentes**. Es decir, fuerza al elemento a moverse *debajo* de cualquier

elemento flotante anterior que esté en el lado especificado.

- **¿Qué hace?**
  Controla la interacción de un elemento con los elementos flotantes que lo preceden en el código HTML. Empuja el elemento hacia abajo hasta que su borde superior quede por debajo del borde inferior de los elementos flotantes especificados.
- **Valores Principales:**
  - `left`: El elemento se mueve hacia abajo hasta despejar (pasar por debajo) cualquier elemento flotante a la izquierda.
  - `right`: El elemento se mueve hacia abajo hasta despejar cualquier elemento flotante a la derecha.
  - `both`: El elemento se mueve hacia abajo hasta despejar elementos flotantes tanto a la izquierda como a la derecha. Este es el valor más comúnmente usado.
  - `none`: (Valor predeterminado) El elemento no se mueve para despejar flotantes y permite que floten a sus lados si hay espacio.
- **Ejemplo (Usando `clear`):**
  - **HTML:**

```
<div class="contenedor">
  <div class="flota-izquierda">Flota Izquierda</div>
  <p>Este texto fluye al lado...</p>
  <div class="despejar-ambos">Este elemento despeja ambos lados.</div>
  <p>Este texto aparece debajo del elemento despejado.</p>
</div>
```

  - **CSS:**

```
.flota-izquierda {
  float: left;
  width: 100px; height: 50px; background: lightcoral; margin-right: 10px;
}
.despejar-ambos {
  clear: both; /* Fuerza a este div a empezar debajo del float */
  border-top: 1px solid black; /* Para visualizar dónde empieza */
  padding-top: 5px;
}
.contenedor { border: 1px solid grey; padding: 10px; width: 300px;}
```

  - **Resultado:** El `div.flota-izquierda` flotará a la izquierda. El primer párrafo fluirá a su derecha. El `div.despejar-ambos` será forzado a empezar en una nueva línea *debajo* del elemento flotante, y el último párrafo aparecerá debajo de este.

Soluciones al Contenedor Colapsado (Clearfix)

Para solucionar el problema del contenedor que colapsa cuando contiene solo elementos flotantes, se utilizan técnicas conocidas como "clearfix". El objetivo es hacer que el contenedor "contenga" a sus hijos flotantes y ajuste su altura correctamente.

1. **Método `overflow`:**
   Aplicar `overflow: hidden;` o `overflow: auto;` (o `scroll`, `clip`) al **contenedor padre**. Esto funciona porque establecer un valor de `overflow` distinto de `visible` crea un nuevo **Contexto de Formato de Bloque (BFC)**. Un BFC, por definición, contiene a sus elementos flotantes internos.

   - **CSS:**

     ```css
     .contenedor-clearfix-overflow {
       border: 2px solid green;
       overflow: auto; /* o hidden */
     }
     .contenedor-clearfix-overflow .columna-flotante {
       float: left;
       width: 100px; height: 100px; background-color: lightgreen; margin: 5px;
     }
     ```

   - **Ventaja:** Muy simple.
   - **Desventaja:** Puede tener efectos secundarios no deseados si realmente necesitabas que el contenido desbordara (`overflow: visible`) o si el contenido interno legítimamente necesita barras de scroll (`auto` / `scroll`) o ser recortado (`hidden` / `clip`) por otras razones.

2. **Método del Pseudoelemento `::after` (Clearfix Moderno):**
   Esta es la técnica más robusta y comúnmente utilizada. Se añade un pseudoelemento invisible (`::after`) al final del contenedor, se le da estilo para que actúe como una tabla (lo que también crea un BFC o un contexto similar) y se le aplica `clear: both;`.

   - **CSS:**

     ```css
     .contenedor-clearfix-after {
       border: 2px solid blue;
       /* La 'magia' del clearfix */
     }
     .contenedor-clearfix-after::after {
       content: ""; /* Necesario para que el pseudo-elemento se genere */
       display: table; /* O display: block; en algunos casos */
       clear: both; /* La clave para despejar los floats internos */
     }

     .contenedor-clearfix-after .columna-flotante {
       float: left;
       width: 100px; height: 100px; background-color: lightblue; margin: 5px;
     }
     ```

   - **Ventaja:** No interfiere con la propiedad `overflow` del contenedor y no requiere HTML adicional. Es la solución preferida en la mayoría de los casos.

3. **Elemento Vacío con `clear` (Método Antiguo - No Recomendado):**
   Consistía en añadir un elemento HTML vacío (ej. `<div style="clear: both;"></div>`) justo antes de cerrar el contenedor.

- **Desventaja:** Añade HTML no semántico solo por motivos de presentación, lo cual es una mala práctica. Evitar este método.

Uso Moderno de `float`:

- **Principalmente:** Para que el texto y elementos en línea fluyan alrededor de imágenes u otros elementos específicos (su propósito original).
- **Evitar para Layouts:** Para crear estructuras de columnas, rejillas o alinear secciones principales de una página, **utiliza Flexbox o CSS Grid**. Son mucho más potentes, flexibles y predecibles para el diseño de layouts complejos. `float` para layout es una técnica heredada con muchas peculiaridades.
