Propiedad `display` en CSS

La propiedad `display` en CSS define cómo se debe mostrar un elemento en la página web. Controla el **modelo de caja** (box model) de los elementos, lo que afecta su disposición y comportamiento dentro del flujo del documento. Esta propiedad tiene varios valores que determinan si un elemento será visible, cómo se organizará dentro de su contenedor y cómo interactuará con los demás elementos.

Valores comunes de `display`

1. `display: none`

Este valor oculta el elemento completamente de la página, es decir, el elemento no se mostrará ni ocupará espacio en el flujo del documento. No es visible y no afecta la disposición de los demás elementos.

- **Ejemplo**:

```
div {
  display: none;
}
```

En este caso, el `div` no se mostrará en absoluto, como si no existiera.

- **Diferencia con `visibility: hidden`**: Mientras que `display: none` elimina el elemento del flujo, `visibility: hidden` lo hace invisible, pero el elemento sigue ocupando espacio.

---

2. `display: block`

El valor `block` hace que el elemento sea un **elemento en bloque**. Los elementos en bloque:

- Ocupan todo el ancho disponible (por defecto).
- Comienzan en una nueva línea.
- Pueden tener un ancho, alto, márgenes y padding definidos.

Los elementos como `<div>`, `<p>`, `<h1>` y `<section>` son, por defecto, elementos en bloque.

- **Ejemplo**:

```
div {
  display: block;
}
```

Aquí, el `div` ocupará toda la anchura de su contenedor, y cualquier contenido que venga después se situará debajo de él en una nueva línea.

---

3. `display: inline-block`

El valor `inline-block` combina comportamientos de **elementos en línea** e **elementos en bloque**:

- Se comporta como un elemento en línea, por lo que no fuerza un salto de línea y puede situarse junto a otros elementos en línea.
- Sin embargo, como un elemento en bloque, puede tener un ancho, alto, márgenes y padding definidos.

Esto lo hace muy útil para elementos como botones, menús o tarjetas que necesitan ser distribuidos horizontalmente pero también tener un tamaño y espaciado controlado.

- **Ejemplo**:

```
span {
  display: inline-block;
  width: 150px;
  height: 100px;
  background-color: lightblue;
}
```

En este ejemplo, el `span` actuará como un elemento en línea, permitiendo que otros elementos se sitúen a su lado, pero también tendrá dimensiones como ancho y alto.

Funcionamiento de `float`

Cuando se aplica `float` a un elemento, este se "saca" del flujo normal del documento y se alinea en la dirección especificada (izquierda o derecha), permitiendo que otros elementos en línea o en bloque lo rodeen.

La propiedad `float` en CSS se utiliza para colocar elementos dentro de un contenedor de manera que el texto o los otros elementos "floten" alrededor de ellos. Es una técnica de diseño usada principalmente para crear disposiciones de elementos en columnas, como imágenes alineadas a un lado con texto que las rodea.

Valores de `float`

1. `float: left;`
    - El elemento se alinea al lado izquierdo de su contenedor.
    - Los elementos que siguen en el flujo del documento lo rodean por el lado derecho.

    ```
    img {
      float: left;
      margin-right: 10px;
    }
    ```

    En este caso, la imagen se alineará a la izquierda y el texto o los demás elementos fluirán a su derecha.

2. `float: right;`

- El elemento se alinea al lado derecho de su contenedor.
- Los elementos que siguen en el flujo del documento lo rodean por el lado izquierdo.

```
img {
  float: right;
  margin-left: 10px;
}
```

Aquí, la imagen se alinea a la derecha y el contenido fluirá a su izquierda.

3. `float: none;`
   - Este es el valor predeterminado. Indica que el elemento no flota y sigue el flujo normal del documento.

```
img {
  float: none;
}
```

4. `float: inherit;`
   - Hace que el elemento herede el valor de `float` de su elemento contenedor.

Cómo se comportan los elementos flotantes

Cuando se aplica `float` a un elemento, el comportamiento del flujo de los elementos cambia:

- Los **elementos en bloque** se ajustan alrededor del elemento flotante.
- Los **contenedores** que no incluyen ningún elemento flotante pueden "colapsar" si no se maneja correctamente el flotado. Esto ocurre porque los elementos flotantes quedan fuera del flujo normal y no son considerados al calcular la altura del contenedor.

Ejemplo con una imagen flotante:

```
<style>
  img {
    float: left;
    margin-right: 20px;
  }
</style>

<div>
  <img src="imagen.jpg" alt="Imagen flotante" />
  <p>
    Este es un párrafo de texto que flota alrededor de la imagen a la izquierda.
```

```
    </p>
  </div>
```

En este ejemplo, la imagen flota a la izquierda del contenedor y el texto se ajusta alrededor de ella.

El problema de contener elementos flotantes y la solución con `clear`

Uno de los problemas más comunes con el uso de `float` es que los elementos contenedores no "envuelven" a los elementos flotantes, lo que puede hacer que el diseño colapse. Para solucionarlo, se utiliza la propiedad `clear` para evitar que los elementos floten a los lados de otros elementos.

Valores de `clear`:

- `clear: left;`: Evita que el elemento flote a la izquierda.
- `clear: right;`: Evita que el elemento flote a la derecha.
- `clear: both;`: Evita que el elemento flote tanto a la izquierda como a la derecha.

```
.clearfix::after {
  content: "";
  display: block;
  clear: both;
}
```

El uso de un **clearfix** (como el anterior) es una técnica común para asegurarse de que el contenedor "abarque" correctamente los elementos flotantes.

Ejemplo completo:

```
<style>
  .container {
    border: 1px solid #000;
  }

  img {
    float: left;
    margin-right: 10px;
  }

  .clearfix::after {
    content: "";
    display: block;
    clear: both;
  }
</style>

<div class="container clearfix">
```

```
    <img src="imagen.jpg" alt="Imagen flotante" />
    <p>
      Este es un párrafo de texto que flota alrededor de la imagen. La imagen está
      alineada a la izquierda y el texto la rodea por la derecha.
    </p>
  </div>
```

En este ejemplo:

- La imagen flota a la izquierda, y el párrafo de texto se ajusta a su derecha.
- La clase `clearfix` asegura que el contenedor abarque tanto la imagen flotante como el texto.

Uso moderno de `float`

Aunque `float` ha sido históricamente una técnica clave para la disposición de elementos en columnas y layouts, su uso ha disminuido en el desarrollo web moderno. Hoy en día, es común usar otras herramientas como **Flexbox** o **CSS Grid** para crear layouts más eficientes y flexibles. Sin embargo, `float` sigue siendo útil en situaciones específicas, como alinear imágenes con texto que fluya alrededor de ellas.

Ir a posicionamiento