Problemas con flotantes en CSS y sus soluciones

- C.F.G.S. DAW
- 6 horas semanales
- Mes aprox. de impartición: Nov - Dic
- iPasen - cjaedia071@g.educaand.es

Carmelo José Jaén Díaz

Índice

OBJETIVO

- Analizar y seleccionar los colores y las tipografías adecuados para la visualización en la pantalla.
- Utilizar marcos y tablas para presentar la información de manera ordenada.
- Identificar nuevos elementos y etiquetas en HTML5.
- Reconocer las posibilidades de modificar etiquetas HTML.
- Valorar la utilidad de las hojas de estilo para conseguir un diseño uniforme en todo el sitio web.

GLOSARIO

Formularios. Documentos interactivos utilizados para recoger información en un sitio web. Esta información es enviada al servidor, donde es procesada. Cada formulario contiene uno o varios tipos de controles que permiten recolectar la información de varias formas diferentes.

Fuentes seguras. Fuentes tipográficas que los usuarios tenían instaladas por defecto en su dispositivo. En la actualidad, gracias a que la mayoría de los navegadores soportan la directiva @font-face, es posible utilizar casi cualquier tipografía a través de Google Fonts.

Guías de estilo. Documentos con directrices que permiten la normalización de estilos. En estas guías se recogen los criterios y normas que debe seguir un proyecto; de esta forma se ofrece una apariencia más uniforme y atractiva para el usuario.

HTML. Lenguaje de marcado de hipertexto utilizado en las páginas web. Este tipo de lenguaje presenta una forma estructurada y agradable, con hipervínculos que conducen a otros documentos y con inserciones multimedia (sonido, imágenes, vídeos...).

GLOSARIO

HTML5. Última versión del lenguaje para la programación de páginas web HTML. Los sitios implementados con este lenguaje solo pueden visualizarse correctamente en los navegadores más actuales.

Legibilidad. Cualidad deseable en una familia tipográfica. Se trata de la facilidad de la lectura de una letra. Esta cualidad puede venir determinada por varios parámetros como el interletrado, el interpalabrado o el interlineado.

Marcos. Son las ventanas independientes incorporadas dentro de la página general. Gracias a ellos, cada página quedará dividida en varias subpáginas, permitiendo realizar un diseño más organizado y limpio a la vista. Con HTML5 ha quedado obsoleto.

Tipografía. Se trata del tipo de letra que se escoge para un determinado diseño. Según la RAE, significa ”modo o estilo en que está impreso un texto” o ”clase de tipos de imprenta”.

INTRODUCCIÓN

Esta lección explora los **problemas comunes** al usar **flotantes en CSS**, como son el contenedor que no contiene a sus flotantes y el colapso de contenedores y alineación irregular.

Se ofrecen técnicas claras y prácticas para manejar el comportamiento de los flotantes.

Problemas con flotantes en CSS y sus soluciones

El uso de **flotantes** (float) en CSS ha sido una técnica común para el diseño de layout antes de la introducción de Flexbox y CSS Grid.

Sin embargo, trabajar con flotantes puede llevar a varios errores o problemas comunes, especialmente en lo que respecta al **flujo del documento y la contención de elementos**.

A continuación, se plantean algunos ejemplos de estos **errores y cómo solucionarlos**.

Problema del contenedor que no contiene sus flotantes

Se parte del siguiente escenario:

Código HTML:

```
<div class="container">
    <div class="floating">Elemento
Flotante 1</div>
    <div class="floating">Elemento
Flotante 2</div>
    <!-- El contenedor no "contiene"
los elementos flotantes -->
</div>
```

Código CSS:

```
.container {
    background-color: lightblue;
    /* El contenedor no se expande
para contener los flotantes */
}

.floating {
    float: left;
    width: 50%;
}
```

Problema del contenedor que no contiene sus flotantes

Problema: El contenedor `.container` no se expande para incluir los elementos flotantes, por lo que el fondo no se muestra como se espera. (En este caso de color *`lightblue`*)

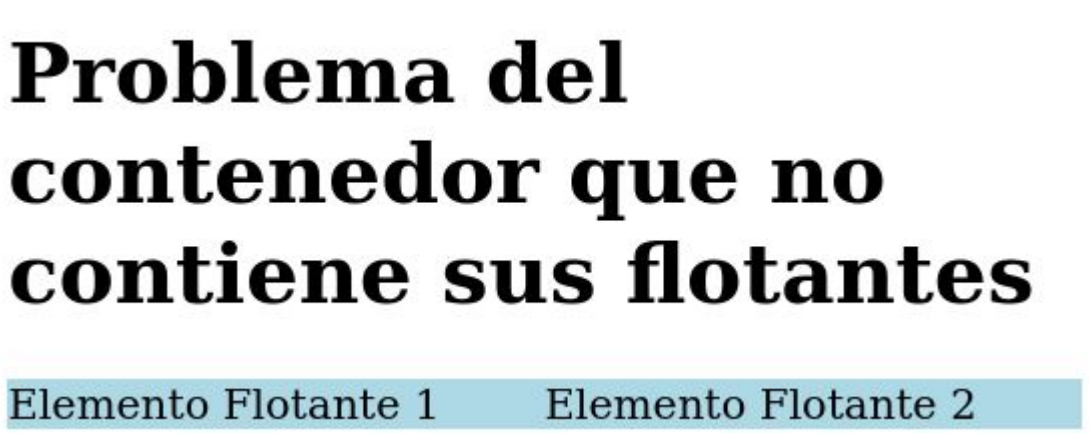
Problema del contenedor que no contiene sus flotantes
Solución I

Solución: Utilizar la técnica del «clearfix» o `overflow: hidden`.

Código CSS corregido:

```css
.container {
    background-color: lightblue;
    overflow: hidden;
}

.floating {
    float: left;
    width: 50%;
}
```

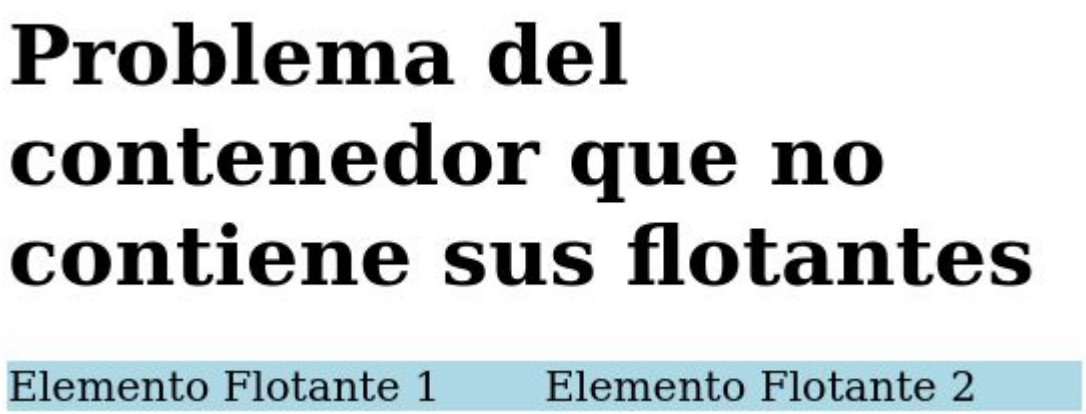


Problema del contenedor que no contiene sus flotantes
Solución II

Solución: limpiar los flotantes con `clear:both;`

Código CSS corregido:

```css
.container {
    background-color: lightblue;
}
.floating {
    float: left;
    width: 50%;
}
.container::after {
    content: "";
    display: block;
    clear: both;
}
```



PROBLEMA DE COLAPSAMIENTO DE CONTENEDORES

Se parte del siguiente escenario:

Código HTML:

```
<div class="container">
    <div class="floating">Elemento
Flotante 1</div>
    <div class="floating">Elemento
Flotante 2</div>
</div>
<p>Otro contenido aquí...</p>
```

Código CSS:

```
.floating {
    float: left;
    width: 50%;
}
.container {
    overflow: hidden;
}
p {
    background-color: lightblue;
}
```

PROBLEMA DE COLAPSAMIENTO DE CONTENEDORES

Problema: Al definir de color *lightblue* el párrafo (<p>) que hay a continuación de los flotantes, se ponen de color azul también los flotantes

Problema de colapsamiento de contenedores. Solución

Solución: Asegurarse de limpiar los flotantes con `clear:both;` y que los flotantes no tengan ese color.

Código CSS corregido:

```css
.floating {
    float: left;
    width: 50%;
}
.container {
    overflow: hidden;
}
p {
    clear: both;
    background-color: lightblue;
}
```

Problema de alineamiento de flotantes

Problema de alineamiento de flotantes

Se parte del siguiente escenario:

Código HTML:

```
<div class="container">
    <div class="floating
tall">Elemento Flotante con mucho
texto: Lorem ipsum dolor sit amet
consectetur adipiscing elit taciti
malesuada odio rhoncus...</div>
    <div class="floating
short">Elemento Flotante Bajo</div>
</div>
```

Código CSS:

```
.floating {
    float: left;
    width: 50%;
}
.container {
    overflow: hidden;
}
p {
    background-color: lightblue;
}
```

Problemas con flotantes en CSS y sus soluciones

En los ejemplos anteriores has estudiado algunos problemas comunes al trabajar con flotantes en CSS y cómo puedes solucionarlos. Sin embargo, en la actualidad, Flexbox y CSS Grid proporcionan métodos más eficientes y flexibles para crear layouts sin muchos de estos problemas inherentes a los flotantes.

El truco de la clase *.clearfix* para solucionar errores con flotantes

El «clearfix» es un truco clásico en CSS que se utiliza para resolver un problema común con los elementos flotantes (`float`). Cuando se usa `float` en un elemento, este elemento sale del flujo normal del documento y no afecta la altura de su contenedor padre. Esto puede llevar a que el contenedor no se expanda para envolver a sus hijos flotantes, afectando el layout de la página.

El truco `clearfix` se emplea para forzar al contenedor a expandirse y envolver completamente a sus hijos flotantes, manteniendo así la integridad del layout. Veamos cómo funciona:

Paso 1: Crear la clase *clearfix*

Primero, se define una clase *clearfix* en CSS:

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

Esta regla CSS utiliza un pseudo-elemento (`::after`) para insertar contenido vacío después del último elemento flotante dentro del contenedor. La propiedad `display: blocl;` asegura que este pseudo-elemento se comporte como un elemento de bloque, y `clear: both;` fuerza al pseudo-elemento a moverse debajo de ambos flotantes (tanto izquierdo como derecho). Esto efectivamente «limpia» los flotantes y hace que el contenedor padre reconozca la altura total de sus elementos hijos flotantes.

Paso 2: Aplicar la clase al contenedor padre de los elementos flotantes

Luego, aplicas esta clase al contenedor de tus elementos flotantes:

```
<div class="clearfix">
    <div style="float: left;">Contenido Flotante 1</div>
    <div style="float: left;">Contenido Flotante 2</div>
    <!-- El clearfix se aplica al contenedor -->
</div>
```

En este ejemplo, el contenedor con la clase `clearfix` ahora se expandirá para envolver completamente a sus elementos flotantes, asegurando que el layout se mantenga como se esperaba.

¿Por qué y cuándo es necesario?

Antes de Flexbox y CSS Grid, los flotantes eran una técnica común para layouts, pero este problema de contenedores que no reconocían la altura de los elementos flotantes causaba muchos dolores de cabeza a los desarrolladores. El truco `clearfix` era una solución ampliamente adoptada para abordar este problema.

Hoy en día, con técnicas modernas como `Flexbox` y `CSS Grid`, el uso de flotantes (y por ende, el uso del truco `clearfix`) ha disminuido, ya que estas nuevas técnicas ofrecen una forma más intuitiva y flexible de manejar layouts complejos sin los problemas asociados con los flotantes.