ThePower business school

Position CSS 🔃🎨

En líneas generales, cuando hablamos de elementos en línea nos referimos a elementos que en nuestro navegador aparecerán apilados unos al lado de otros de izquierda a derecha. Por otro lado, los elementos en bloque ocuparán todo el ancho disponible por lo que aparecerán apilados uno encima de otro. La mezcla de estos elementos compondrán nuestra página web.

A la hora de cambiar el funcionamiento por defecto de la posición de los elementos, podremos darle a la propiedad position los siguientes valores:

- static: es el posicionamiento por defecto y no será necesario especificarlo en nuestros elementos.

- relative: los elementos se colocan igual que con el valor static pero la diferencia será que se nos activarán las propiedades top, bottom, left y right para posicional los elementos a nuestro antojo.

- absolute. Con este valor los elementos se posicionarán tomando como referencia el primer contenedor que tenga posicionamiento diferente a static.

  Este posicionamiento es el más complejo de usar y entender. Por ejemplo, si el contenedor padre tiene posicionamiento estático, pasamos a mirar el posicionamiento del padre del contenedor padre, y así sucesivamente hasta encontrar un contenedor con posicionamiento no estático o llegar a la etiqueta `<body>`, en el caso que se comportaría como el ejemplo anterior.

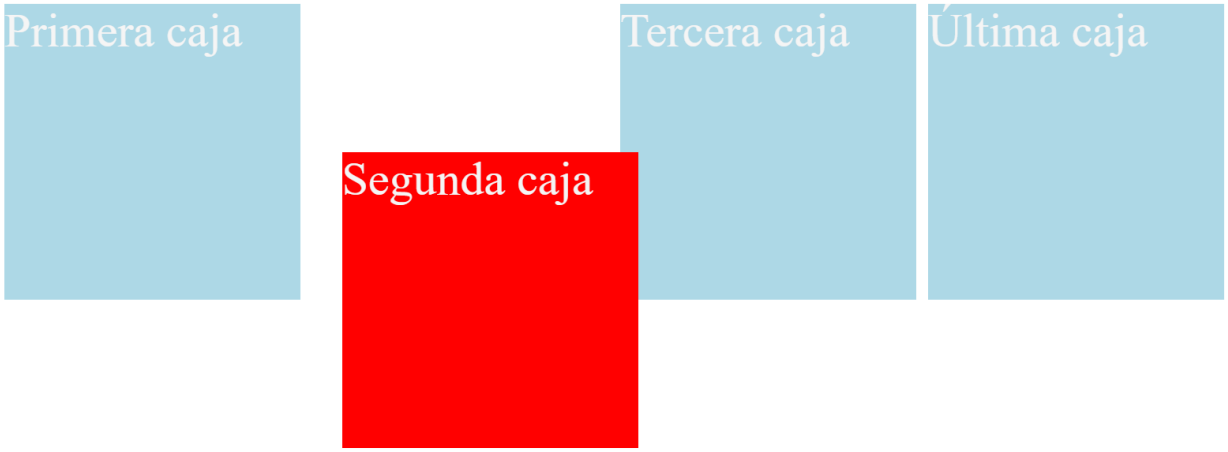
- fixed: es parecido al absolute, salvo por el detalle de que el posicionamiento fixed el elemento se mostrará en una posición fija **dependiendo de la región visual del navegador. Por lo tanto, con ese tipo de posicionamiento, aunque hagamos scroll el elemento seguirá en la misma posición.**

- sticky. Se denomina también “pegado”. Se asemeja al relativo, y se suele usar para fijar los <nav> de las páginas de forma que aunque hagamos scroll no perderemos el <nav> de vista.

Una vez desactivamos el comportamiento por defecto (es decir, establecemos la propiedad position en algún valor distinto de static), se nos activarán las propiedades top, bottom, left y right que hará que los elementos se desplacen en la dirección que marquemos. Estas propiedades serán fundamentales.

Os dejamos un ejemplo de código para que probéis y practiquéis con esta propiedad:

```
<div class="box" id="one">Primera caja</div>
<div class="box" id="two">Segunda caja</div>
<div class="box" id="three">Tercera caja</div>
<div class="box" id="four">Última caja</div>
```




```
* {
    box-sizing: border-box;
  }

  .box {
    display: inline-block;
    width: 100px;
    height: 100px;
    background: lightblue;
    color: whitesmoke;
  }

  #two {
```

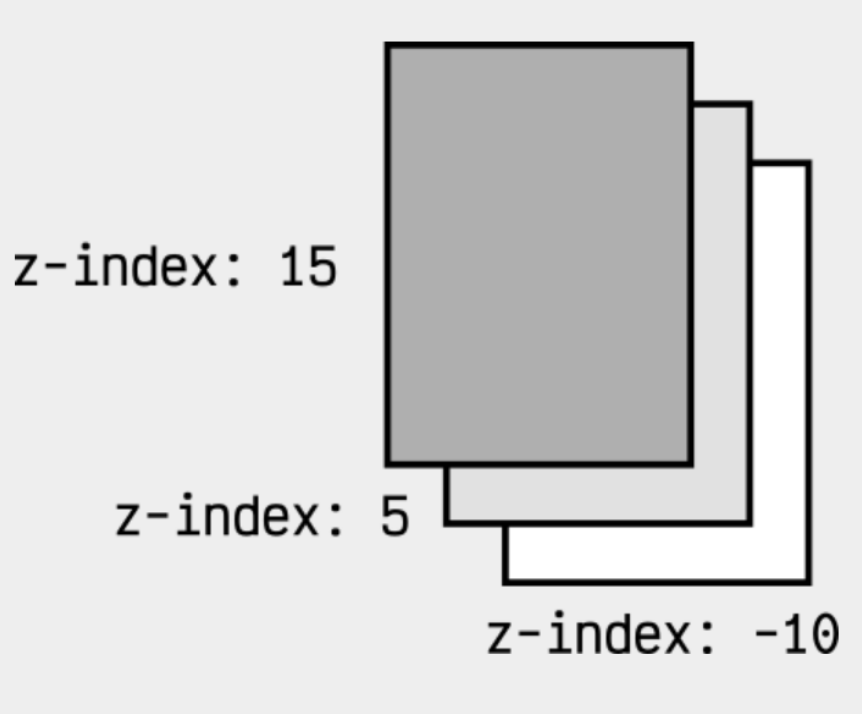
```
    position: relative;
    top: 50px;
    left: 10px;
    background: red;
  }
```

Un concepto interesante también es el de la profundidad o niveles en un eje z, que define la profundidad de un elemento. Esta profundidad de la que hablamos se cambia con la propiedad z-index, y su funcionamiento es que habrá que indicar el valor que queremos darle a la profundidad del elemento.

Tened en cuenta que el elemento que tenga el z-index más alto será el que mayor profundidad tenga. Esto nos será útil a la hora de traer al fondo-frente algunos elementos y apilarlos. Esta propiedad requerirá un valor de la propiedad position diferente de static.

Os dejamos un ejemplo de código y una imagen para que os aclare el funcionamiento.

```
<div class="rojo">Rojo
    <div class="verde">Verde</div>
</div>
<div class="gris">Gris</div>
<div class="azul">Azul</div>
```




```
.amarillo {
    z-index: 0;
}
.rojo {
    z-index: 1;
    width: 100px;
    height: 20px;
    background-color: red;
}
.gris {
    z-index: 2;
}
.verde {
    z-index: 3;
    position: relative;
    bottom: 15px;
    left: 40px;
    width: 100px;
    height: 20px;
    background-color: green;
}
.azul {
    z-index: 100;
}
```

SI queremos más control aún a la hora de posicionar nuestros elementos tenemos una propiedad float disponible. Esta propiedad como su nombre indica hará que el elemento “flote” a izquierda o derecha dependiendo del valor que le demos.

Aún así, siempre será mejor usar la propiedad display en lugar de float, ya que el resultado será más limpio con display y float es más complejo de usar.

Si por lo que sea lo que queremos es impedir elementos flotantes tenemos la propiedad clear.

¿Qué hemos aprendido?

1. Hemos explicado lo que implica posicionar elementos y con qué propiedad se puede modificar.
2. Los diferentes valores que puede tomar la propiedad position y para qué nos pueden ser útiles.
3. El término profundidad y cómo modificarlo.
4. Propiedad float para complementar el posicionamiento de elementos.