Aplicar diseño

10. header

Avatar

Probar flotar hacia la derecha para ver qué pasa.

```
header #avatar
{
    float: right;
}
```

Aplicar diseño

Avatar
Probar flotar hacia la derecha para ver qué pasa.

```
header #avatar
{
    float: right;
}
```

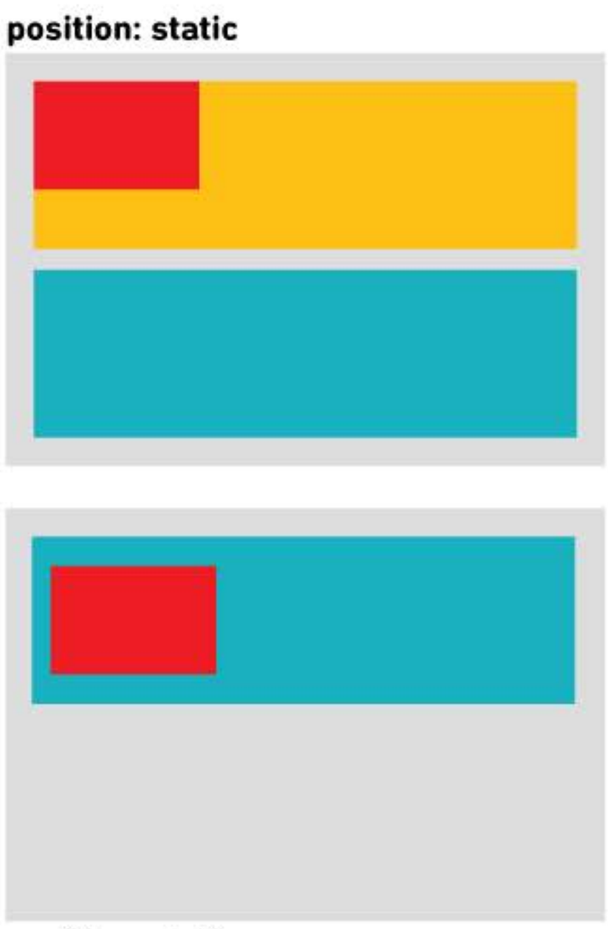
El avatar si se posiciona a la derecha pero queda a la altura inferior porque el logo lo está afectando.

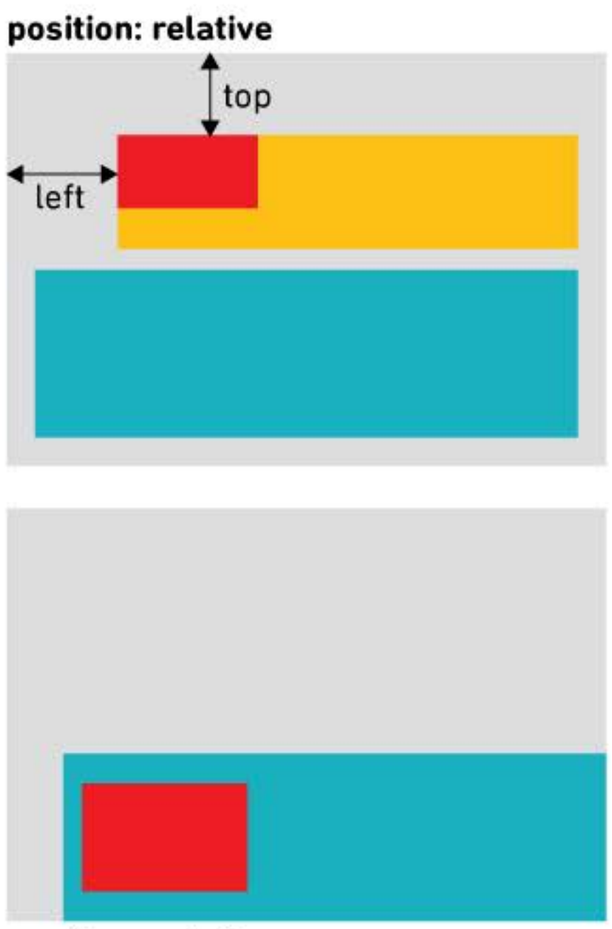
Un recurso para que no quede por debajo es poner en negativo los márgenes.

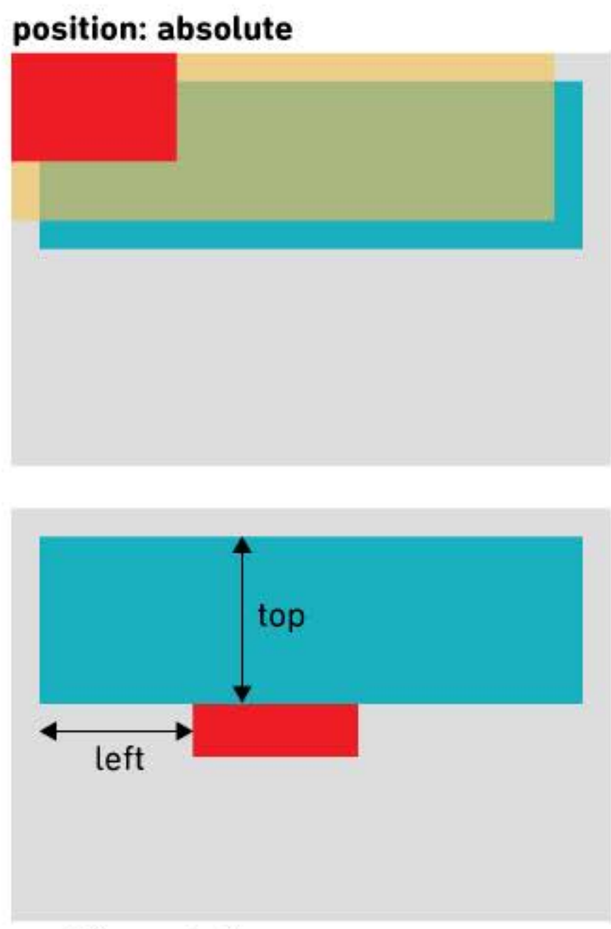
```
header #avatar
{
    float: right;
    margin: -70px 0 0 0;
}
```

Position

Cómo funcionan las posiciones en el modelo de caja







position: static

position: relative

position: relative
position: absolute

Posiciones absolutas

Las cajas tienen posiciones.
La posición por defecto de una caja es **static**.

El Display block hace que "llene" el espacio lateral y salte de línea lo que sigue.

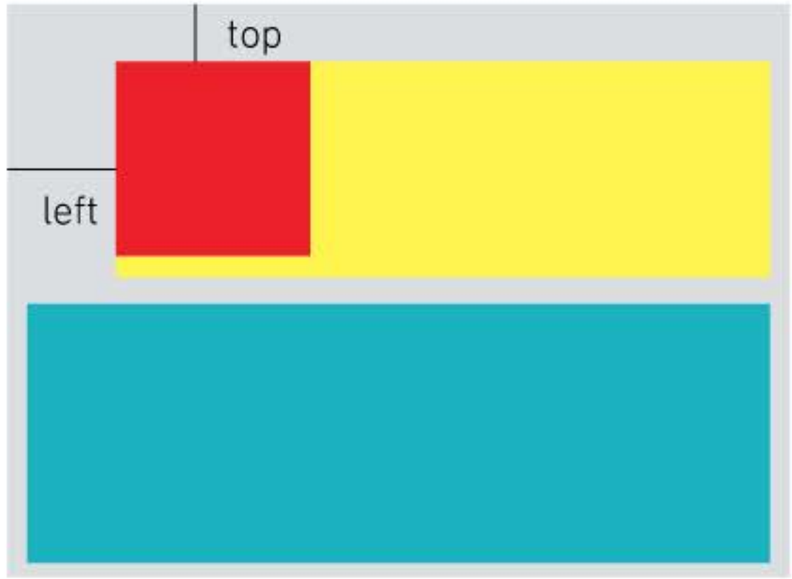
position: static

Posiciones absolutas

Al gregarle a una caja relative, ésta obtiene 4 atributos. Son top, left, right y bottom.
Por defecto no hacen nada ya que están en cero.

Con esos atributos, podemos mover sobre un eje cartesiano sin que importen los márgenes.

No afecta a los márgenes pero si a lo demás que está dentro del contenedor.



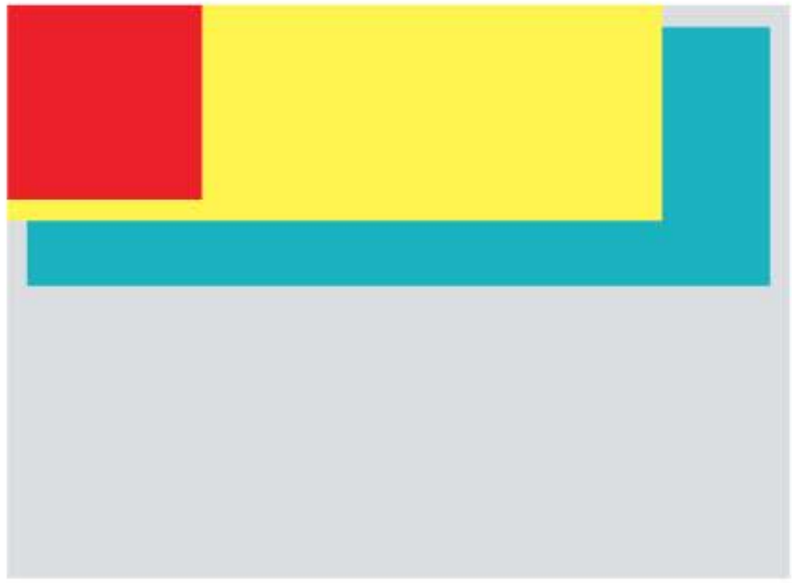
position: relative

Posiciones absolutas

Al gregarle a una caja absolute, ésta se desvincula-
del eje cartesiano y se dirije al punto cero más cercano que encuentre primero.

El punto cero más cercano siempre va a ser <body>.

Para "mover" ese punto cero, podemos aplicarle al contenedor position relative.



position: absolute

Posiciones absolutas

En este ejemplo tenemos una **caja static**, que es la posición por defecto dentro de otra **caja**.

position: static

Posiciones absolutas

Si a nuestra **caja** le aplicamos **relative**, la **caja static**, se mantiene posicionada.

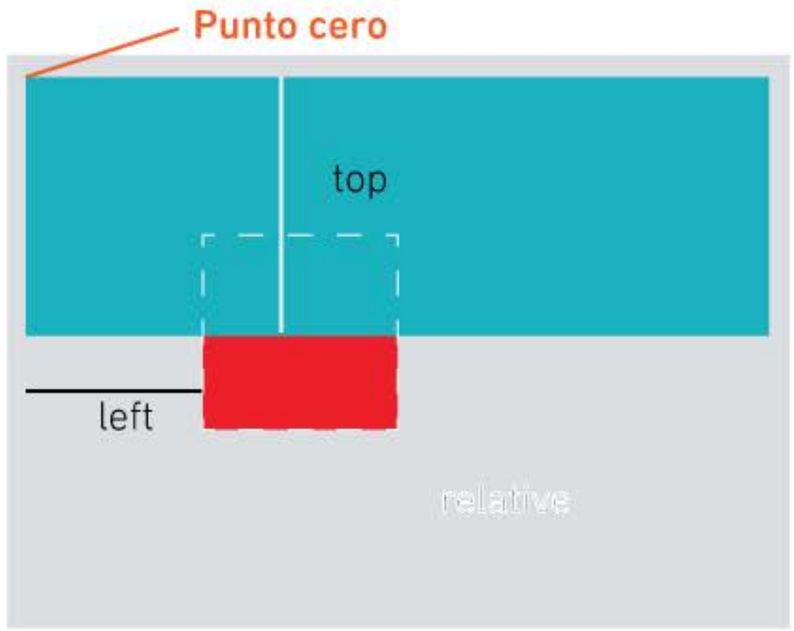
position: relative

Posiciones absolutas

Si a la caja **contenedora** le aplicamos position relative no va a pasar nada, pero si a la **caja** le aplicamos absolute, ésta va a busacr cuál es el contenedor más cercano para fijar su punto cero.

Esta es la forma que se utiliza position en la actualidad.

Punto cero: Es la posición contenedora más cercana que posea relative.



position: relative
position: absolute

Aplicar diseño

header

Posicionar el Avatar.

Queremos llevarlo a la parte superior derecha dentro del header.

```
header #avatar
{
    left: 0; /*para ver la posición del punto cero */
    position: absolute;
    top: 0;
}
```

Para visualizar mejor el ejemplo, darle margen al header temporalmente. (Luego eliminar)

```
header
{
    background: #474747;
    color: white;
    margin: 2em;
    overflow: hidden;
    padding: 0.25em;
}
```

Aplicamos relative al header. El avatar va a tomar como punto cero a la posición del header.

```
header
{
    background: #474747;
    color: white;
    overflow: hidden;
    padding: 0.25em;
    position: relative;
}
```

Para llevar el avatar a la derecha.

```
header #avatar
{
    margin: 0.5em 0.5em 0 0;
    position: absolute;
    right: 0;
    top: 0;
}
```

Aplicar diseño

Menú. Empezamos de lo general a lo particular.

Color de fondo del menú #FBBF16

```
nav
{
    background: #FBBF16;
    margin: 0; /* No quita el margen superior */
}
```

¿Por qué pasa esto? Porque los <ul> por defecto tienen márgenes.
En CSS cuando un <ul> es el primer elemento de una caja y ésta no tiene padding.

Aplicar diseño

Menú.

Para eliminar el espacio que se generó entre el **nav** y el **header**, primero a **nav** lo dejamos:

```
nav
{
    background: #FBBF16;
}
```

Y llevamos los árgenes a cero del **ul**

```
nav ul
{
    list-style: none; /* Elimina los puntos */
    margin: 0;
    padding: 0;
}
```

Aplicar diseño

Menú.

Fíjense a esta altura, los elementos del menú me salen uno debajo de otro. Esto se debe a que los **li** son cajas, por defecto cada **li** es un **block**.
Entonces tenemos que reacomodar a esas cajas para que no se ubiquen una debajo de la otra.

```
nav ul li
{
    display: inline-block;
}
```

Damos color al texto.

```
nav ul li a
{
    color: #474747;
}
```

Aplicar diseño

Menú.

Cada **li** es **inline-block** o sea que mantengo los márgenes. Para darle espacio dentro del **nav** a cada **li**:

```
nav ul li
{
    display: inline-block;
    margin: 0 1em 0 0;
}
```

Damos **padding** al **nav** contenedor de los **li**.

```
nav
{
    background: #FBBF16;
    padding: 0 0.5em; /* El primer valor cooresponde a los 2 primeros, arriba y abajo. El otro es a los costados */
}
```

Le daremos **padding** a los **a** arriba y abajo.

Aplicar diseño

Menú.

Damos **padding** a los **a** arriba y abajo.

```
nav ul li a
{
    color: #474747;
    display: block; /* Para poder poner padding interno a los "a", así tienen padding y margin */
    padding: 0.5em 0;
}
```

Hacemos esto para hacer clickeable a toda la zona de cada **a**. Si le hubiéramos dado **padding** al **nav**, la superficie clickeable del **a** se rediciría.

Aplicar diseño

Menú. Nuevo+

Damos **border** a **#nuevo_nav** para ver cómo lo posicionamos.

```
nav ul #nuevo_nav
{
    border: 1px solid black;
    position: absolute;
    right: 0;
    top: 0;
}
```

El **<li>** de **Nuevo +** se sale del **nav** y se coloca en una esquina del **body**. Esto se debe a que nuestro **nav** no tiene un **position relative**.

Aplicar diseño

Menú. Nuevo+

Damos **border** a **#nuevo_nav** para ver cómo lo posicionamos.

```
nav ul #nuevo_nav
{
    position: absolute;
    right: 0;
    top: -0.4em; /* Para centrar el botón */
}

nav ul #nuevo_nav a
{
    background: #F16028;
    color: white;
    padding: 0.8em; /* Para centrar el botón */
}
```

Aplicar diseño

Menú. Nuevo+

Damos **border** a **#nuevo_nav** para ver cómo lo posicionamos.

```
nav ul #nuevo_nav a
{
    background: #F16028;
    border-radius: 0.25em; /* Redondeamos esquinas */
    color: white;
    padding: 0.8em 0.5em;
    text-decoration: none; /* Elimina el subrayado */
    text-shadow: rgba (0,0,0,0.5) 2px 2px 5px;
}
```

Los valores de colores van de 0 a 255.

Alpha tiene un rango de 0 a 1, donde 0 es 100% transparente y 1, 100% opaco.

Aplicar diseño

footer

Además de las clases, existen las **pseudo clases**

```
footer
{
    background: #474747;
    color: #FBBF16;
    padding: 1em;
    text-align: center;
}
footer p:first-child /* Pseudo clase */
{
    margin-bottom: 0;
}
footer p:last-child /* Pseudo clase */
{
    margin-top: 0;
}
```

Aplicar diseño

footer / before + after

Agregamos contenido reusable con pseudo clases con **after** y **before.**

```
footer p:first-child:after /* Después de */
{
    content: " -";
}
footer p:first-child:before /* Antes de */
{
    content: "- ";
}
```

Esta es una técnica avanzada. Se lo aplicamos al **last-child** encadenando pseudo elementos.

Aplicar diseño

article

Estudiemos el código html del article:

```
<section id="contenido">
    <article class="item">
        <figure class="imagen_item"><img src="foto.jpg"></figure>
        <h2 class="titulo_item">
            <a href="#">Titulo del articulo</a>
        </h2>
        <div class="autor_item">
            por <a href="#">Marcelo Tinelli</a>
        </div>
        <div class="info_item">
            <a href="#" class="tag_item">Categoria</a>
            <span class="fecha_item">5 min</span>
            <a href="#" class="comentario_item">12</a>
            <a href="#" class="favorito_item"></a>
        </div>
        <div class="votacion">
            <a href="#" class="up"></a>212
            <a href="#" class="down"></a>
        </div>
    </article>
</section>
```

Aplicar diseño

article

En HTML entonces:

```
<section id="contenido">
    <article class="item">
        <figure class="imagen_item"><img src="foto.jpg"></figure>
        <h2 class="titulo_item">
            <a href="#">Titulo del articulo</a>
        </h2>
        <div class="autor_item">
            por <a href="#">Marcelo Tinelli</a>
        </div>
        <div class="info_item">
            <a href="#" class="tag_item">Categoria</a>
            <span class="fecha_item">5 min</span>
            <a href="#" class="comentario_item">12</a>
            <a href="#" class="favorito_item"></a>
        </div>
        <div class="votacion">
            <a href="#" class="up"></a>212
            <a href="#" class="down"></a>
        </div>
    </article>
</section>
```

En CSS entonces :

```
#contenido

#contenido .item

#contenido .item .imagen_item
#contenido .item .imagen_item img

#contenido .item .titulo_item
#contenido .item .titulo_item a

#contenido .item .autor_item

#contenido .item .info_item
#contenido .item .info_item .tag_item
#contenido .item .info_item .fecha_item
#contenido .item .info_item .comentario_item
#contenido .item .info_item .favorito_item

#contenido .item .votacion
```

Importante

Orden del código CSS

1. Enunciar las etiquetas
2. Los id
3. Las clases

Entre sí, ordenarlas alfabéticamente.

Aplicar diseño

article

Ordenado alfabéticamente en CSS :

#contenido
#contenido .item
#contenido .item .autor_item
#contenido .item .imagen_item
#contenido .item .imagen_item img
#contenido .item .info_item
#contenido .item .info_item .fecha_item
#contenido .item .info_item .tag_item
#contenido .item .titulo_item
#contenido .item .titulo_item a
#contenido .item .votacion
#contenido .item .votacion .comentario_item
#contenido .item .votacion .favorito_item