Posicionamiento y Layout (Parte I)

Introducción

Ya conociendo el proceso que conlleva crear una maqueta, ahora profundizaremos en algunos conceptos importantes al momento de construir maquetas que representen fielmente el diseño presentado en una representación visual.

En esta unidad conoceremos algunas propiedades que nos ayudarán a controlar las dimensiones, bordes, márgenes y relleno de una caja o contenedor y los tipos de cajas que podremos usar. También aprenderemos a posicionar elementos dentro de un diseño usando reglas CSS de posicionamiento y cambiaremos el flujo de los elementos usando flotaciones.

Conociendo el modelo de cajas o “box model”

Competencias

- Entender cómo se renderiza el tamaño de un elemento en nuestro sitio web.
- Entender cómo cambiar la forma en que se renderizan por defecto los elementos html

Todo elemento html se representa por una caja (o rectángulo) que se compone por el contenido, padding, margin y border estas 4 partes le dan el tamaño real a nuestro elemento html. En la siguiente gráfica podemos ver esta representación:

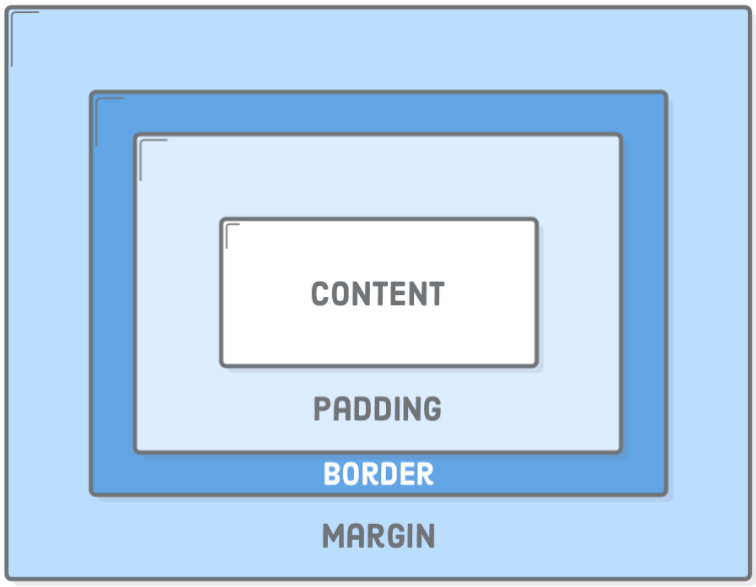


Imagen 1. Representación de componentes de un elemento Html.

Al incorporar un contenedor o caja en Html podemos apreciar propiedades por defecto, tal como lo es el box-sizing:conten-box, esta propiedad nos entrega la suma de los cuatro valores, contenido, padding, border y margin, tal como se explica en el siguiente ejemplo:

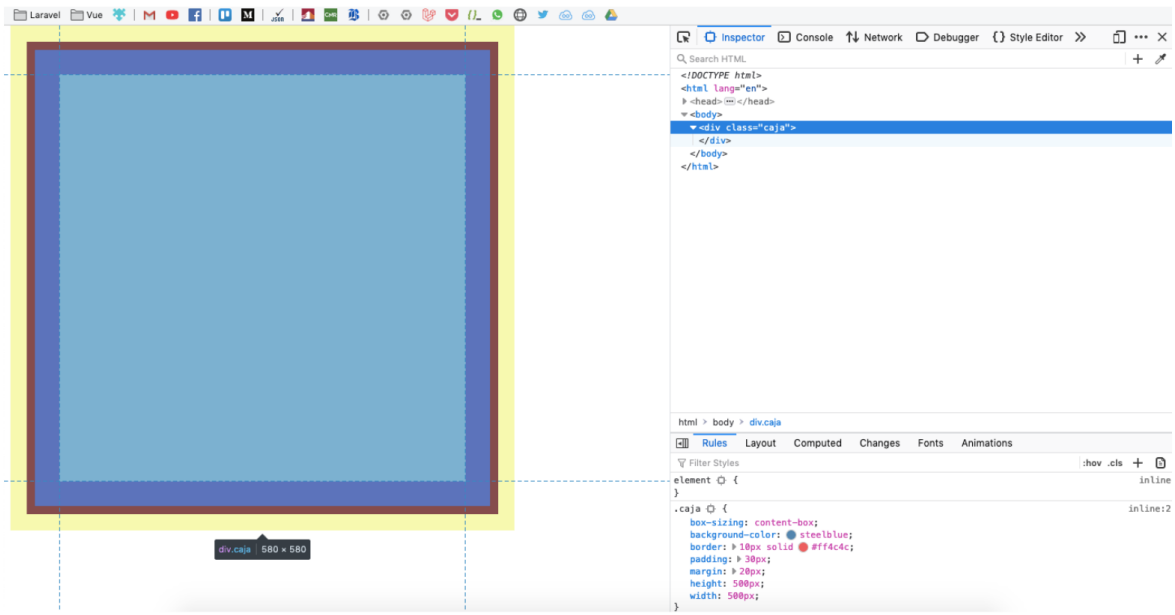


Imagen 2. Ejemplo de la propiedad box-sizing:content-box.

Una de las cualidades que tiene css3 es poder cambiar las propiedades por defecto que tiene un elemento html, por lo tanto continuando con el ejemplo anterior, si quisieramos que nuestro div o caja ocupe realmente el espacio asignado de 500px aplicamos la propiedad border-box; Esto nos permite mantener los valores entregados de margin, padding y border pero dentro de los 500px, no sumando valores que nos genere un problema en nuestro desarrollo a futuro. A continuación podemos ver ejemplificado lo mensionado anteriormente:

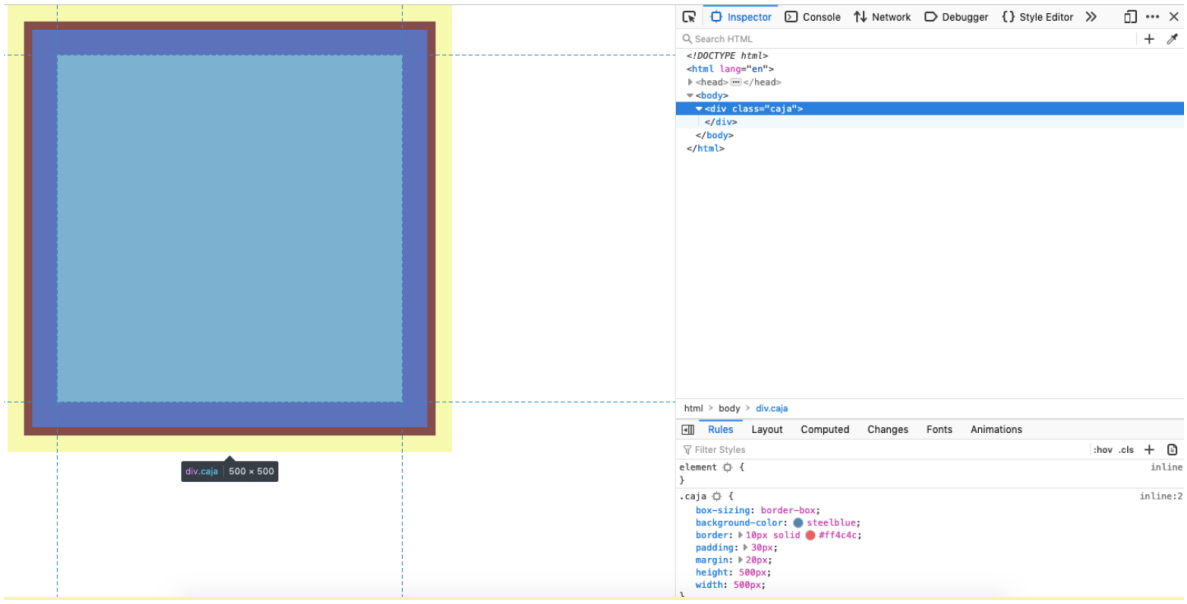


Imagen 3. Ejemplo de la propiedad box-sizing:border-box.

En la imagen (4) siguiente, entregada por el inspector de elemento, podemos apreciar los valores de el modelo caja, 420px x 420px asignado al contenido, con un padding de 30px en todas sus direcciones, 10px para los borders y 20px de margin tambien en todas sus direcciones, top, right, bottom y left:

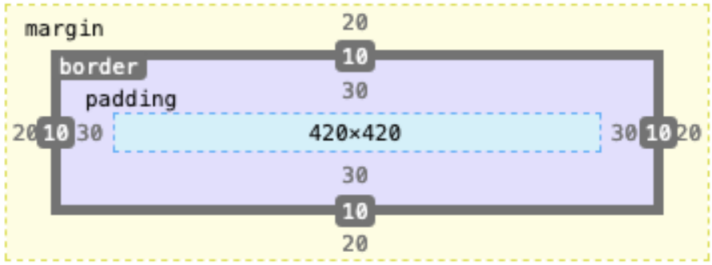


Imagen 4. Representación del modelo de caja en una caja de 500px.

Si la propiedad box-sizing no es soportada en el navegador se debe utilizar vendor prefixes (-moz, -webkit, -ms, -o), los prefijos de navegador son palabras que se anteponen a un propiedad de css para que esta pueda ser leída o interpretada por el navegador correspondiente. Ejemplo del uso de prefijos:
• -webkit-box-sizing: border-box; • -moz-box-sizing: border-box; • -o-box-sizing: border-box; • -ms-box-sizing: border-box

Prefijos de Navegador

Los vendor prefixes son una forma en que los navegadores usan para darnos acceso a los desarrolladores de CSS a funciones más nuevas que aún no se consideran estables.

Los prefijos utilizados para los distintos navegadores son:

```
•  webkit- (Chrome, Safari, iOS Safari / iOS WebView, Android)
• -moz- (Firefox)
• -ms- (Edge, Internet Explorer)
• -o- (Opera, Opera Mini)
```

Ejemplo de Prefijo:

```
.example {
 -webkit-box-sizing: border-box;
         box-sizing: border-box;
}
```

Existen herramientas online que nos facilitan la creación de los prefixes tal como autoprefixer y tablas de soporte de navegador tal como "Can I use", que actualiza constantemente soporte sobre font-end en navegadores web de escritorios y dispositivos móviles.

Autoprefixer online: https://autoprefixer.github.io/

Can I use: https://caniuse.com/

Tipos de cajas

Competencias

- Entender las diferentes maneras en que un elemento html es desplegado en nuestro sitio web.
- Comprender las diferencias entre un elemento en línea y bloque.

Los elementos que componen un documento HTML, por lo general, tienen un comportamiento que los define dentro del flujo del HTML. Existen 2 tipos de cajas o elementos html, los elementos en línea y los elementos de bloque.

Elementos de bloque

Los elementos de bloque siempre comenzarán en una nueva línea y ocupan todo el espacio disponible de la página, siempre y cuando no se defina un ancho para este elemento.

Algunos ejemplos de elementos de bloque son las etiquetas: `

```
- <li>
- <h1>
- <p>
- <div>
```

Elementos en línea

Por otra parte, existen otros elementos que se comportan diferente a los elementos de bloque los cuáles son llamados elementos inline. Estos se caracterizan por utilizar sólo el espacio necesario para mostrar su contenido.

Alguno de los elementos HTML que tienen este comportamiento son las etiquetas:

- < img>
- < span>
- < a>
- < i>

Cambiando el comportamiento de los elementos

Display

Con la propiedad display podremos cambiar el comportamiento de los elementos a modo de especificar el tipo de comportamiento que deseamos que tenga este dentro de un layout.

En palabras simples, con esta regla podremos cambiar el estado de los elementos de línea a bloque y viceversa dándonos el poder absoluto para definir la disposición de los elementos dentro de nuestra maqueta.

Algunos de los valores que puede tomar la propiedad display son:

- Display:block

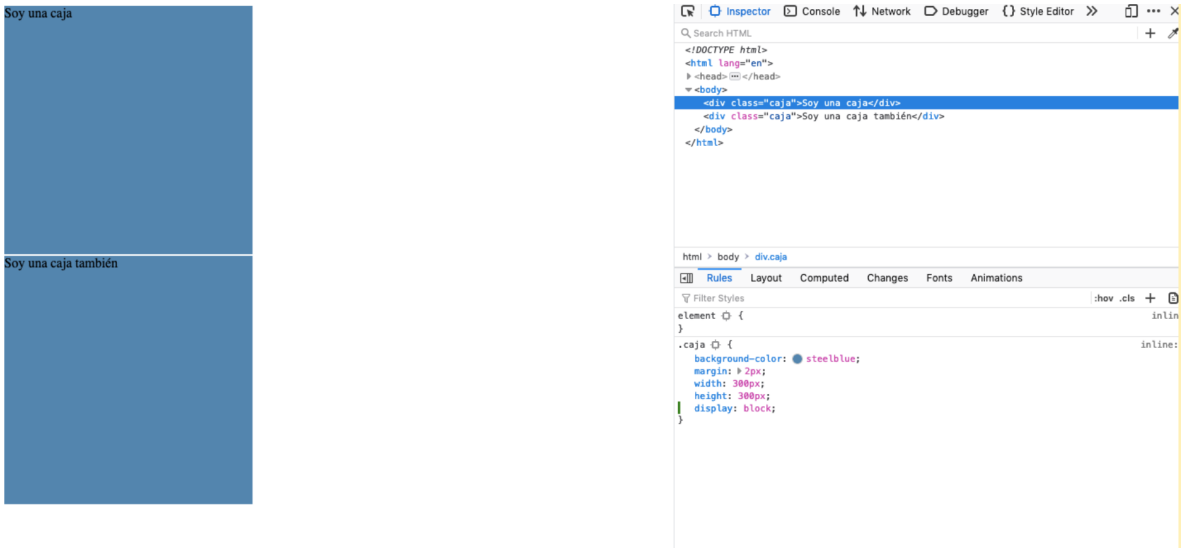


Imagen 5. Ejemplo de la propiedad display:block.

Al ser elementos en bloque por defecto (se agregó a modo visual, ya que no es necesario) se distribuyeron según el orden escrito en nuestro html (forma vertical) y las cajas respetan el alto y ancho indicado en nuestra clase css.

- Display:inline

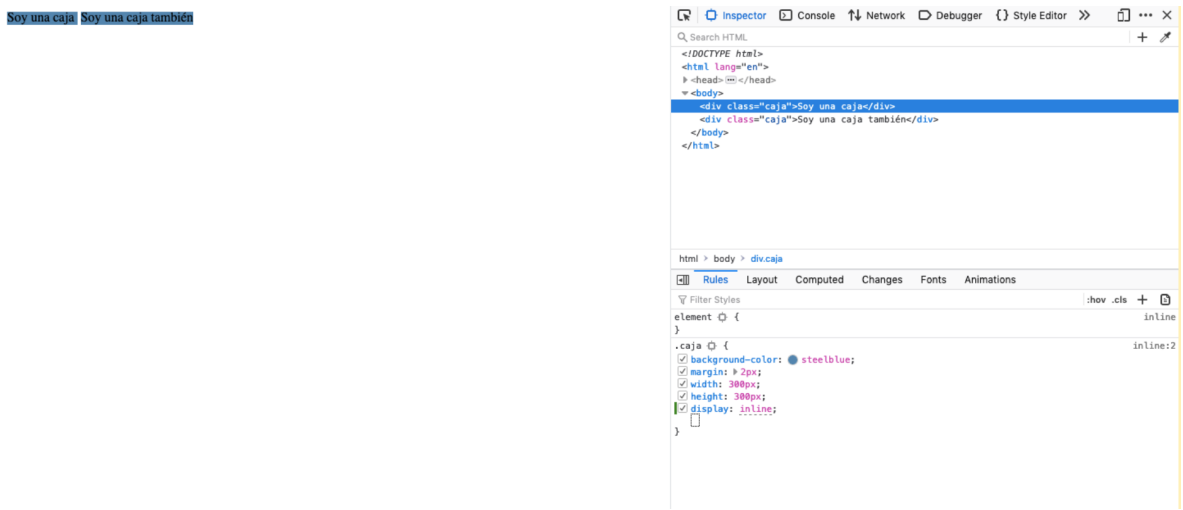


Imagen 6. Ejemplo de la propiedad display:inline.

Al cambiar la propiedad display:block a display:inline pasaron 2 cosas:

```
• La distribución de los elementos en la pantalla sera de manera horizontal
(uno al lado del otro)
• Las cajas no respetaran el alto y ancho definido de 500px, quedando como
tamaño el mínimo necesario por su contenido.
```

- Display:inline-block

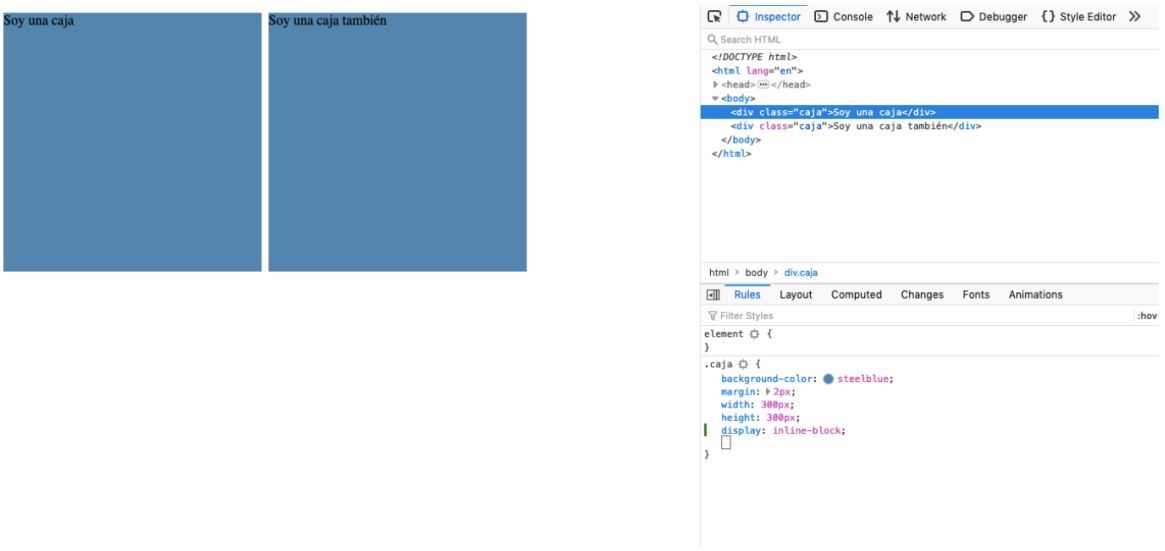


Imagen 7. Ejemplo de la propiedad display:inline-block.

El valor de inline-block en la propiedad display saca lo mejor del display:inline como el de block. Ya que por un lado nos permite que nuestros elementos no se comporten como bloque y se realice un despliegue horizontal, pero si permita que las cajas tomen un alto y un ancho previamente indicado.

Esta propiedad es muy utilizada al momento de realizar una maqueta html y css ya que gran parte de los diseños proporcionados tienen elementos distribuidos de forma horizontal, por ejemplo el menú de un sitio web.

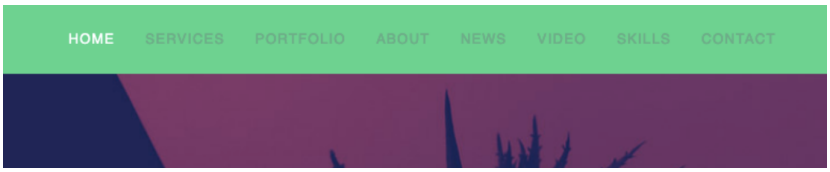


Imagen 8. Ejemplo de la propiedad display:inline-block, puede revisar en https://technext.github.io/boxus/ .

Consejo: siempre recordar que cuando se requiera alinear elementos utilizando la propiedad display:inline-block, todos los elementos que se desea alinear o que tengan un despliegue horizontal deben presentar esta propiedad.

- Display:none

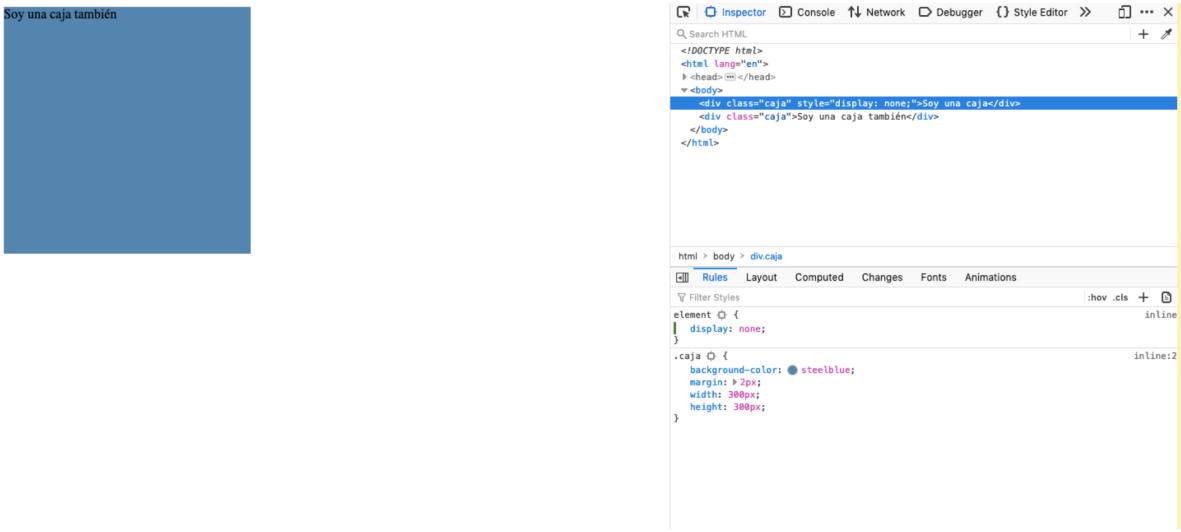


Imagen 9. Ejemplo de la propiedad display:none.

Esta propiedad elimina visualmente un elemento que se encuentre dentro de nuestra maqueta, no obstante, este no desaparece del código, o sea que sí abrimos el inspector de elementos veremos que el elemento aún se encuentra ahí.

Por otra parte, existe una propiedad que podremos usar para esconder un elemento sin necesidad de eliminar el elemento desde la maqueta usando visibility.

Visibility

La propiedad visibility es otra propiedad que nos ayudará a esconder los elementos que no queramos mostrar hacia los usuarios dentro de la interfaz.

Al usar esta regla el elemento se esconderá dejando un espacio en donde este se encontraba.

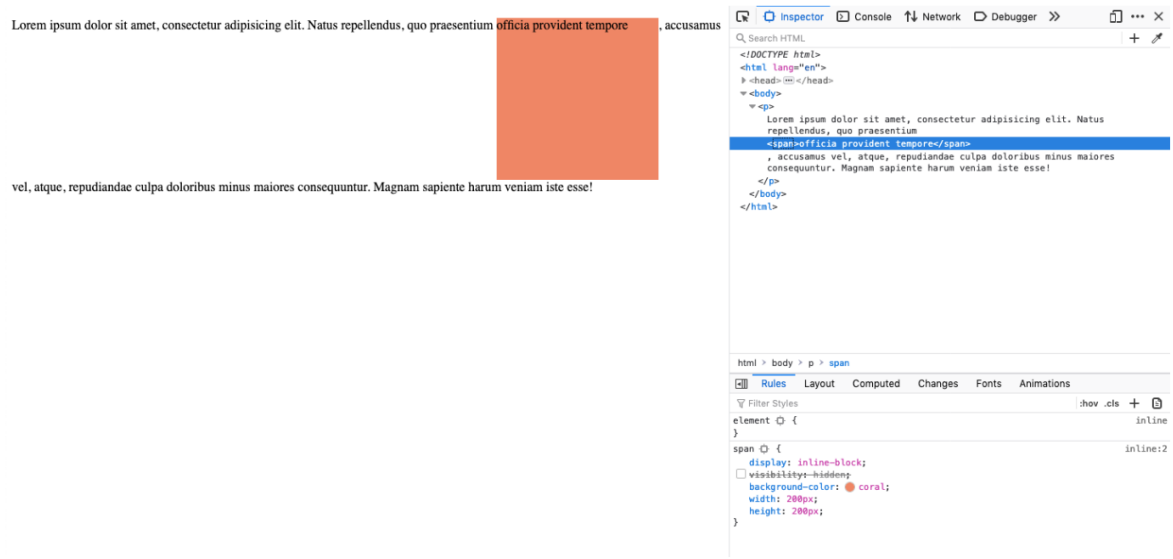


Imagen 10. Ejemplo de un texto en donde se aplica visibility:hidden en el span.

Esta propiedad agrega al párrafo con el `<span>` un `visibiity:hidden;`, el resultado es el siguiente:

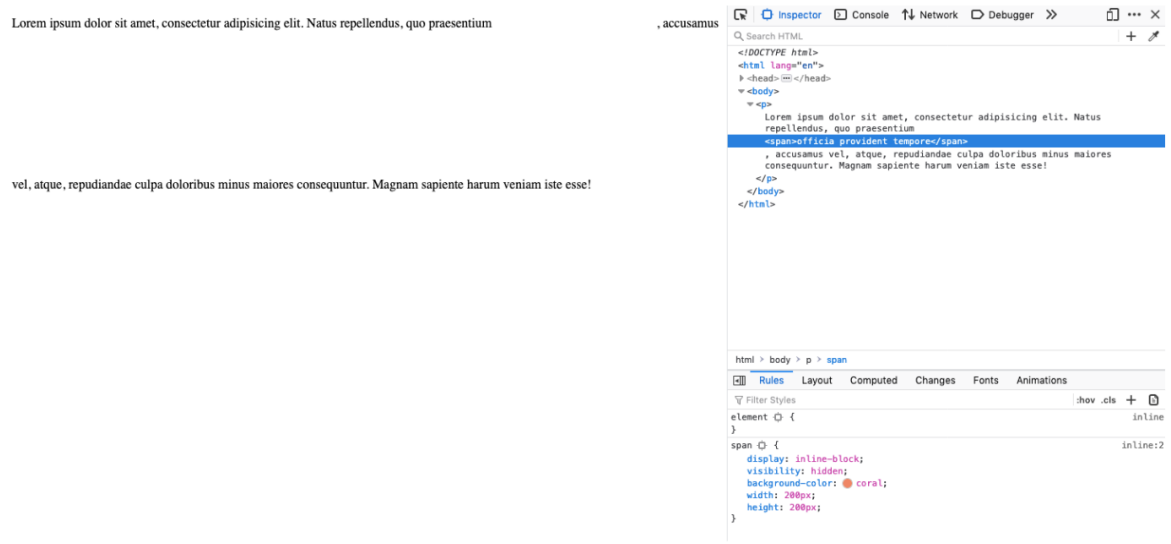


Imagen 11. Resultado luego de aplicar visibility:hidden en el span.

El elemento desapareció, pero mantuvo el espacio asignado. Si se quisiera mostrar el contenido usando esta propiedad, el valor es `visiblity: visible;`

¿Cuál es la diferencia entre visibility: hidden y display: none?

Estos dos valores, tienen una gran diferencia la que radica en el hecho que la propiedad `visibility: hidden;` esconde los elementos manteniendo el espacio que utilizaban y `display: none;` los elimina del documento y re asigna el espacio a otro elemento (ver imagenes 9 y 11).

Tipos de posicionamiento

Competencias

- Conocer las diferentes formas de posicionar un elemento en nuestro layout.
- Aprender a cambiar el posicionado por defecto de un elemento html.

Por defecto Html posiciona un elemento de forma estática (static); no está posicionado de ninguna manera especial; siempre se posiciona de acuerdo con el flujo normal de la página

El posicionamiento dentro de un sitio web es clave a la hora de programar, es por esto que profundizaremos en los distintos tipos de posiciones y en donde aplicarlas.

Alineando texto

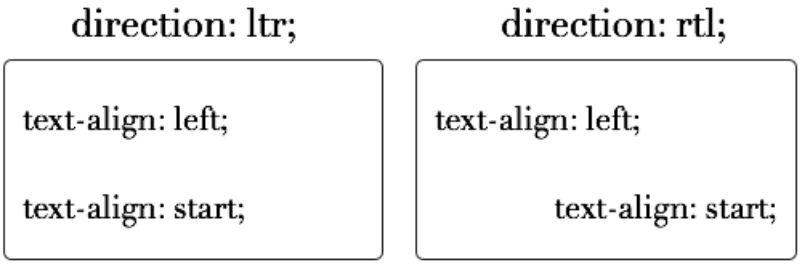
La propiedad text-align en css especifica la alineación horizontal del texto en un elemento. Los valores tradicioanles para alinear texto a traves de la propiedad text-align son:

- left: el valor predeterminado. El contenido se alinea al lado izquierdo.
- right: el contenido se alinea al lado derecho.
- center: el contenido se centra entre los bordes izquierdo y derecho. El espacio en blanco en los lados izquierdo y derecho de cada línea debe ser igual.
- justify: el contenido se espacia de manera tal que se puedan colocar tantos bloques en una línea como sea posible y la primera palabra en esa línea está a lo largo del borde izquierdo y la última palabra a lo largo del borde derecho.
- inherit: el valor será el que sea el elemento principal.

Si bien la propiedad indica que la alineación es para el texto, tiene la particularidad de afectar a todos los elementos en linea o bloque dentro de un contenedor, por ejemplo una imagen.

También hay dos nuevos valores en CSS3, inicio (start) y fin (end). Estos valores facilitan la compatibilidad con varios idiomas. Por ejemplo, el inglés es un idioma de izquierda a derecha (ltr) y el árabe es un idioma de derecha a izquierda (rtl). Usar "derecha" e "izquierda" para los valores es demasiado rígido y no se adapta a medida que cambia la dirección. Estos nuevos valores se adaptan:

```
start: igual que "left" en ltr, igual que "right" en rtl.
end: igual que "right" en ltr, igual que "left" en rtl.
```



Posicionando Elementos

Por defecto los navegadores utilizan position: static;, de modo que todos los elementos vistos en los ejemplos anteriores se encuentran estáticos, esto quiere decir que los elementos se renderizan de acuerdo al flujo normal del HTML. Asimismo, cualquier elemento el cual no se especifique su posición será static.

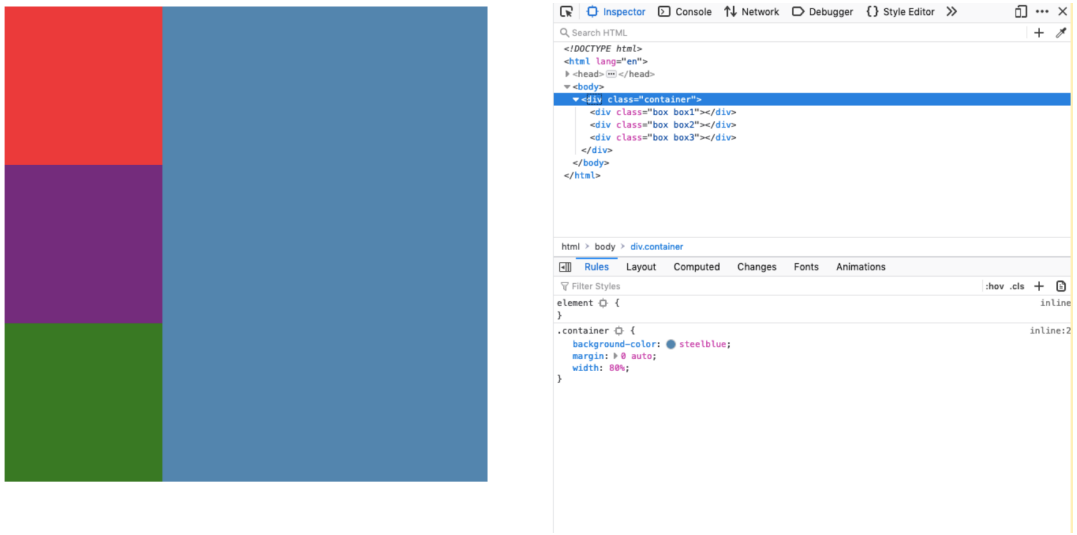


Imagen 12. Ejemplo de la propiedad position:static.

Posición relativa (position: relative)

Con los valores relativos podemos desplazar un elemento en el *HTML* respecto a su posición original establecida, que como lo señalamos anteriormente es estatica (static). Este desplazamiento lo podemos controlar con las propiedades:

- top
- Right
- Bottom
- left

En el siguiente ejemplo podemos apreciar que al agregar la propiedad position con el valor relative de 300px left, desplaza 300px a la izquierda nuestro elemento desde el punto original:

Imagen 13. Ejemplo de la propiedad position:relative.

Los valores para desplazar pueden ser distintos a los pixeles, estos pueden definirse en %, em, rem, vw y vh, dependiendo del dispositivo que estemos usando y requerimiento establecidos en el proyecto de forma inicial.

Posición absoluta (position:absolute)

Los valores absolutos nos permiten posicionar elementos en función a un elemento padre, siempre y cuando este tenga una posición distinta a static. En el caso de ser así el elemento con posición absoluta tomará como padre al html y no al viewport.

Los elementos con posiciones absolutas se encuentran fuera del flujo normal del HTML, de modo que este no afecta a otros elementos posicionados en el documento.

Para situar el contenido en estas posiciones usaremos las propiedades top, right, bottom y left.

Aplicándo positon:absolute a la segunda caja (purple), el resultado simula que la caja 3 (green) desapareció, pero esto es sólo un efecto visual ya que la caja 3 (green) se encuentra detrás de la caja 2 (purple).

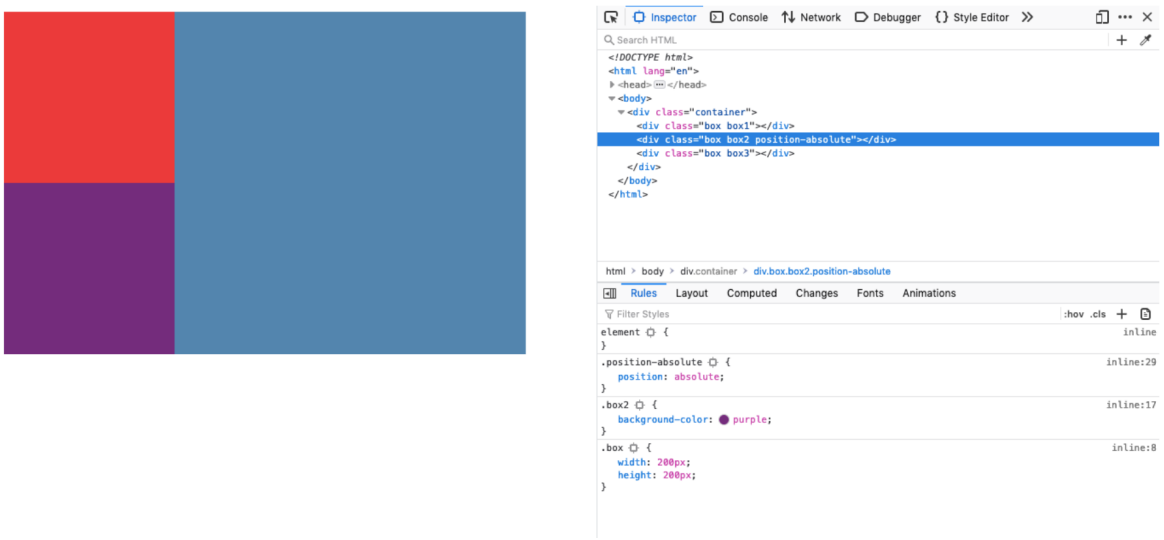


Imagen 14. Ejemplo de la propiedad position:absolute sobre segundo elemento.

En la siguiente imagen se aplica un left de 300px a nuestra caja absoluta (purple), descubriendo la caja 3 (green).

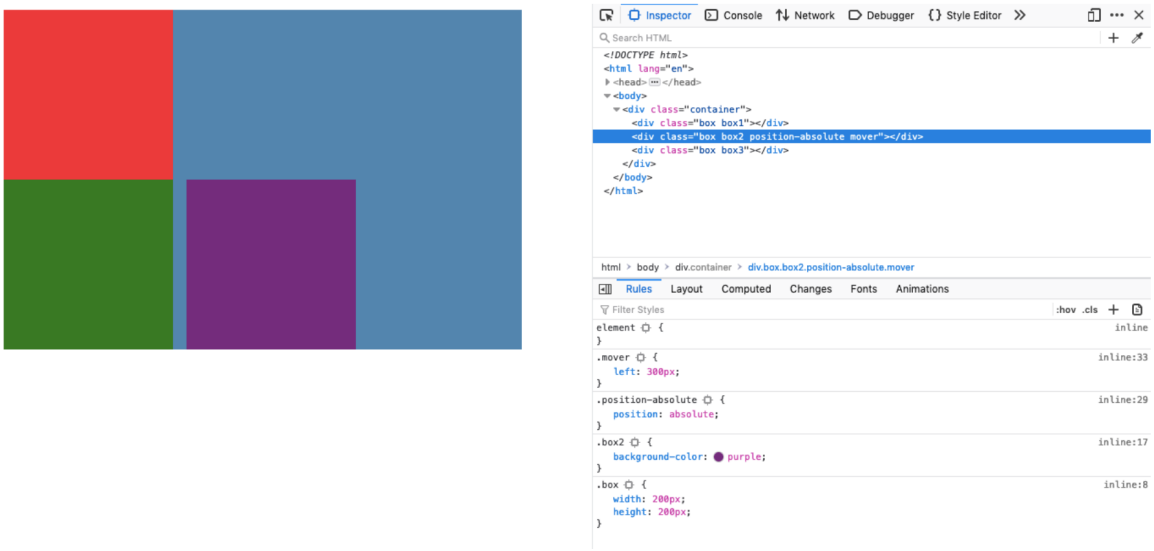


Imagen 15. Resultado de aplicar left:300px al elemento con position:absolute

Posición fija (position:fixed)

En el posicionamiento fijo el elemento se elimina del flujo del documento. Los elementos de posición fija siempre son relativos al documento, no a ningún padre en particular, y no se ven afectados por el desplazamiento. El posicionamiento fijo nos permite fijar contenido en función del viewport, o sea, de la ventana del navegador.

En el siguiente ejemplo, al aplicar en la caja 2 (purple) el valor de position:fixed y un right:0, el resultado sera que el div se fije al costado derecho en una posición 0 y al realizar scroll el div se quedara en la posicion asignada:

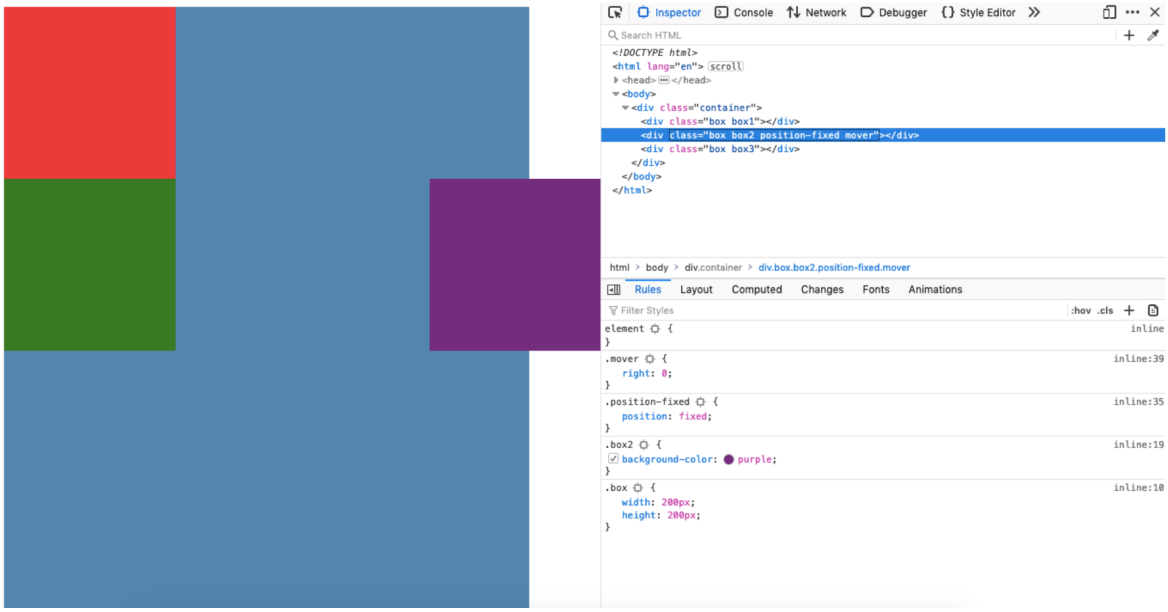


Imagen 16. Ejemplo de la propiedad position:fixed.

De igual manera es importante mencionar que esta posición no respeta el flujo normal del HTML, por lo tanto se superpondrá ante cualquier elemento anterior o posterior a ella.

¿Qué es z-index?

Esta propiedad define si los elementos estarán sobre o debajo de otros. Para hacerlo deberemos dar valores positivos para que estén arriba y negativos para que estén debajo. Importante recordar que la propiedad z-index, sólo funcionará en elementos que no tengan un posicionamiento diferente a estático.

Si revisamos el siguiente ejemplo, vemos que al aplicarle position absolute a las 2 cajas (purple y green), más valores negativos en z-idex -2 y -1 respectivamente, quedan por detras del h1 (Propiedad Z-index).

Imagen 17. Ejemplo de la propiedad z-index.

Estos son los valores usados en CSS y HTML:

CSS

```css
.box {
  position: absolute;
  left: 0px;
  top: 0px;
  z-index: -2;
  height: 150px;
  width: 150px;
  background-color: purple;
}

.box2 {
  position: absolute;
  left: 50px;
  top: 10px;
  z-index: -1;
  height: 150px;
  width: 150px;
  background-color: green;
}

h1{
  z-index: 1;
  font-family: monospace;
}
```

Html

```html
<!DOCTYPE html>
<html>
  <head>
  </head>
    <body>

      <h1>Propiedad Z-index</h1>

      <div class="box"></div>
      <div class="box2"></div>

    </body>
</html>
```

Imagen 18. Resultado de cambiar los valores de z-index de los elementos.

```css
.box {
  position: absolute;
  left: 0px;
  top: 0px;
  z-index: 2;
  height: 150px;
  width: 150px;
  background-color: purple;
}

.box2 {
  position: absolute;
  left: 50px;
  top: 10px;
  z-index: 1;
  height: 150px;
  width: 150px;
  background-color: green;
}

h1{
  z-index: -1;
  font-family: monospace;
}
```