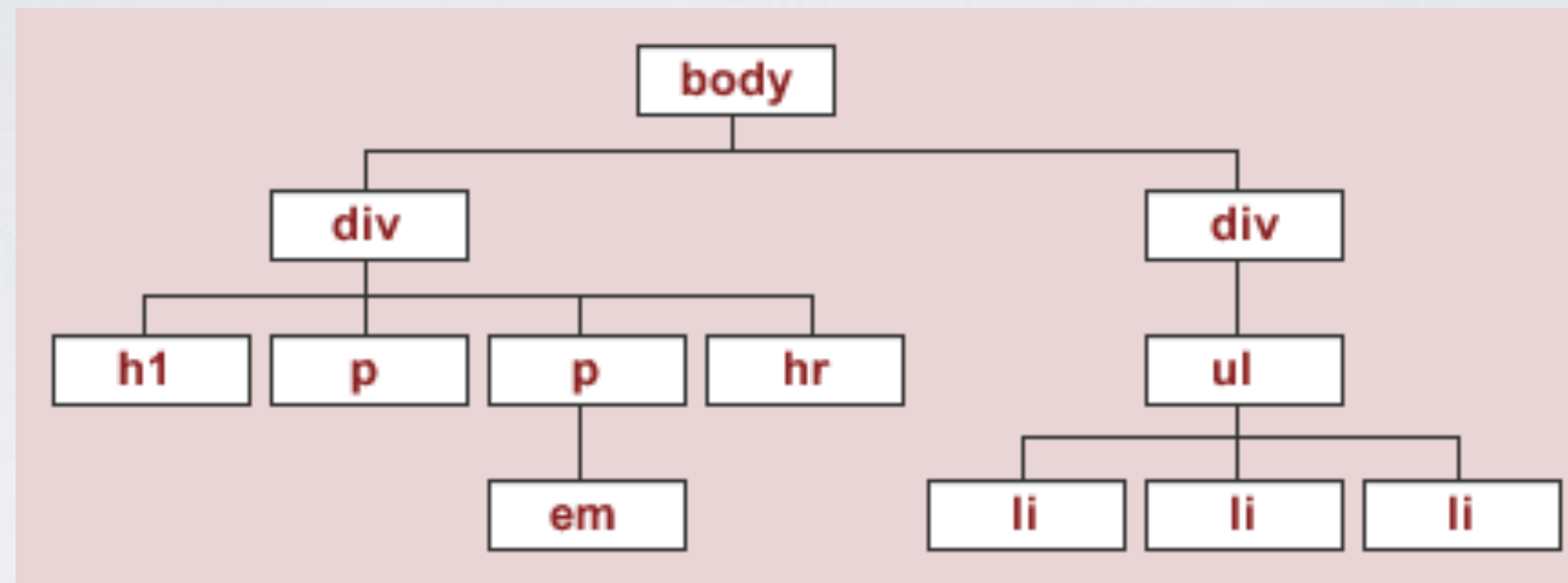


EN LAS SIGUIENTES CÁPSULAS DE  
VIDEO HABLAREMOS DE TAMAÑO Y  
POSICIONAMIENTO.

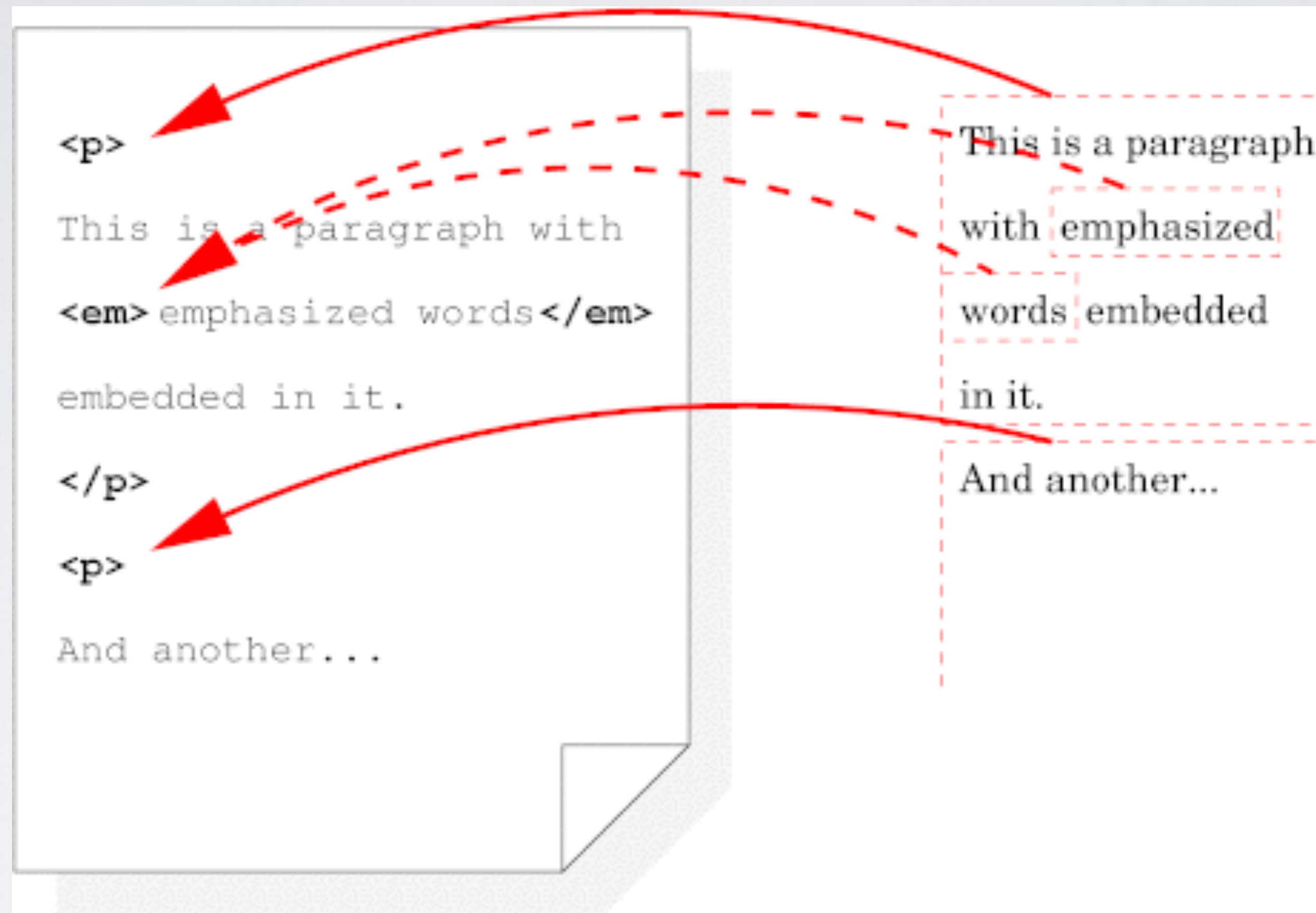


EN LAS SIGUIENTES CÁPSULAS DE  
VIDEO HABLAREMOS DE TAMAÑO Y  
POSICIONAMIENTO.





HTML asume que tu documento está modelado como un árbol de elementos



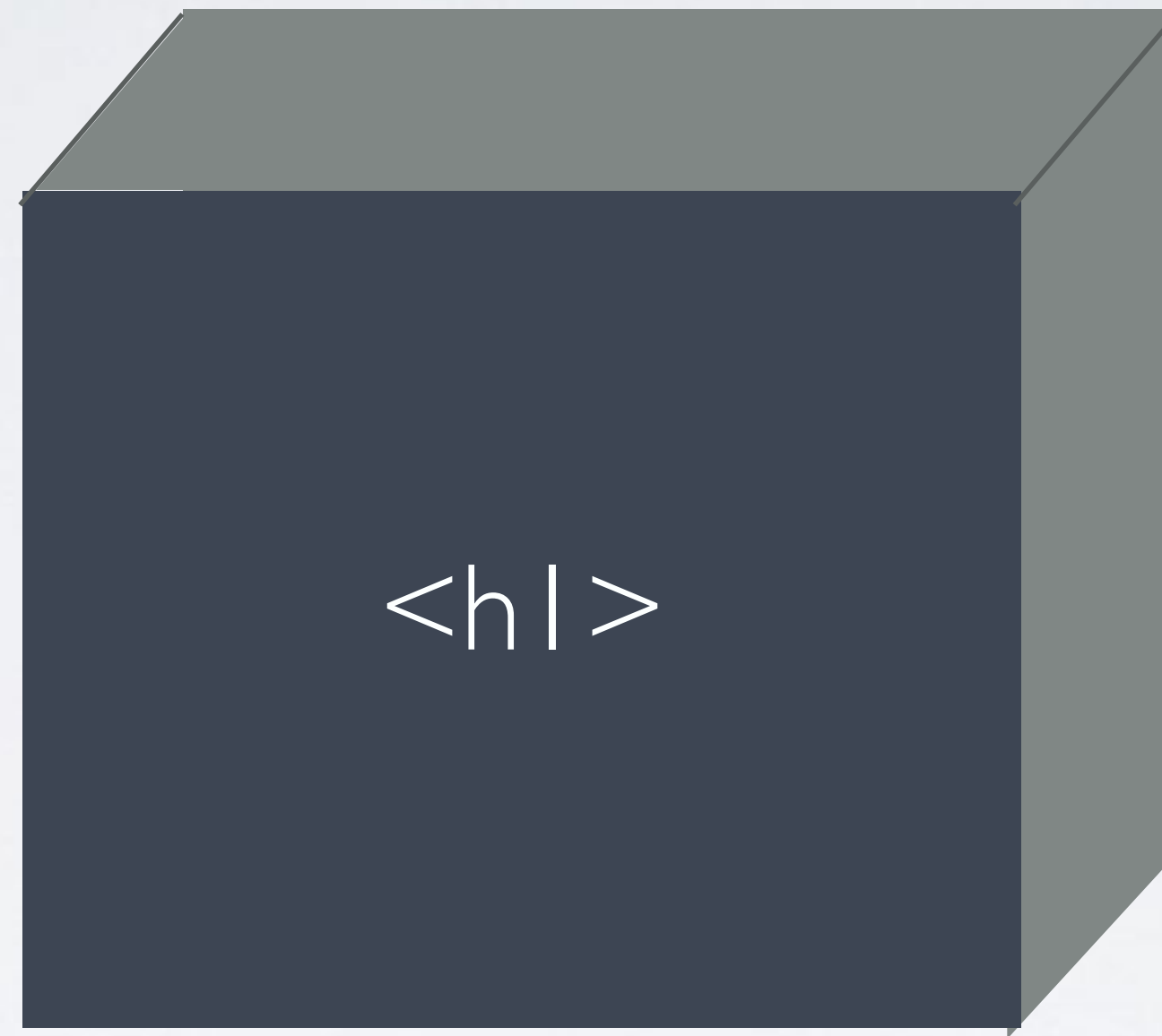
Para dibujarse cada uno de estos elementos se transforma en una caja que tiene un **tamaño** y una **posición**

El tamaño es cuanto espacio ocupa el elemento en nuestro sitio, y la posición tiene que ver donde se sitúa.

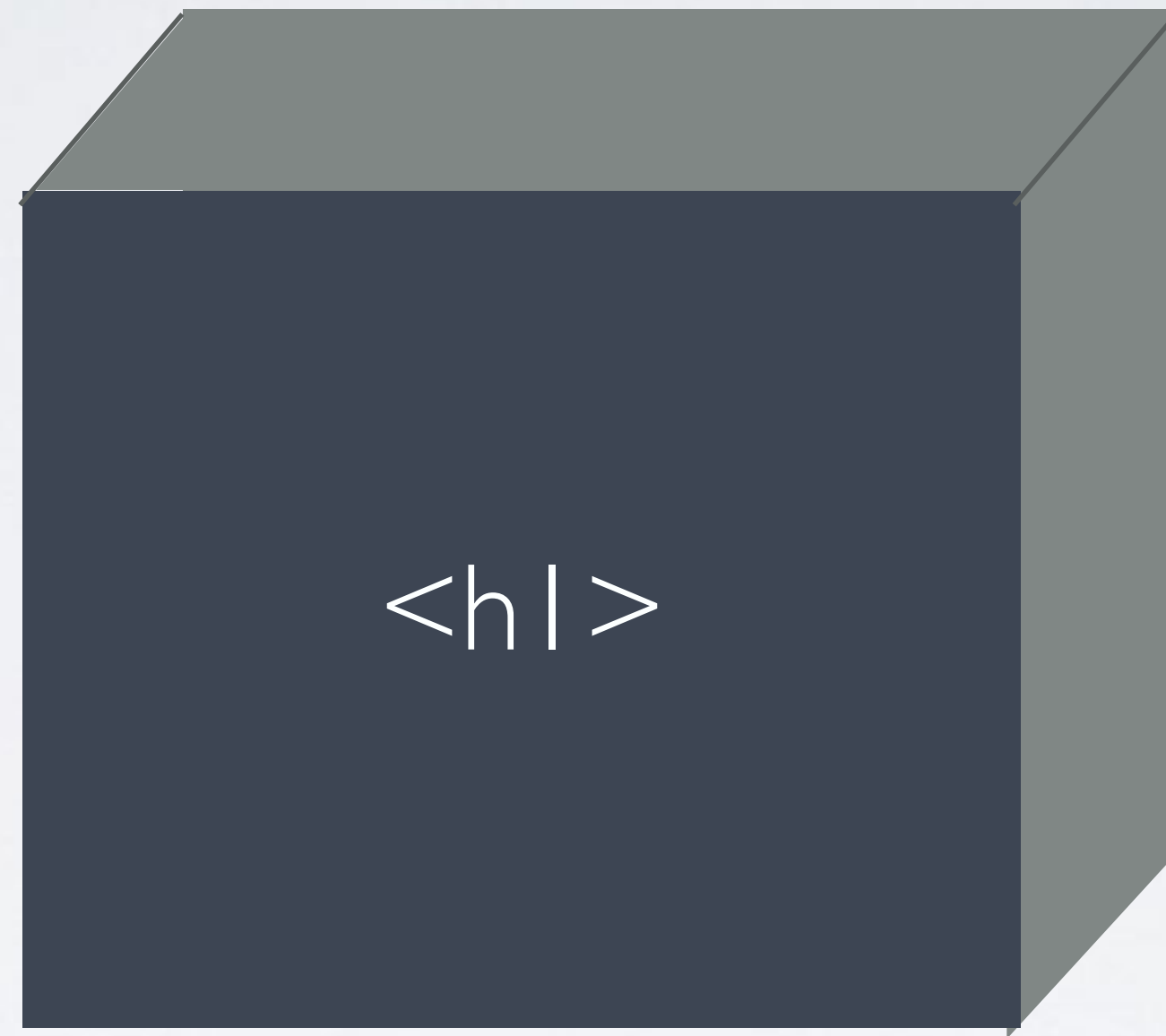


AHORA HABLAREMOS DEL TAMAÑO

# MODELO DE CAJAS.



# MODELO DE CAJAS.



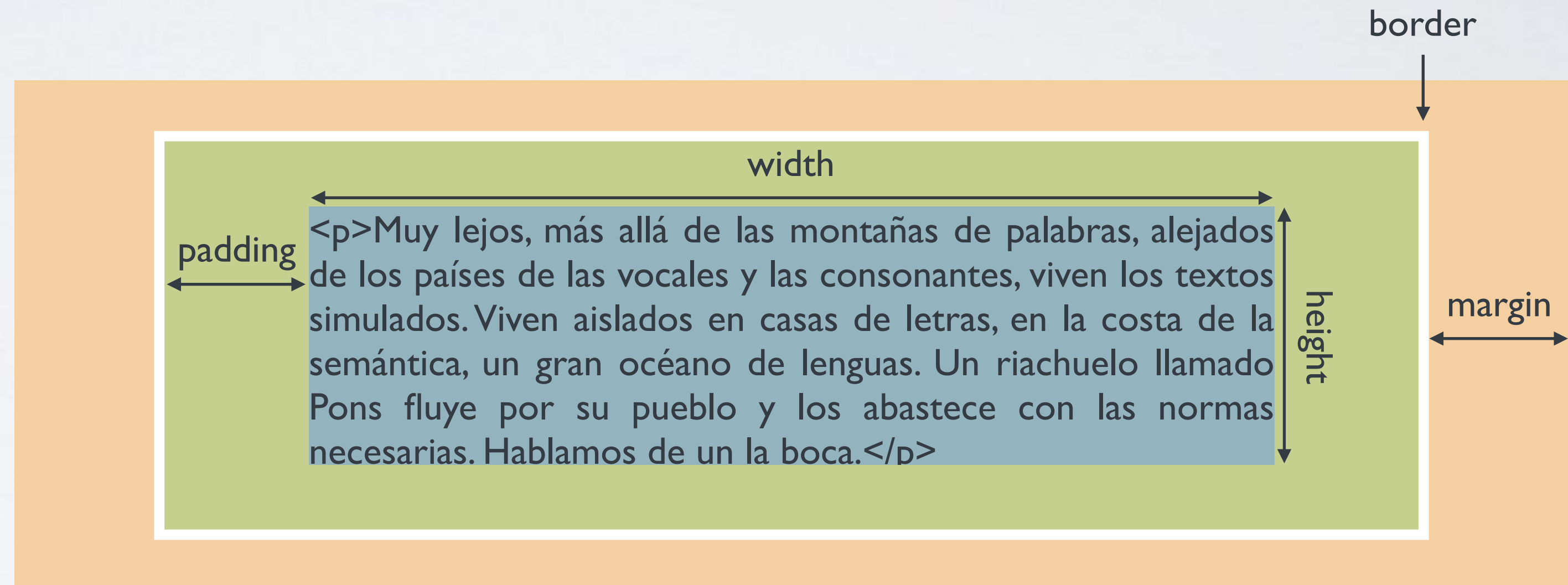
A manejar las propiedades asociadas a tamaño, bordes, márgenes y padding de los elementos.



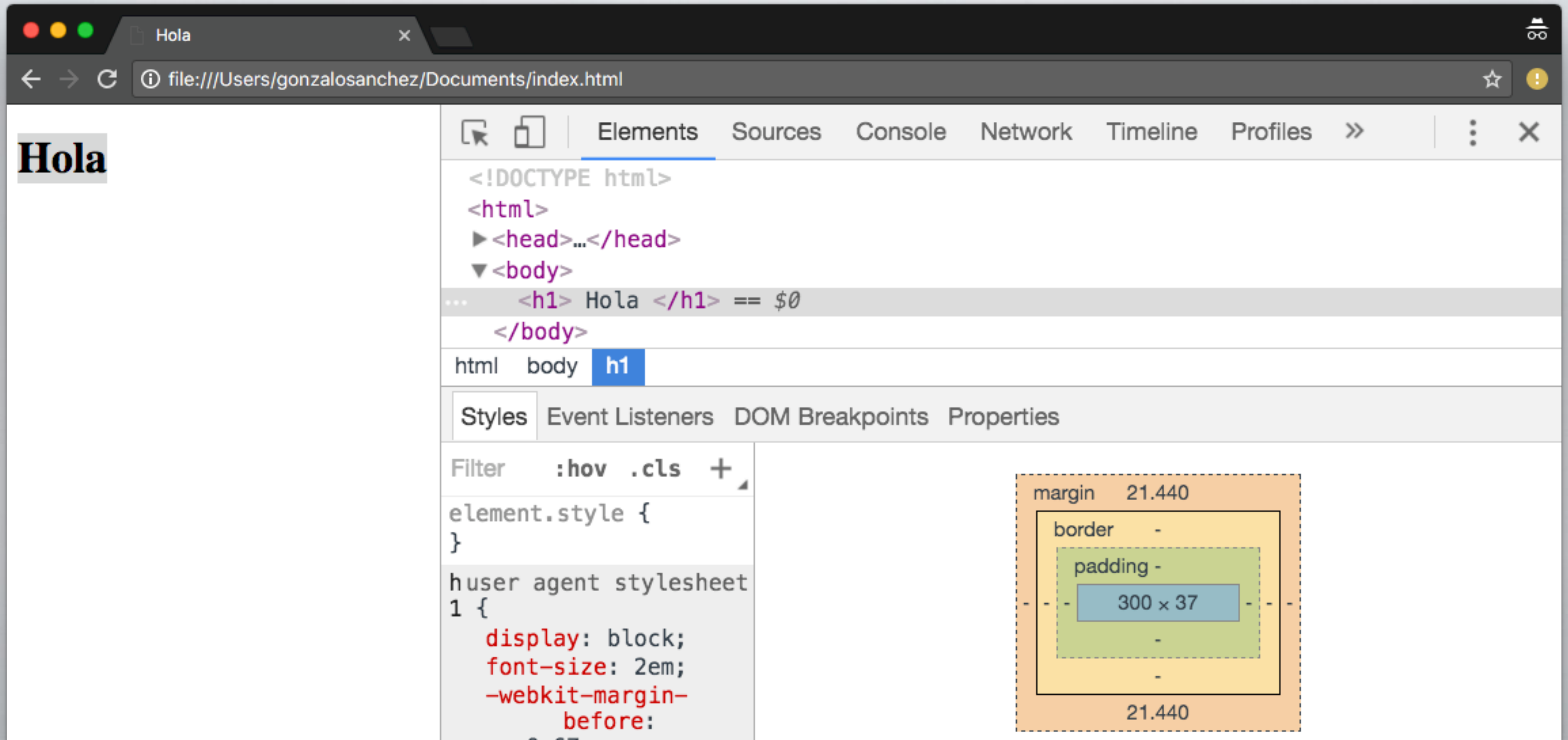
# BORDER BOX

El modelo de cajas implica que todos los elementos son una caja rectangular, cada una de las cuales tiene ciertas propiedades. Estas son:

- Width
- Height
- Border
- Padding
- Margin



# VEAMOS UN EJEMPLO EN ESPECÍFICO



The screenshot shows a web browser window with a single tab titled 'Hola'. The address bar displays the file path: `file:///Users/gonzalosanchez/Documents/index.html`. The page content is a large 'Hola' text.

The browser's developer tools are open, showing the 'Elements' panel. The DOM tree is expanded to the `h1` element, which contains the text 'Hola'. The breadcrumb path is `html > body > h1`.

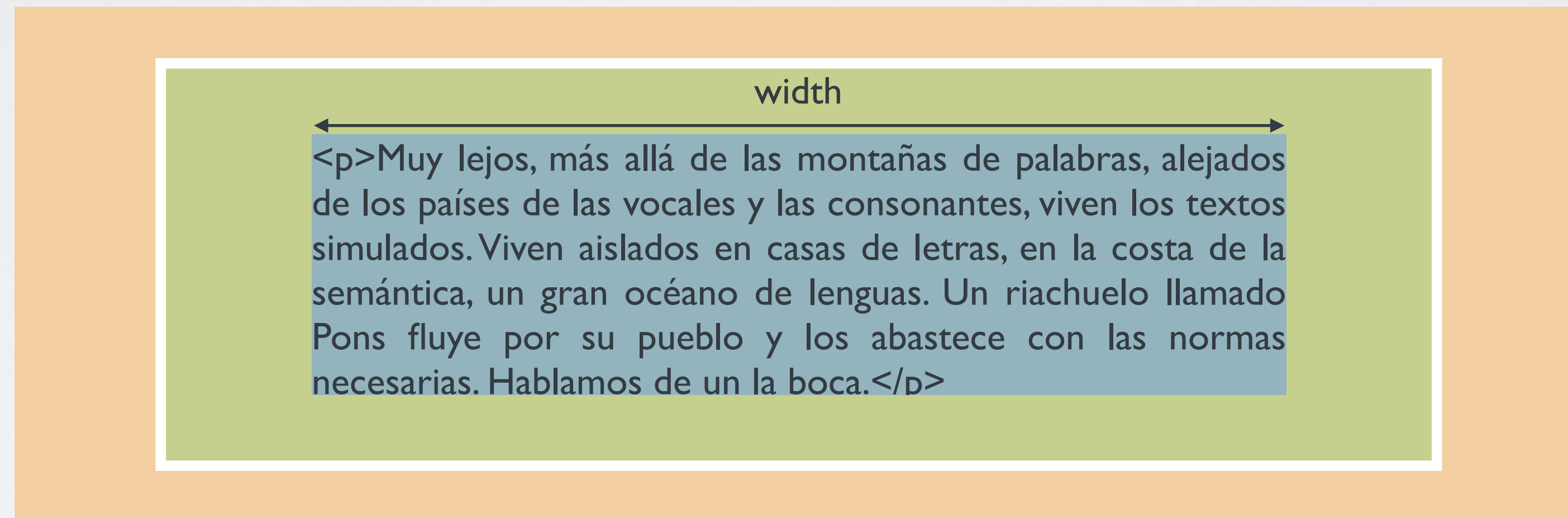
The 'Styles' panel is active, showing the default user agent styles for the `h1` element:

```
element.style {  
}  
  
huser agent stylesheet  
1 {  
  display: block;  
  font-size: 2em;  
  -webkit-margin-before:  
    0.67em;  
  margin-bottom: 1em;  
  margin-top: 1em;  
}
```

On the right side of the Styles panel, a box model diagram is displayed. It shows a blue box representing the content area with dimensions `300 x 37`. This is surrounded by a green box for padding, a yellow box for border, and an orange box for margin. The margin is labeled with `margin 21.440` on the top and bottom edges.

# PROPIEDAD WIDTH

Esta propiedad indica el ancho del elemento.



# PROPIEDAD HEIGHT

Establece la altura de los elementos

`<p>`Muy lejos, más allá de las montañas de palabras, alejados de los países de las vocales y las consonantes, viven los textos simulados. Viven aislados en casas de letras, en la costa de la semántica, un gran océano de lenguas. Un riachuelo llamado Pons fluye por su pueblo y los abastece con las normas necesarias. Hablamos de un la boca.`</p>`

height

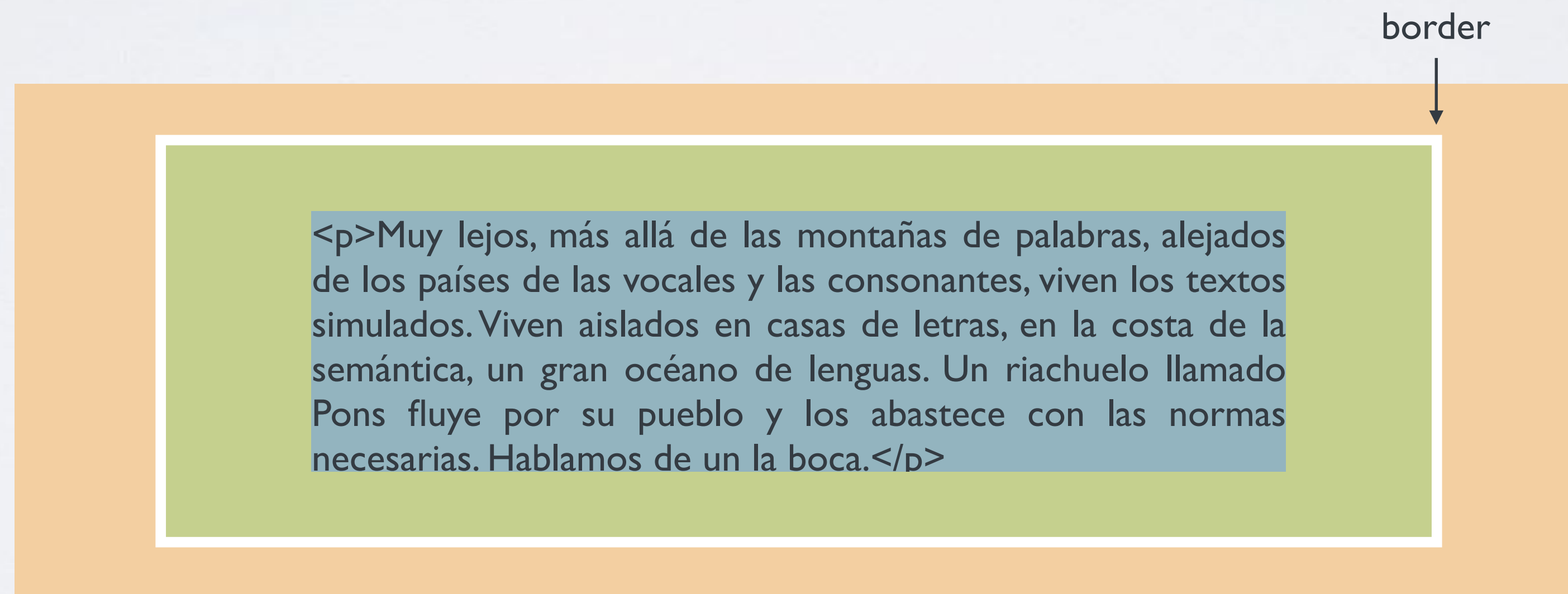


CREEMOS UN CUADRADO DE  
COLOR VERDE CON WIDTH Y HEIGHT



# PROPIEDAD BORDER

Permite especificar las características del borde de un elemento.



`<p>`Muy lejos, más allá de las montañas de palabras, alejados de los países de las vocales y las consonantes, viven los textos simulados. Viven aislados en casas de letras, en la costa de la semántica, un gran océano de lenguas. Un riachuelo llamado Pons fluye por su pueblo y los abastece con las normas necesarias. Hablamos de un la boca.`</p>`

# PROPIEDAD BORDER

Para agregar bordes a un elemento hay tres propiedades claves involucradas.

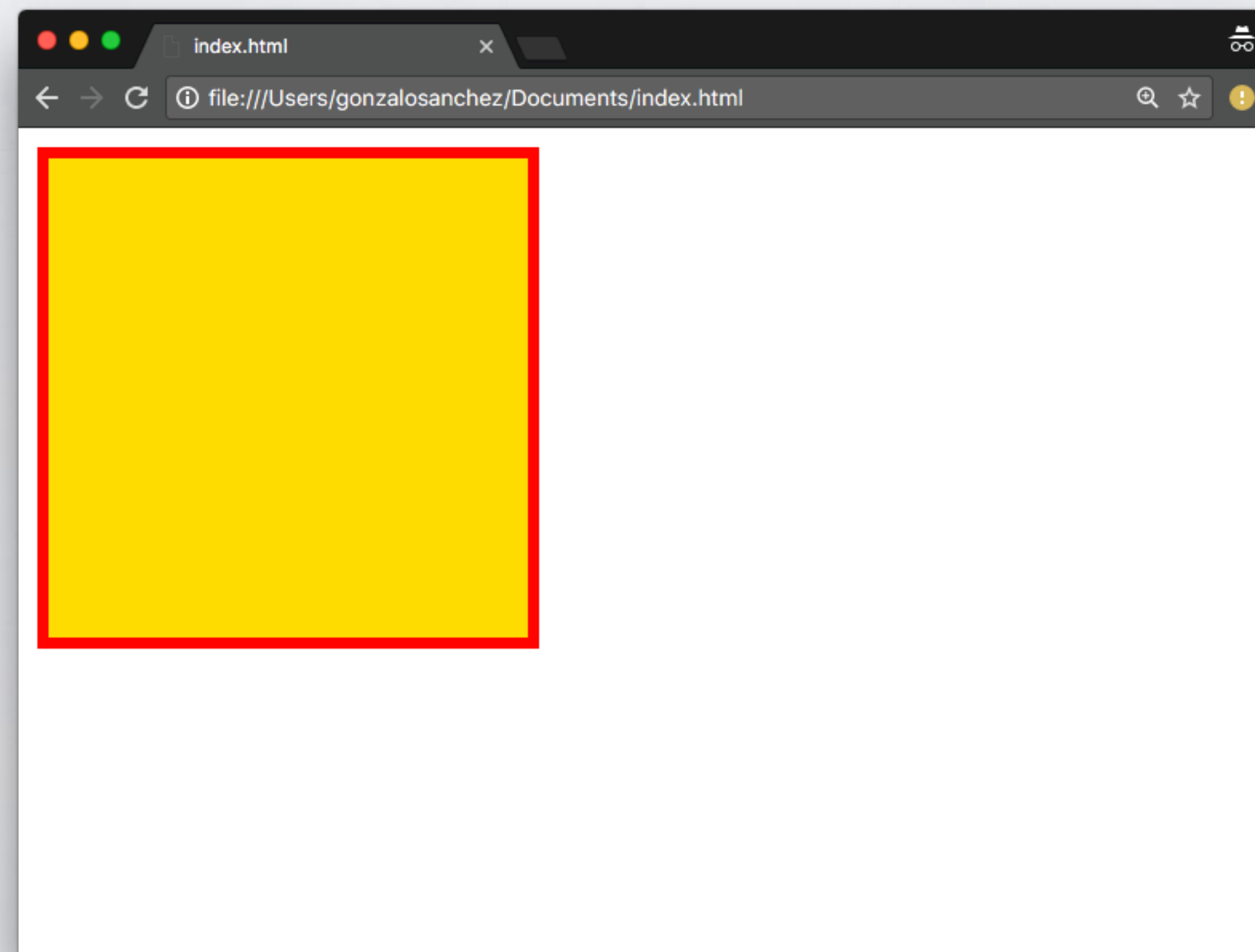
```
p {  
  border-style: solid;  
  border-width: 1px;  
  border-color: black;  
}
```

**border-style:** Estilo de borde (dotted, dashed, solid, etc).

**border-width:** Ancho de borde (px, cm, em, etc).

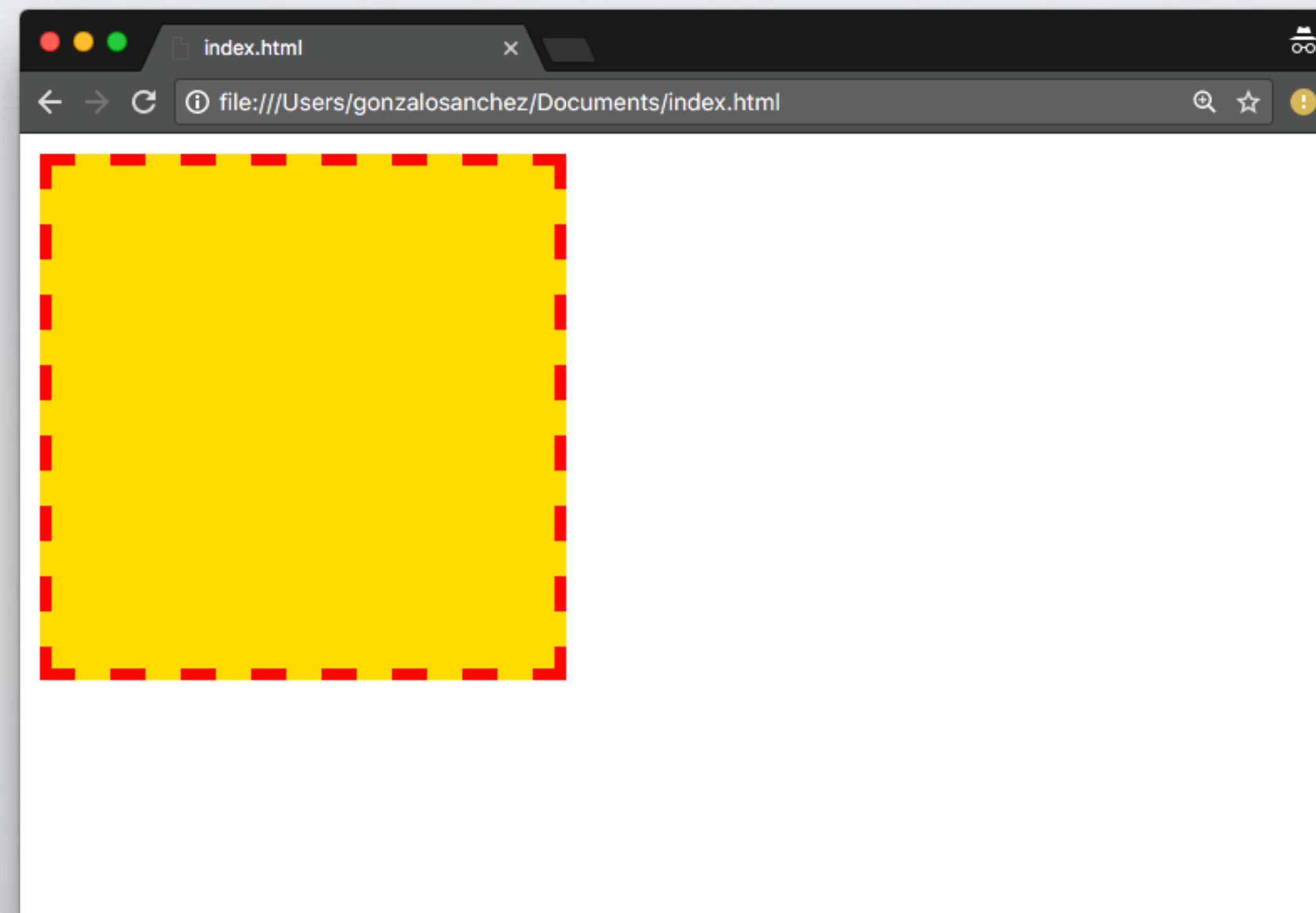
**border-color:** Color de borde. (name, hex, RGB, transparent).

CREEMOS UN CUADRADO DE COLOR AMARILLO CON WIDTH Y HEIGHT Y UN BORDE SÓLIDO DE COLOR ROJO



# BUSCA EN GOOGLE COMO PODRÍAS CREAR UN BORDE COMO ESTE

[https://www.w3schools.com/css/css\\_border.asp](https://www.w3schools.com/css/css_border.asp)



# PROPIEDAD PADDING

Es el espacio que hay entre el contenido y el borde

padding



`<p>`Muy lejos, más allá de las montañas de palabras, alejados de los países de las vocales y las consonantes, viven los textos simulados. Viven aislados en casas de letras, en la costa de la semántica, un gran océano de lenguas. Un riachuelo llamado Pons fluye por su pueblo y los abastece con las normas necesarias. Hablamos de un la boca.`</p>`



AGREGUEMOS CONTENIDO AL  
DIVY PROBEMOS EL PADDING

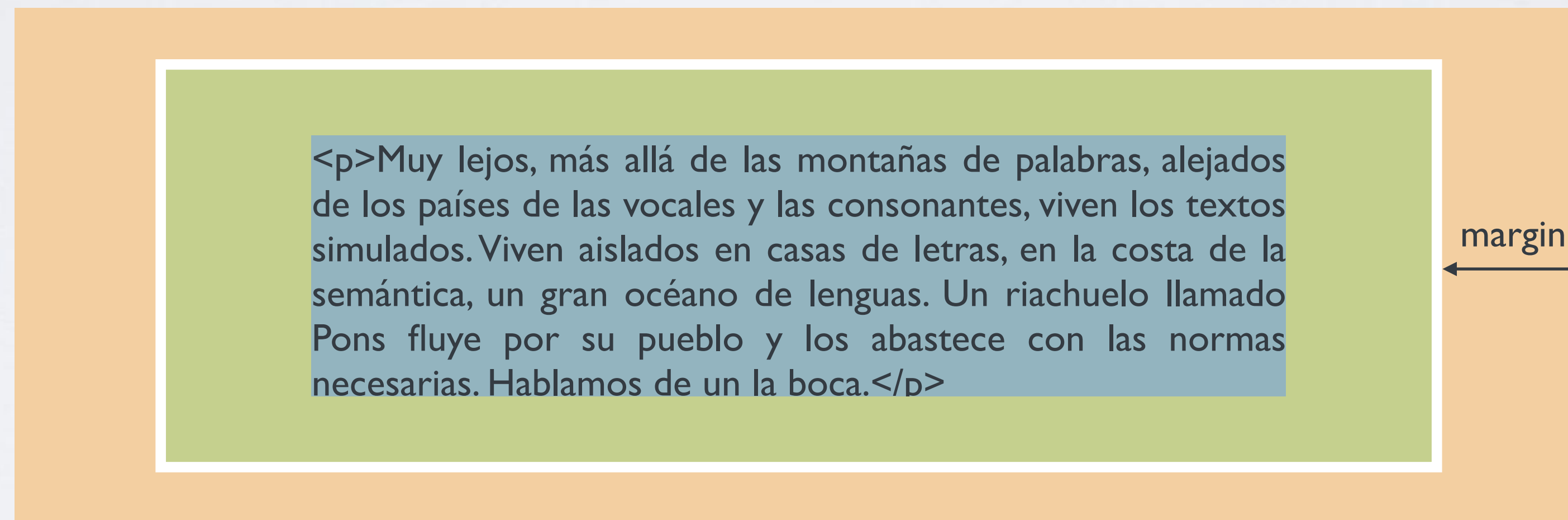
# PROPIEDAD PADDING

Se puede establecer un valor definido para todas las zonas de padding, como también de forma independiente.

```
p {  
  padding: 200px;  
  padding-left: 20px;  
}
```

# PROPIEDAD MARGIN

Es el espacio que hay entre el borde y los elementos que lo rodean.



# PROPIEDAD MARGIN

Al igual que el padding, podemos determinar un margen para todos los lados, o para un lado específico.

```
p {  
  margin: 200px;  
  margin-left: 20px;  
}
```

# TIP

Si en las propiedades de padding o margin se indican cuatro valores, el primero corresponde a **top**, el segundo a **right**, tercero a **bottom** y el cuarto a **left**.

```
p {  
  padding: 10px 20px 30px 40px;  
}
```

=

```
p {  
  padding-top: 10px;  
  padding-right: 20px;  
  padding-bottom: 30px;  
  padding-left: 40px;  
}
```



# CUANDO HABLEMOS DE POSICIÓN EXISTEN TRES PROPIEDADES IMPORTANTES

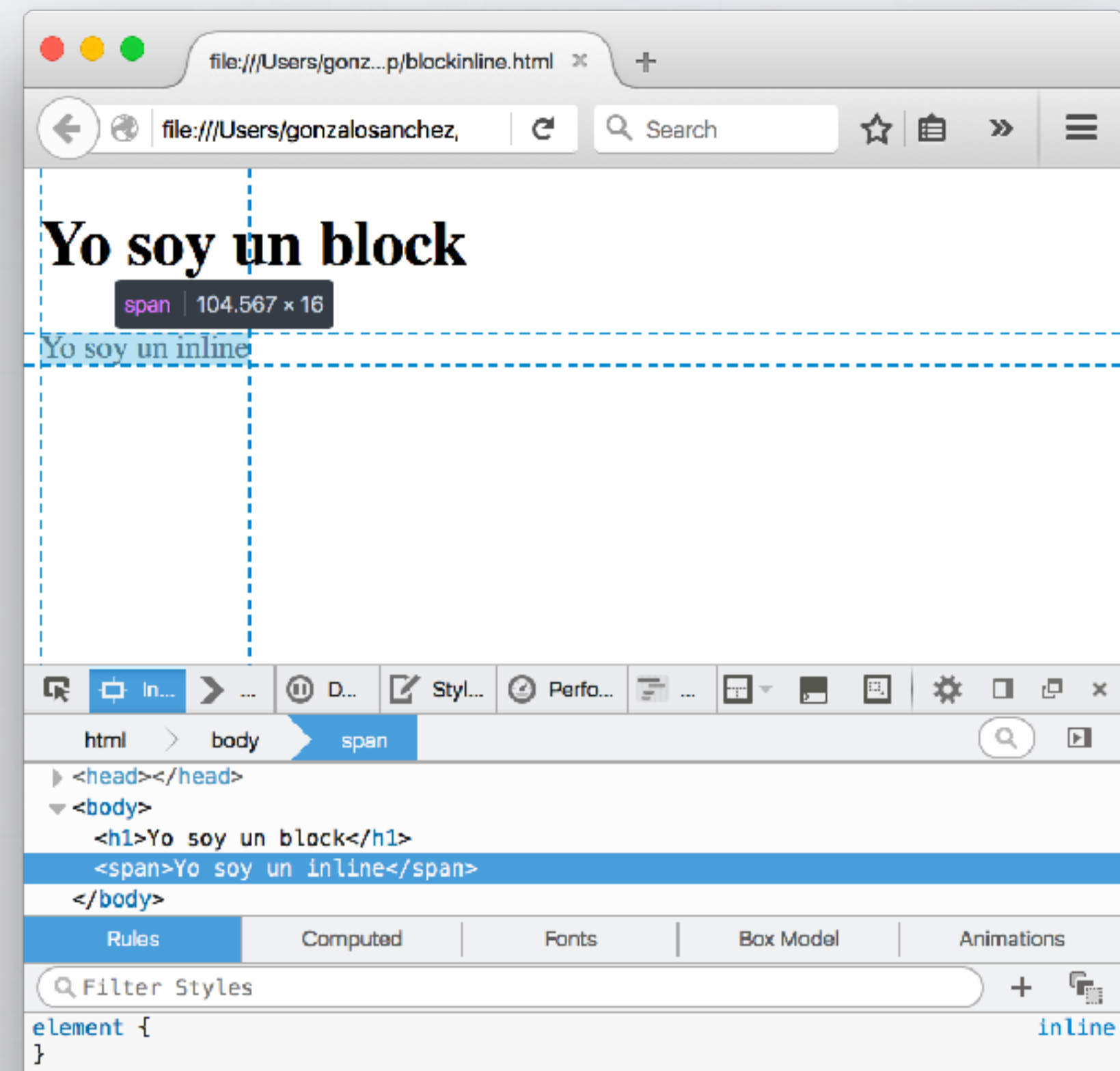
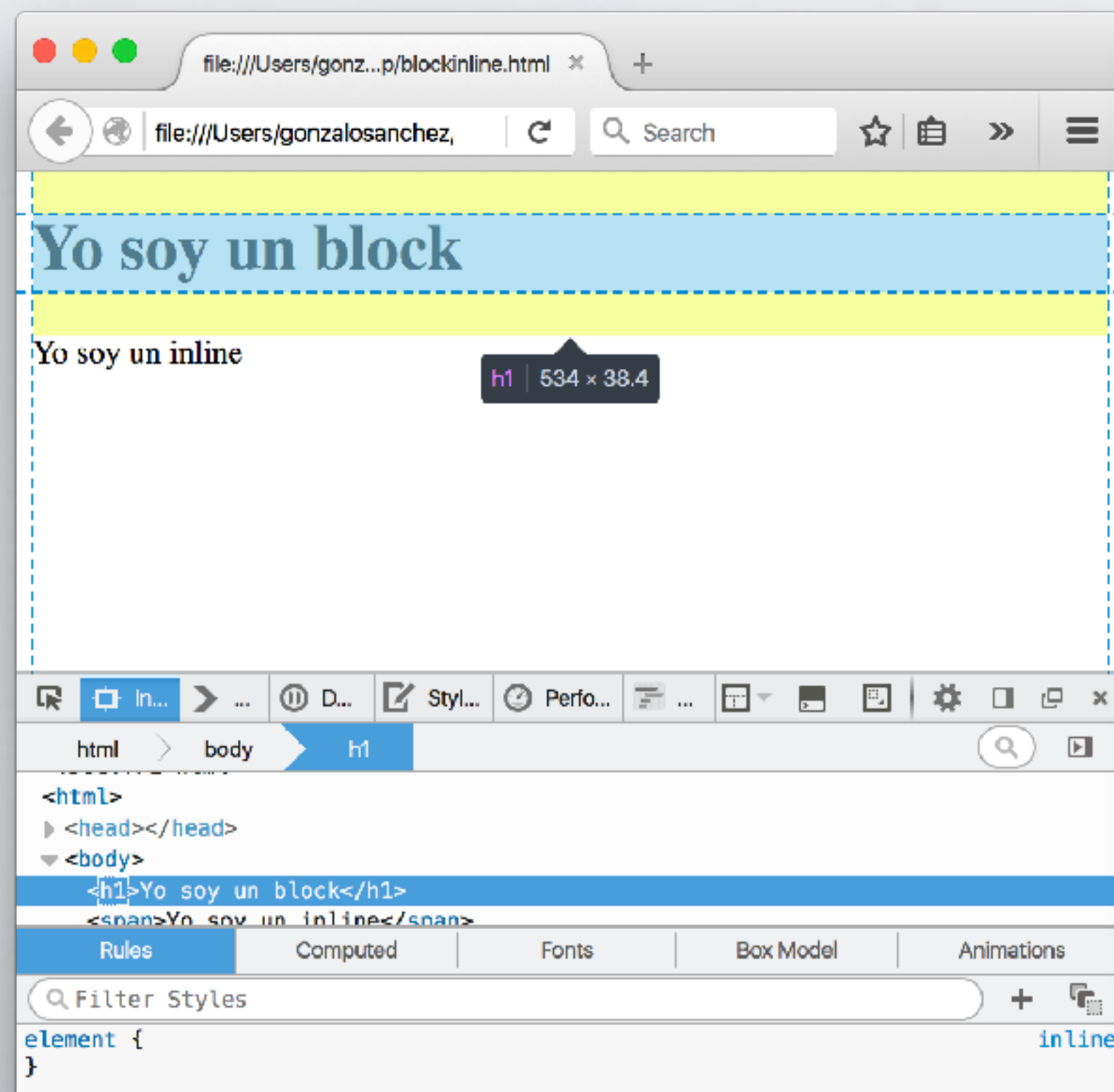
- Display
- Float
- Position

Cada una de estas 3 sirve para especificar el tipo de caja que será generada para el elemento

LA PROPIEDAD **DISPLAY** ES MUY IMPORTANTE  
PORQUE PERMITE CAMBIAR LA CANTIDAD DE  
ESPACIO QUE OCUPA UN ELEMENTO DE HTML

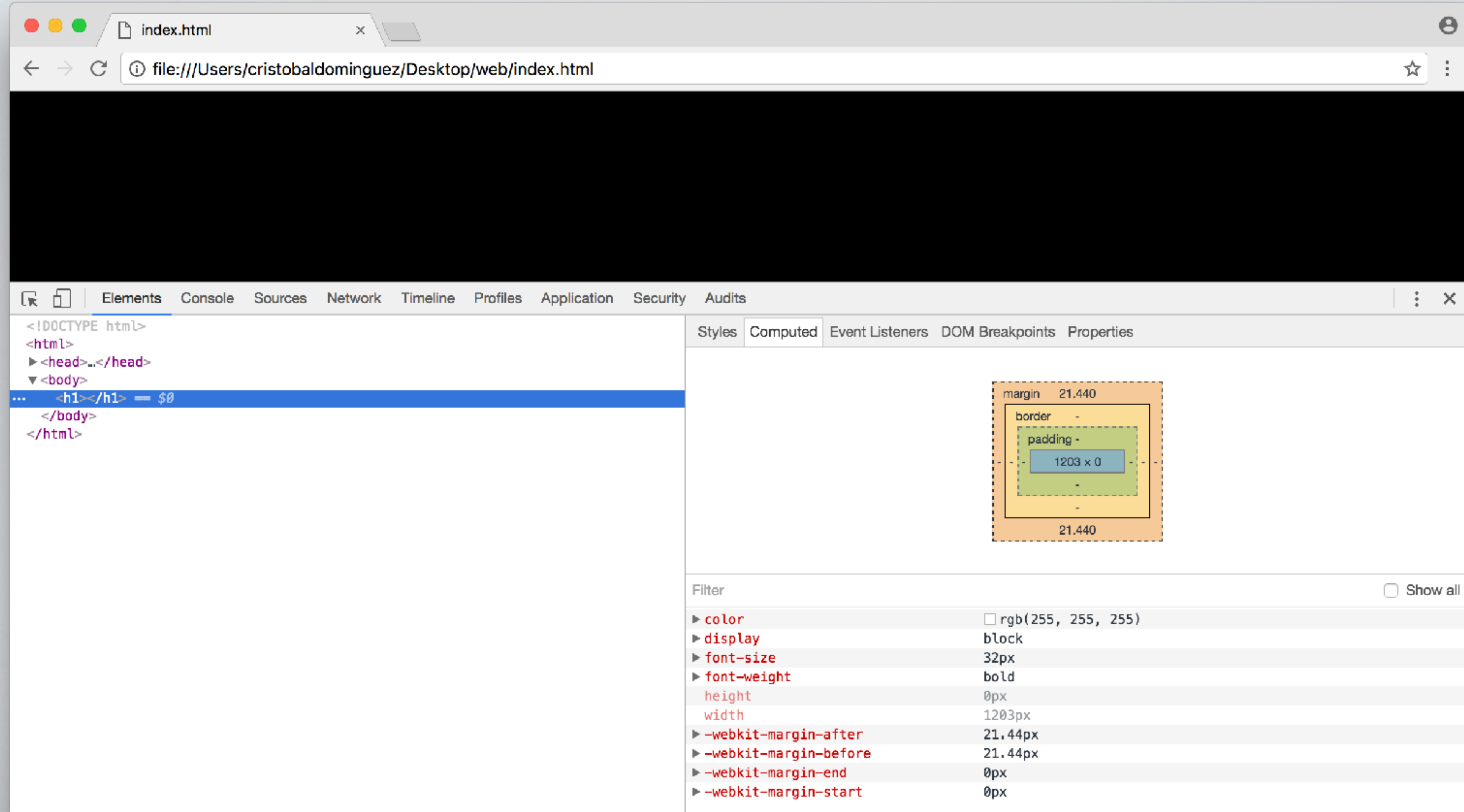
Display puede ser block o inline

Los elementos **block** ocupan todo el espacio posible. Los elementos **inline** el mínimo espacio posible



# UN H1 ¿ES BLOQUE O INLINE?

Ante la duda, inspector de elementos



The screenshot shows a web browser window with the address bar displaying `file:///Users/cristobaldominguez/Desktop/web/index.html`. The browser's developer tools are open, showing the **Elements** panel on the left and the **Styles** panel on the right.

In the **Elements** panel, the HTML structure is shown:

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    ... <h1></h1> == $0
  </body>
</html>
```

The **Styles** panel shows the default styling for the selected **h1** element:

- margin**: 21.440
- border**: -
- padding**: -
- width**: 1203px
- height**: 0px
- color**: rgb(255, 255, 255)
- display**: block
- font-size**: 32px
- font-weight**: bold

The **Filter** section at the bottom of the Styles panel shows a list of properties with checkboxes next to them, indicating they are not currently filtered out.



<IMG>  
¿Es bloque o Inline?



# LA PROPIEDAD DISPLAY SE PUEDE CAMBIAR CON CSS

- `p { display: inline }`
- `span { display: block }`

# ALINEAMIENTO

Podemos alinear bloques como los h\* y los párrafos con:

`text-align:left`

`text-align:center`

`text-align:right`

`text-align:justify`

DEBEMOS TENER ESTO EN  
CONSIDERACIÓN CUANDO  
CENTRAMOS O ALINEAMOS

```
h1 {text-align: center;}  
span {text-align: center;}
```

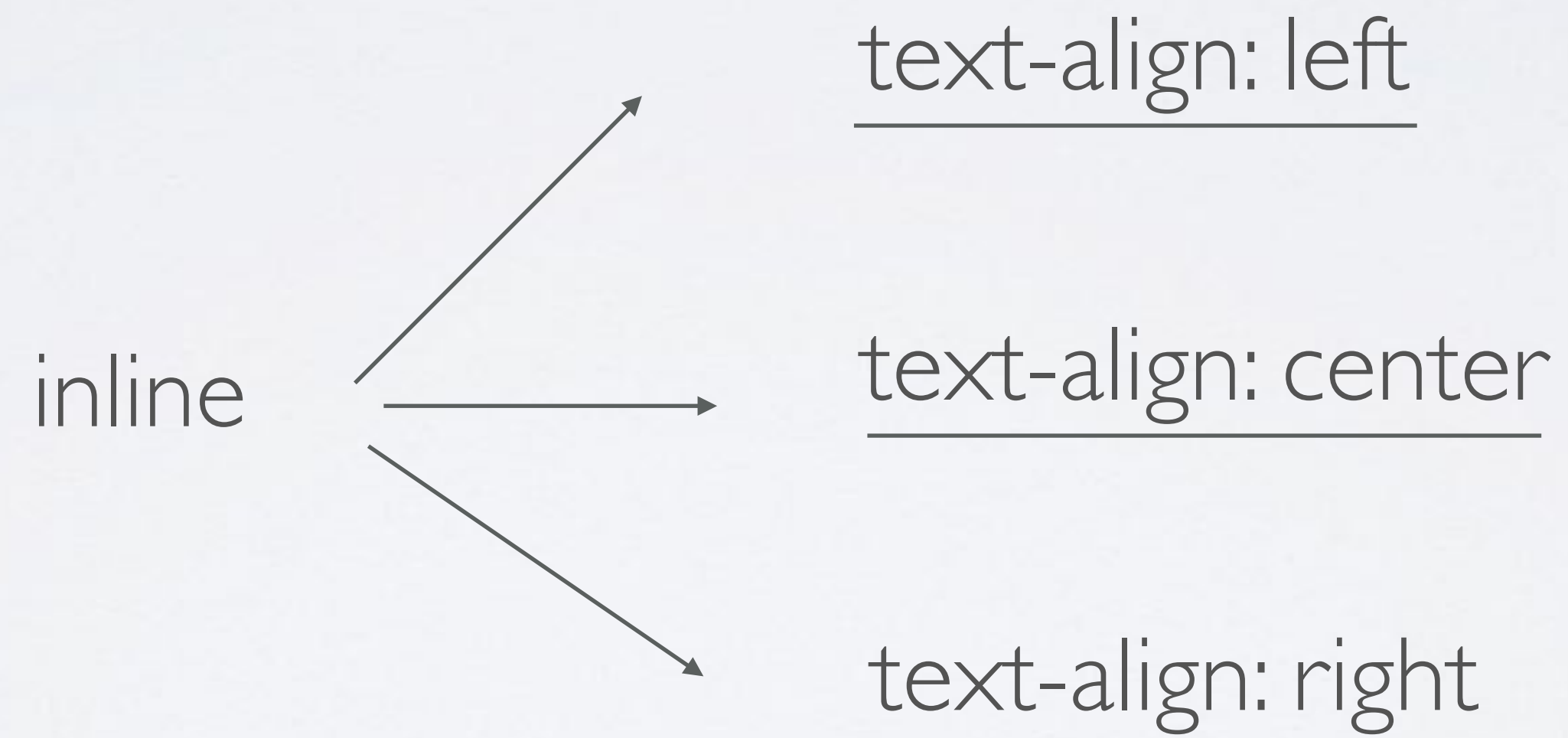
EN UN BLOQUE EL TEXTO SE ALINEA POR DEFECTO A LA DERECHA PERO SE PUEDE CAMBIAR CON TEXT-ALIGN

block text-align: left  
ancho: 100%

block text-align: center  
ancho: 100%

block text-align: right  
ancho: 100%

En inline el ancho es del tamaño del texto  
no tiene sentido alinear el texto





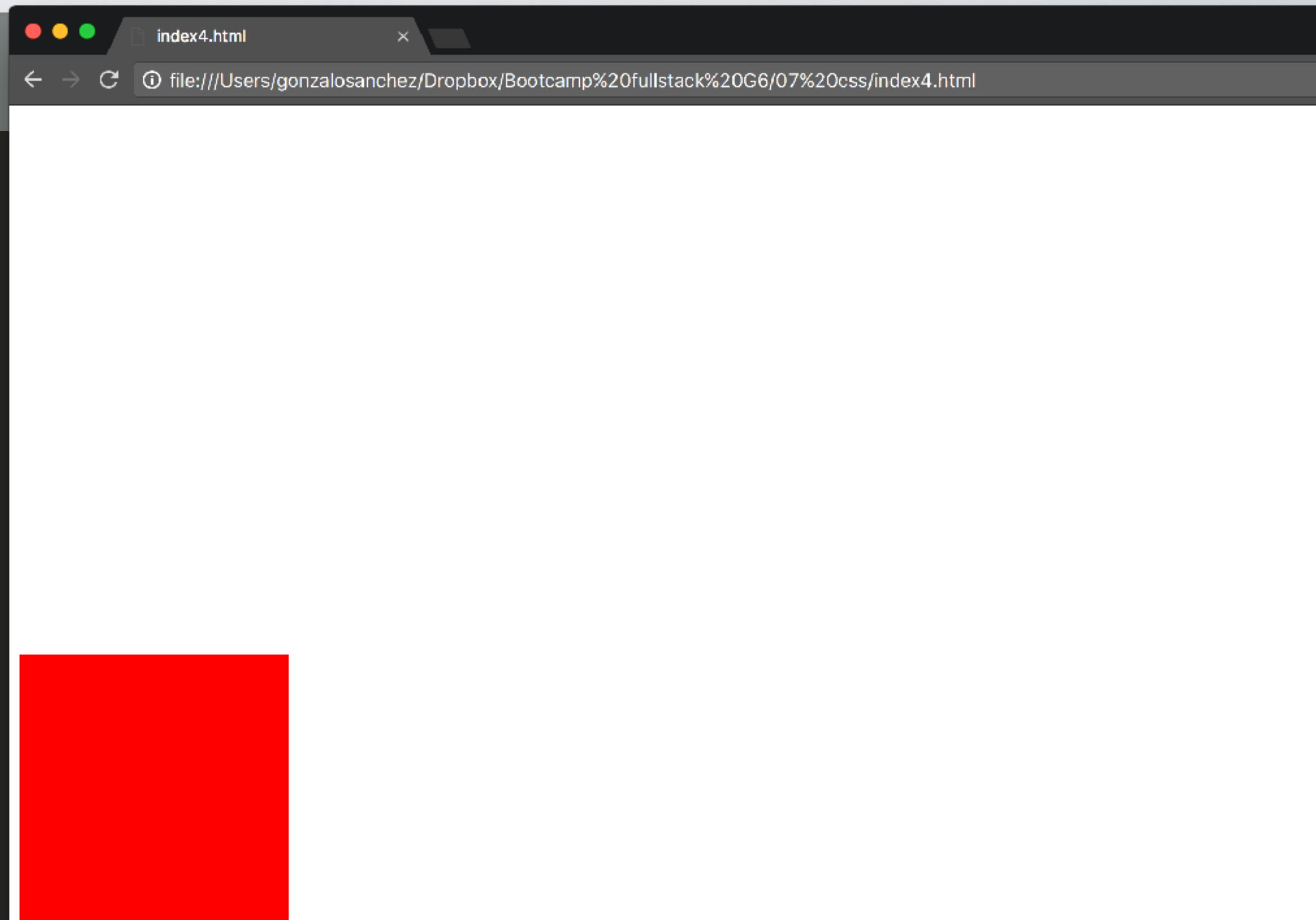
# ¿POR QUÉ SPAN NO SE CENTRA?

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <style>
    span {
      text-align: center;
    }
  </style>
</head>
<body>
  <span> hola </span>
</body>
</html>
```

# ¿POR QUÉ EL CUADRADO ROJO APARECE ABAJO?

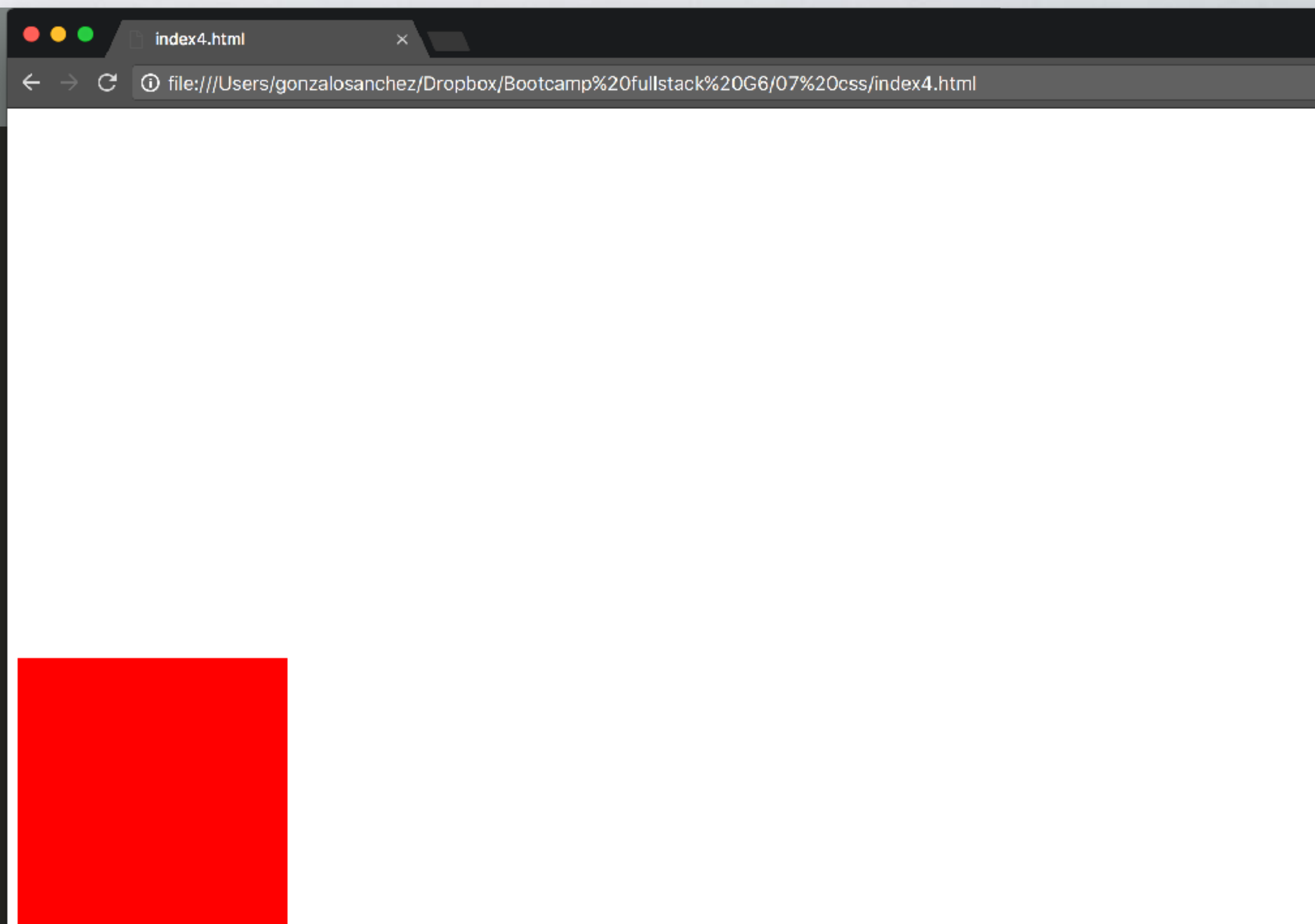


```
<!DOCTYPE html>
<html>
<head>
  <style>
    div {
      width: 200px;
      height: 200px;
    }
    .rojo{
      background-color: red;
    }
  </style>
</head>
<body>
  <div> </div>
  <div> </div>
  <div class="rojo"> </div>
</body>
</html>
```



# ¿CÓMO HACEMOS PARA QUE EL PRIMER DIV EL SEGUNDO SEAN AMARILLO? (AGREGAR SOLO 1 LÍNEA)

```
<!DOCTYPE html>
<html>
<head>
  <style>
    div {
      width: 200px;
      height: 200px;
    }
    .rojo{
      background-color: red;
    }
  </style>
</head>
<body>
  <div> </div>
  <div> </div>
  <div class="rojo"> </div>
</body>
</html>
```



¿POR QUÉ SI HACEMOS INLINE LOS  
DIVS NO PODEMOS VER LOS COLORES?

Porque inline siempre ocupa el mínimo espacio posible, al no haber contenido ocupa espacio 0, y no podemos verlo, probar agregando contenido dentro de los divs

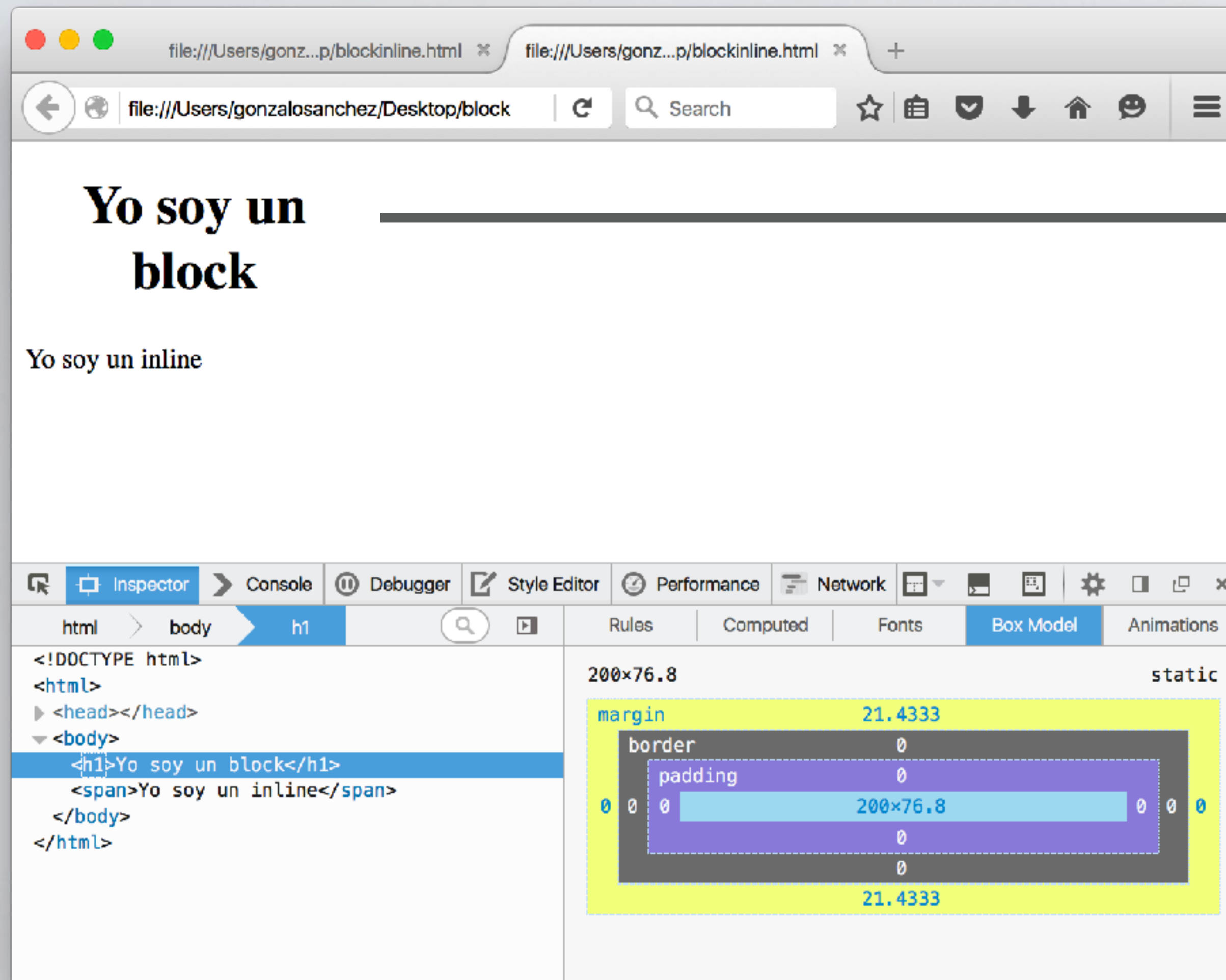


# EJERCICIO EN CLASES

- Definir 2 divs
- De tamaño 500 x 500
- border: 1px solid #000;
- Color de fondo amarillo (yellow)
- Padding: 20px
- margin: 40px

¿POR QUÉ NO QUEDAN  
UNO AL LADO DEL OTRO?

# PORQUE ES UN BLOQUE



Un **Bloque** ocupa el 100% del ancho del padre y, por ende, empuja el siguiente contenido abajo

¿QUÉ SUCEDE SI LOS  
CONVERTIMOS EN INLINE?

¿Y SI ADEMÁS DE INLINE  
AGREGAMOS TEXTO?



# EXISTE UN PUNTO MEDIO ENTRE INLINE Y BLOCK

inline-block

Es muy útil a la hora de hacer un menú

# RECORDAMOS QUE PODEMOS CAMBIAR EL TIPO

CON LA PROPIEDAD DISPLAY

`display: block`

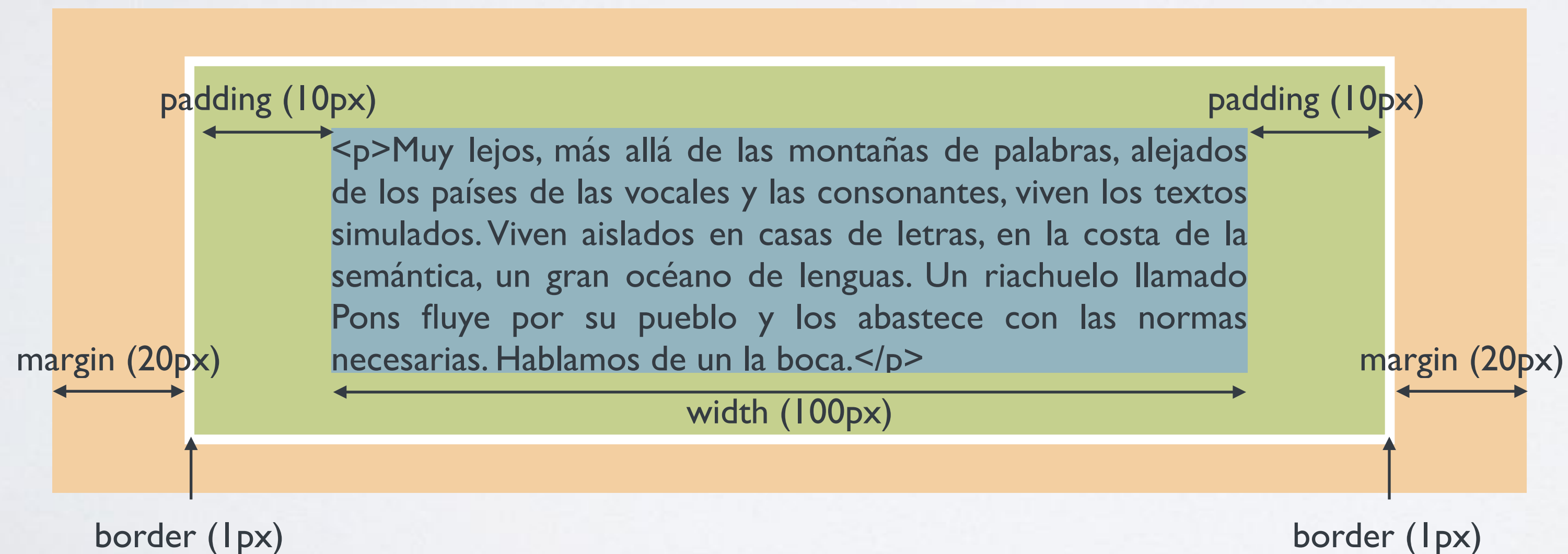
`display: inline`

`display: inline-block`

De esta forma un div o un p pueden ser inline  
y una imagen se puede transformar en bloque

# IMPORTANTE

Si queremos calcular el ancho total de un elemento, debemos considerar, a parte del width, también el padding, border y el margin que contenga ese elemento.



Ejemplo:

20px (margin) x2

1px (border) x2

10px (padding) x2

100px (width)

**El ancho total del elemento es de 162px**

# CUANDO HABLEMOS DE POSICIÓN EXISTEN TRES PROPIEDADES IMPORTANTES

- Display
- **Float**
- Position

Cada una de estas 3 sirve para especificar el tipo de caja que será generada para el elemento



ANTES DE PASAR A LA PROPIEDAD FLOAT  
REVISEMOS DE NUEVO EL COMO SE  
DIBUJA UNA PÁGINA WEB





inline

inline

inline

inline

inline

bloque

inline

inline

bloque

inline



inline

inline

bloque

inline

bloque (40% de ancho)

inline

inline

bloque

inline

bloque (40% de ancho)

bloque (50% de ancho)

# FLOAT

Cuando le agregamos la propiedad float a un objeto lo estamos sacando del flujo normal de la página para llevarlo a la izquierda o derecha.



inline



inline

inline

inline

inline

bloque

inline

inline

bloque

inline

inline

inline

bloque

inline

bloque (40% de ancho)  
con float

inline

inline

bloque

inline

bloque (40% de ancho)  
con float

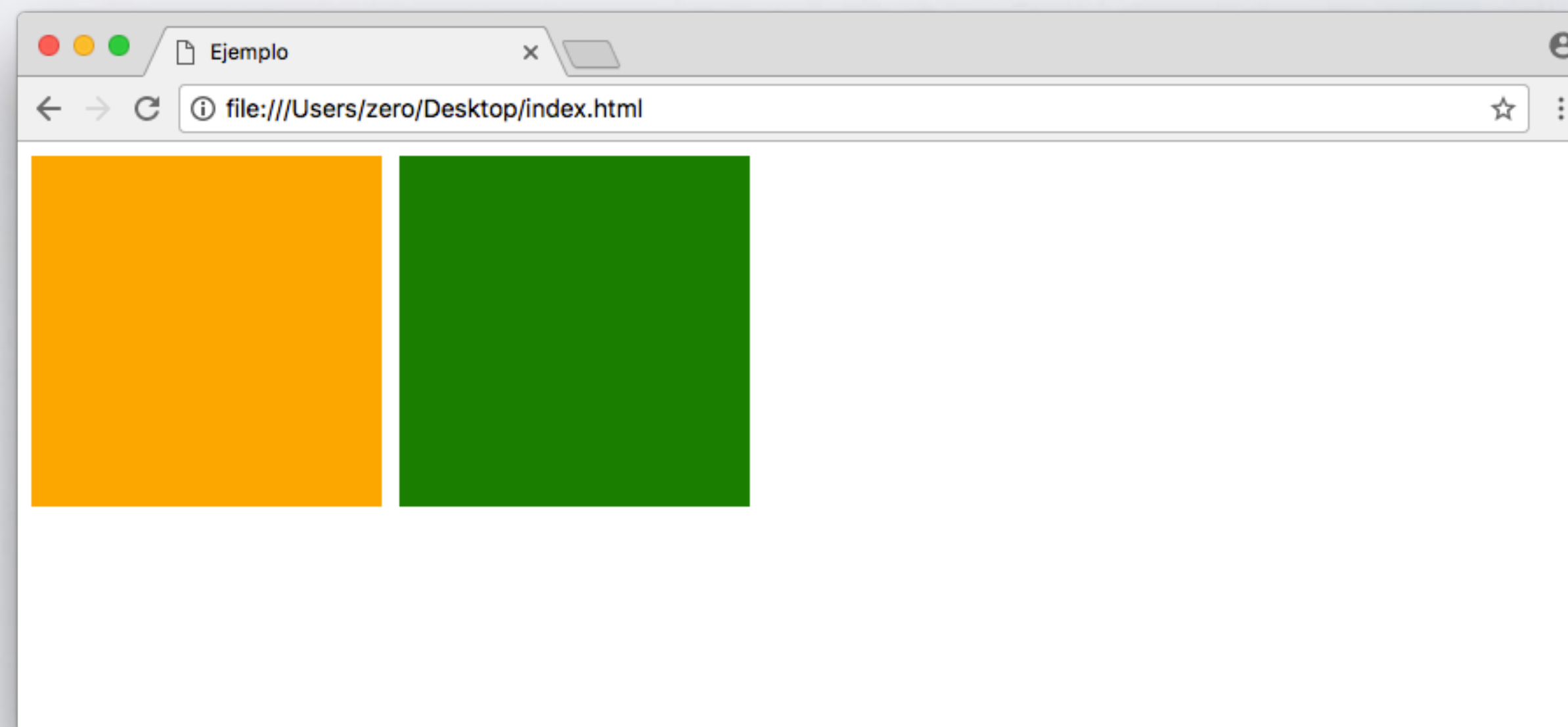
bloque (50% de ancho) con  
float



# VEAMOS UN EJEMPLO DE FLOAT

Los dos elementos div del ejemplo son bloques, sin embargo, con la propiedad float podemos posicionarlos uno al lado del otro.

```
index.html
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>Ejemplo</title>
6     <style>
7       div{
8         float: left;
9         width: 200px;
10        height: 200px;
11      }
12      div.div2{
13        background-color: green;
14      }
15      div.div1{
16        background-color: orange;
17        margin-right: 10px;
18      }
19    </style>
20  </head>
21  <body>
22
23    <div class="div1">
24    </div>
25    <div class="div2"></div>
26  </body>
27 </html>
28
```



VEAMOS EJEMPLOS CUANDO SOLO  
UNO DE LOS ELEMENTOS ES FLOAT

# EN EL SIGUIENTE CÓDIGO LA IMAGEN APARECE A LA DERECHA O A LA IZQUIERDA

```

```

```
<p style="float:left; width:300px; margin-right:20px">
```

```
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod  
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,  
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo  
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse  
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non  
proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
```

```
</p>
```

¿Qué sucede si agregamos otro bloque de 30% de ancho pero sin float?

bloque (30% de ancho)  
con float

¿Qué sucede si agregamos otro bloque de 30% de ancho pero sin float?

bloque (30% de ancho)  
con float

bloque (30% de ancho)  
sin float

Aparece abajo porque no tiene espacio



¿Y que sucede con el segundo elemento  
si el float es right?

bloque (30% de ancho)  
con float left

¿Y que sucede con el segundo elemento  
si el float es right?

bloque (30% de ancho)  
con float left

bloque (30% de ancho)  
con float right

Sucede algo similar si el segundo elemento  
tiene margin

bloque (**30% de ancho**)  
con float left

Sucede algo similar si el segundo elemento  
tiene margin

bloque (**30% de ancho**)  
con float left

bloque (40% de ancho)  
con margin **mayor a**  
**30%**

# O SI EL BLOQUE ES MÁS GRANDE

Inline (30% de ancho)  
con float



# ○ SI EL BLOQUE ES MÁS GRANDE

Inline (30% de ancho)  
con float

bloque (70% de ancho)

# ¿Y SI TENEMOS UNA IMAGEN?



Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# CLEAR

- El problema de los float es que en algunas situaciones podrían afectar al resto de los elementos sin que este sea nuestro objetivo.
- Para evitar que esto suceda se puede utilizar la propiedad clear.

```
<p style="clear:left">  
    Este párrafo es independiente de un float anterior  
</p>
```



# UN EJEMPLO DE CLEAR

block 30%  
(float: left)

block 40%  
(float: left)

# UN EJEMPLO DE CLEAR

block 30%  
(float: left)

block 40%  
(float: left)

block 30%  
(float: left)

block 40% (float: left)



# UN EJEMPLO DE CLEAR

block 30%  
(float: left)

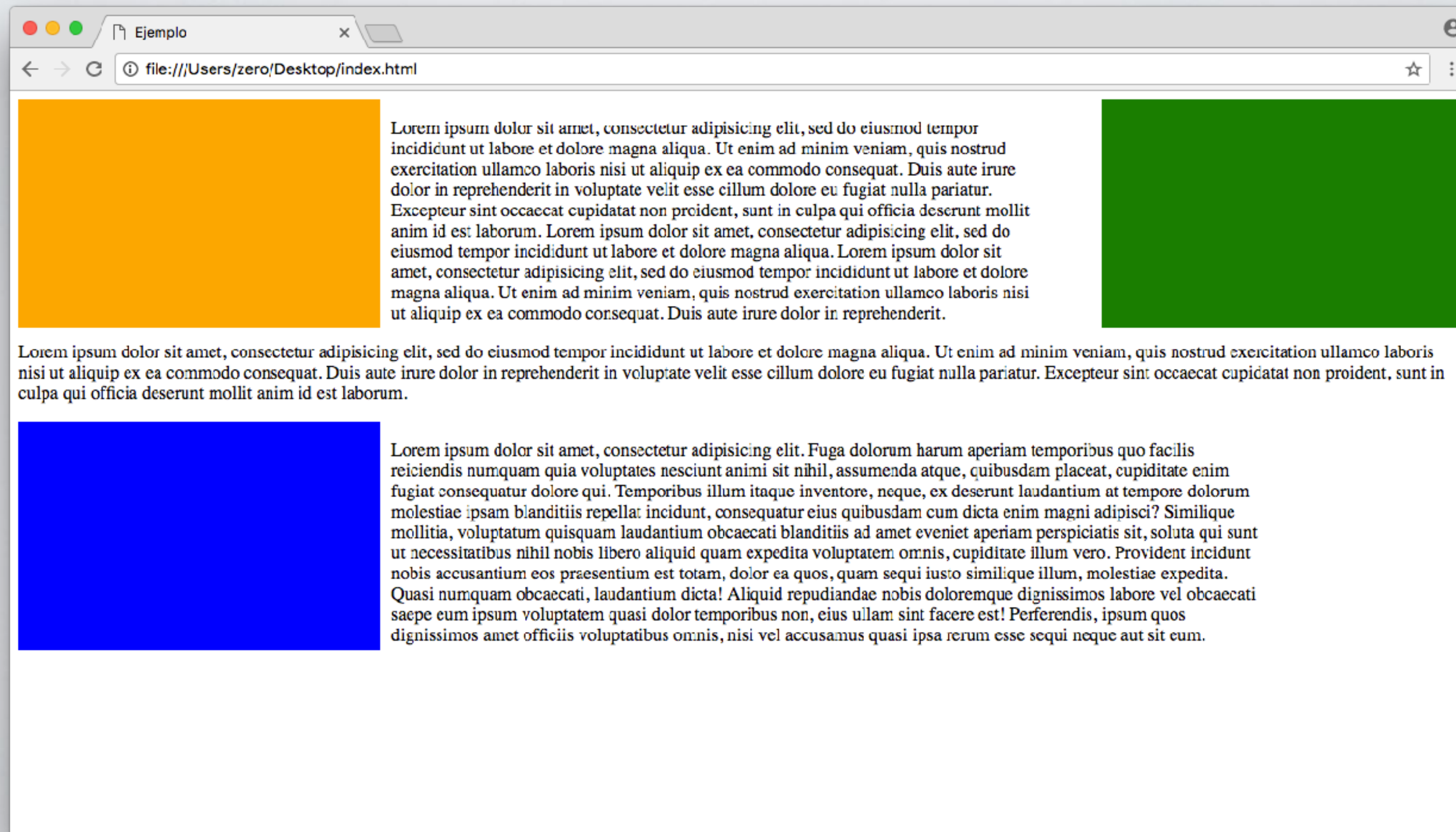
block 40%  
(float: left)

block 30%  
(**clear:left**; float:left)

block 40% (float: left)

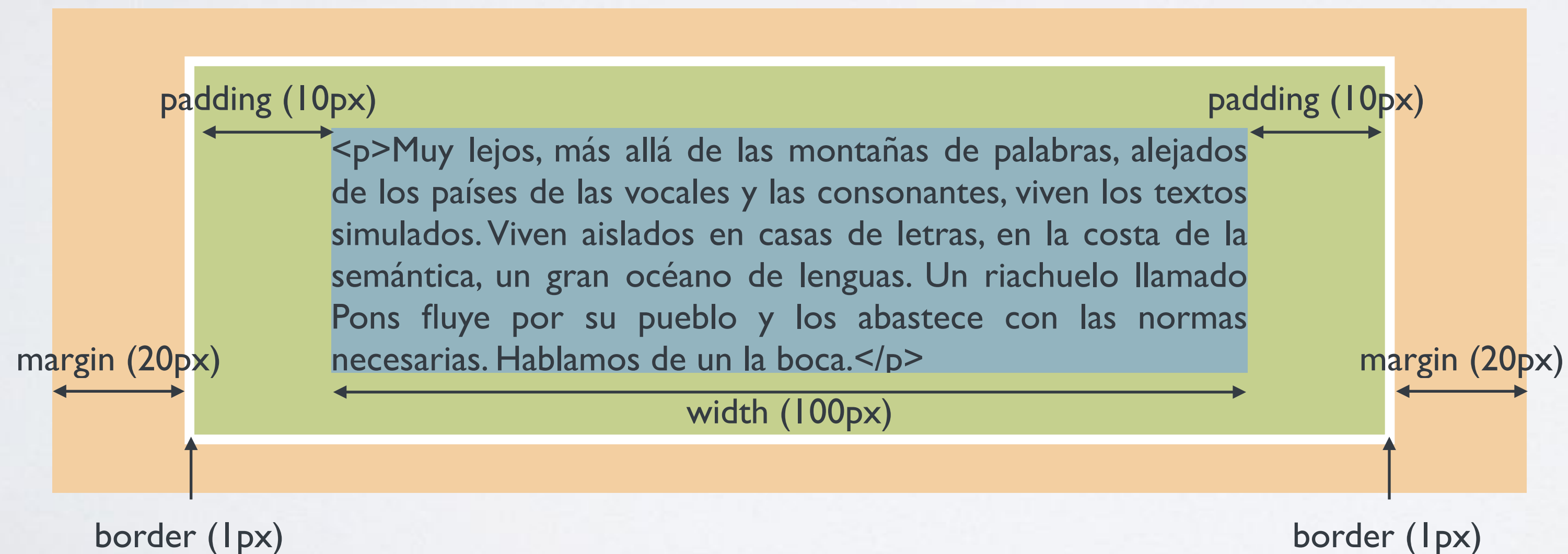
# DESAFÍO 2

A partir del desafío anterior, completar el layout aplicando clear fix



# IMPORTANTE

Si queremos calcular el ancho total de un elemento, debemos considerar, a parte del width, también el padding, border y el margin que contenga ese elemento.



Ejemplo:

20px (margin) x2

1px (border) x2

10px (padding) x2

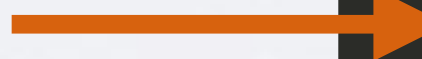
100px (width)

**El ancho total del elemento es de 162px**



# BOX-SIZING

border-box hace que el borde y padding sean parte del ancho total, haciendo más fácil el cálculo del layout.



```
.box {  
  box-sizing: border-box;  
  width: 50%;  
  padding: 10px;  
}
```

# DESAFÍO

Crear cuatro columnas en una misma fila, de aprox 25% de tamaño cada una utilizando divs y floats, y poner contenido dentro. Agregar padding y borders a las columnas, pero deben seguir habiendo 4 columnas en fila.



# CAMBIANDO TODOS LOS ELEMENTOS A BOX SIZING

```
html {  
  box-sizing: border-box;  
}  
*, *:before, *:after {  
  box-sizing: inherit;  
}
```

# REHACER EL EJERCICIO DE LAS COLUMNAS APLICANDO BORDER-BOX

# CUANDO HABLEMOS DE POSICIÓN EXISTEN TRES PROPIEDADES IMPORTANTES

- Display
- Float
- **Position**

Cada una de estas 3 sirve para especificar el tipo de caja que será generada para el elemento

# POSITION

Nos permite fijar contenido en función de otro, como, por ejemplo, hacer un sticky navbar o poner un texto sobre una imagen.

# POSITION: STATIC

- Es el valor por defecto, todos los elementos se encuentran estáticos de acuerdo al flujo normal del HTML.
- Hasta el momento todos los elementos que hemos utilizado utilizaron este valor.

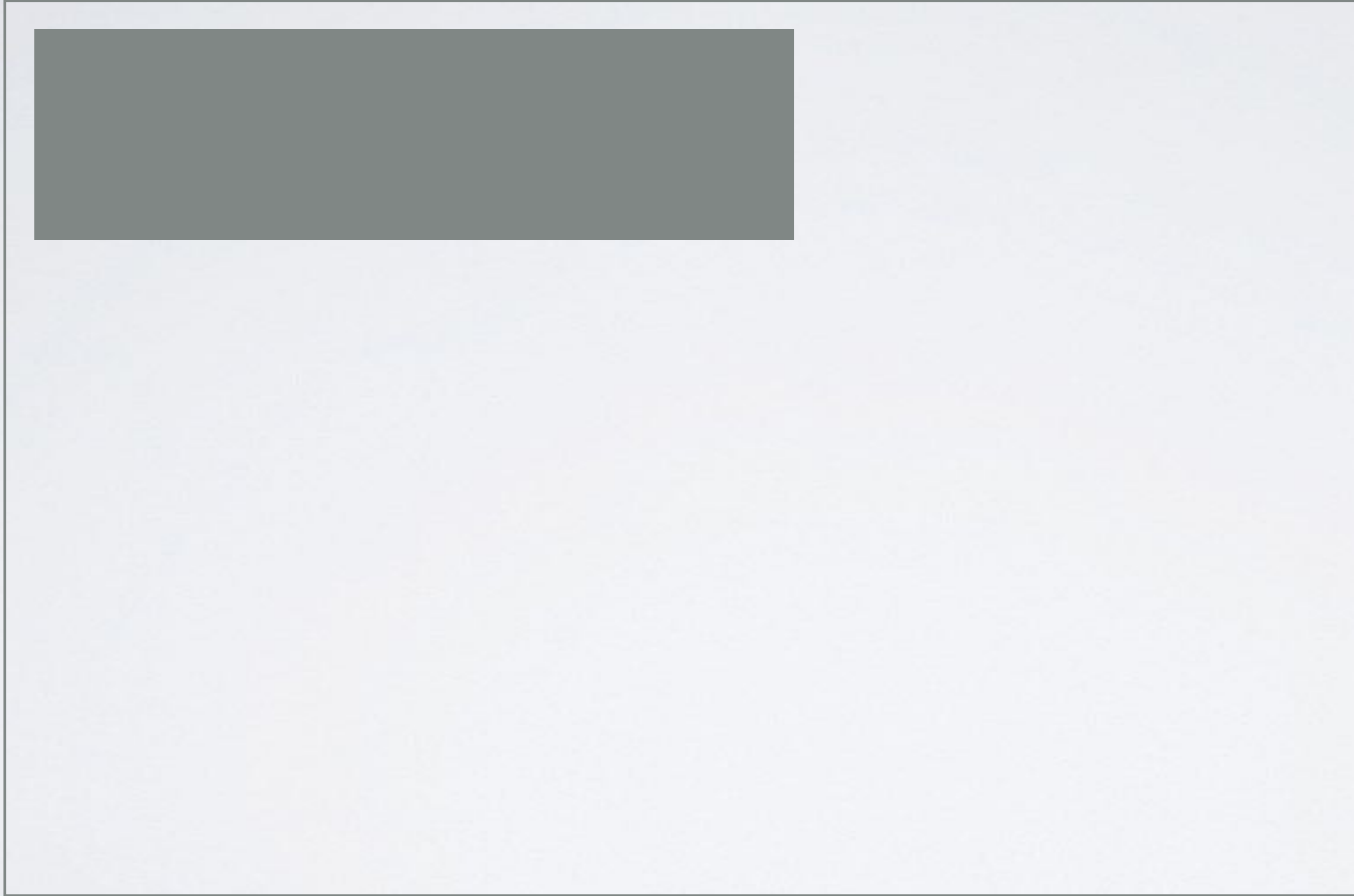


# POSITION: STATIC

# POSITION: STATIC



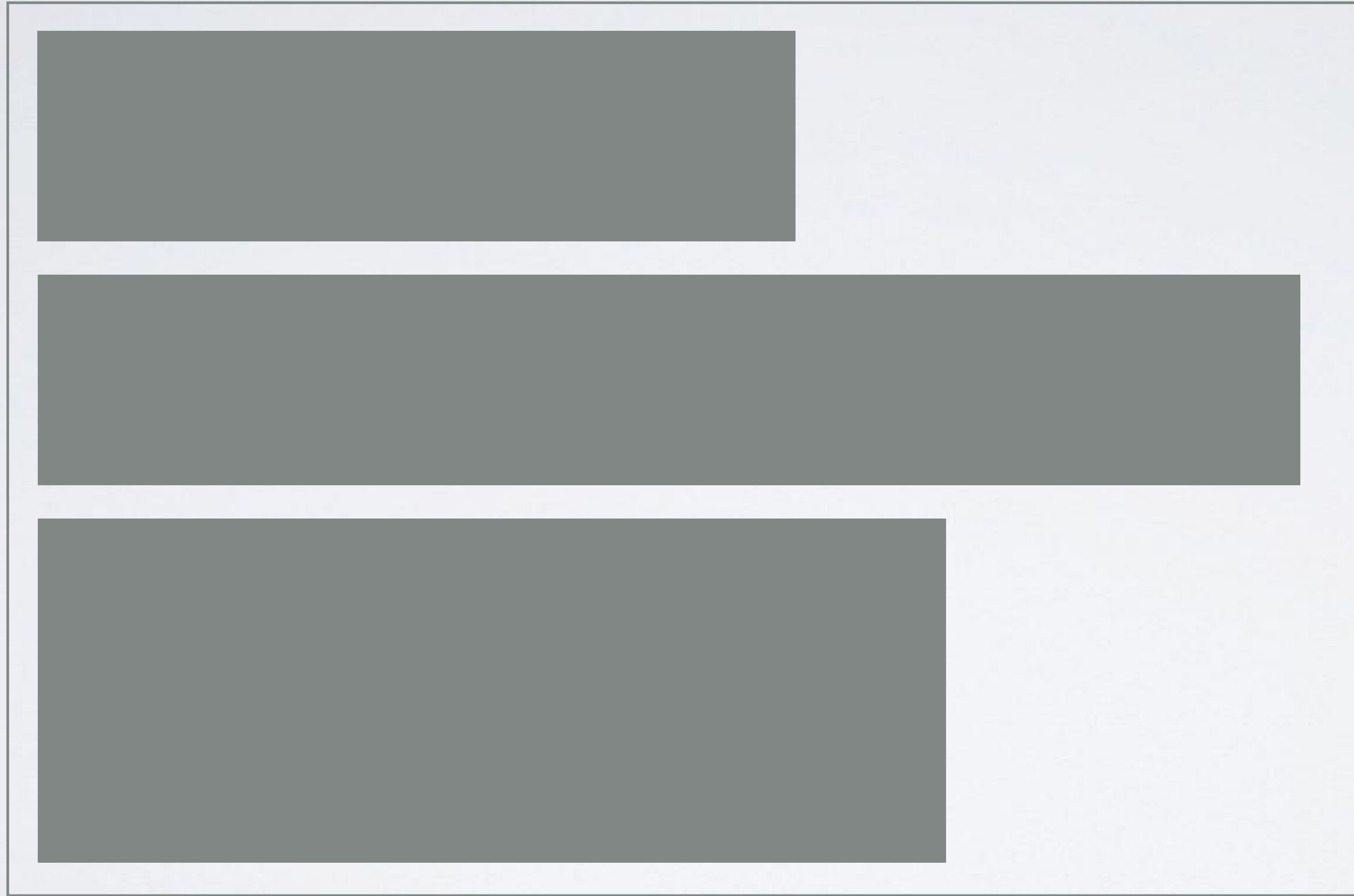
# POSITION: STATIC



# POSITION: STATIC



# POSITION: STATIC





# POSITION: STATIC



Hace que los elementos sigan el flujo normal de dibujado de elementos en HTML

# POSITION RELATIVE

# POSITION RELATIVE



# POSITION RELATIVE

position: static

# POSITION RELATIVE

```
position: static
```

```
position: relative;  
float: left;
```



# POSITION RELATIVE

```
position: static
```

```
position: relative;  
float: left;
```

Position relative  
sitúa los elementos  
acorde al flujo normal  
pero luego se pueden  
reposicionar ocupando  
las propiedades como  
left, top, right y bottom

# POSITION RELATIVE

# POSITION RELATIVE



# POSITION RELATIVE

position: static

# POSITION RELATIVE

position: static

position: relative



# POSITION RELATIVE

position: static

position: relative

Si los valores top, bottom, left y right son cero, entonces el elemento se sitúa como si fuera un elemento con posición estática.

# ¿CUÁL ES LA DIFERENCIA ENTRE UTILIZAR MARGIN Y POSICIONES?

Versión con margin

```
position: relative; float: left  
width: 50%; margin-left:  
100px;
```

```
position: static; float: left  
width: 50%;
```

# ¿CUÁL ES LA DIFERENCIA ENTRE UTILIZAR MARGIN Y POSICIONES?

Versión con margin

```
position: relative; float: left  
width: 50%; margin-left:  
100px;
```

```
position: static; float: left  
width: 50%;
```

# ¿CUÁL ES LA DIFERENCIA ENTRE UTILIZAR MARGIN Y POSICIONES?

Versión con left

```
position: relative; float: left  
width: 50%; left: 100px;
```

```
position: static; float: left  
width: 50%;
```

# ¿CUÁL ES LA DIFERENCIA ENTRE UTILIZAR MARGIN Y POSICIONES?

Versión con left

```
position: relative; float: left  
width: 50%; left: 100px;
```

```
ition: static; float: left  
width: 50%;
```



# Z-INDEX

Position: static; float: left  
width: 50%; **left: 100px;**

Position: relative; float: left  
width: 50%;

El elemento con mayor z-index va sobre el otro

# Z-INDEX

Position: static; float: left  
width: 50%; **left: 100px;**

n: relative; float: left  
width: 50%;

El elemento con mayor z-index va sobre el otro

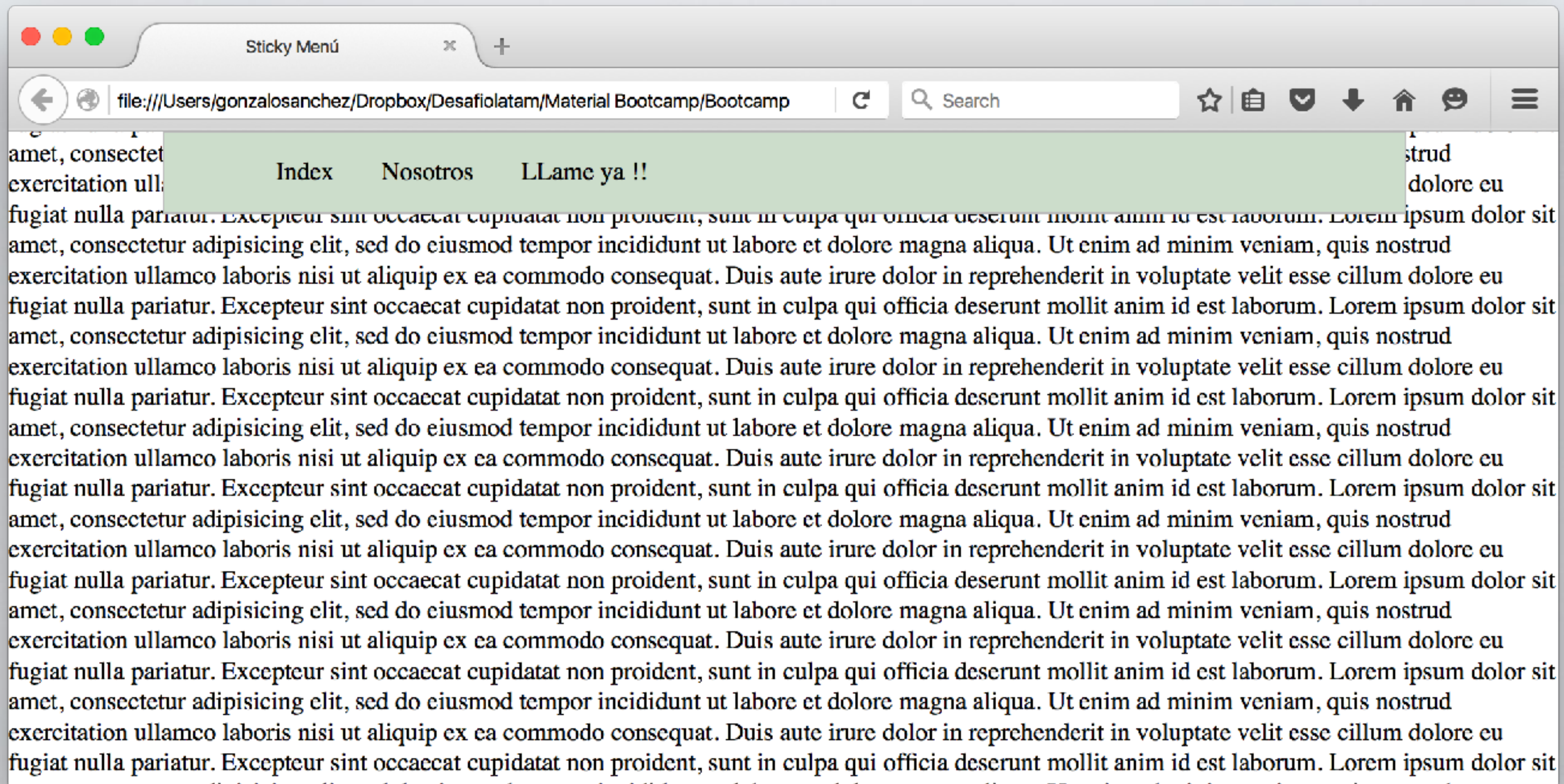
# POSITION: FIXED

- Sirve para fijar contenido en función del **viewport**, eso quiere decir que te sigue independiente de que hagas scroll
- Sirve para construir sticky navbars o un conjunto de iconos sociales que te siga a lo largo de la página



# EJEMPLO POSITION FIXED

Un sticky navbar te sigue independiente de que hagas scroll





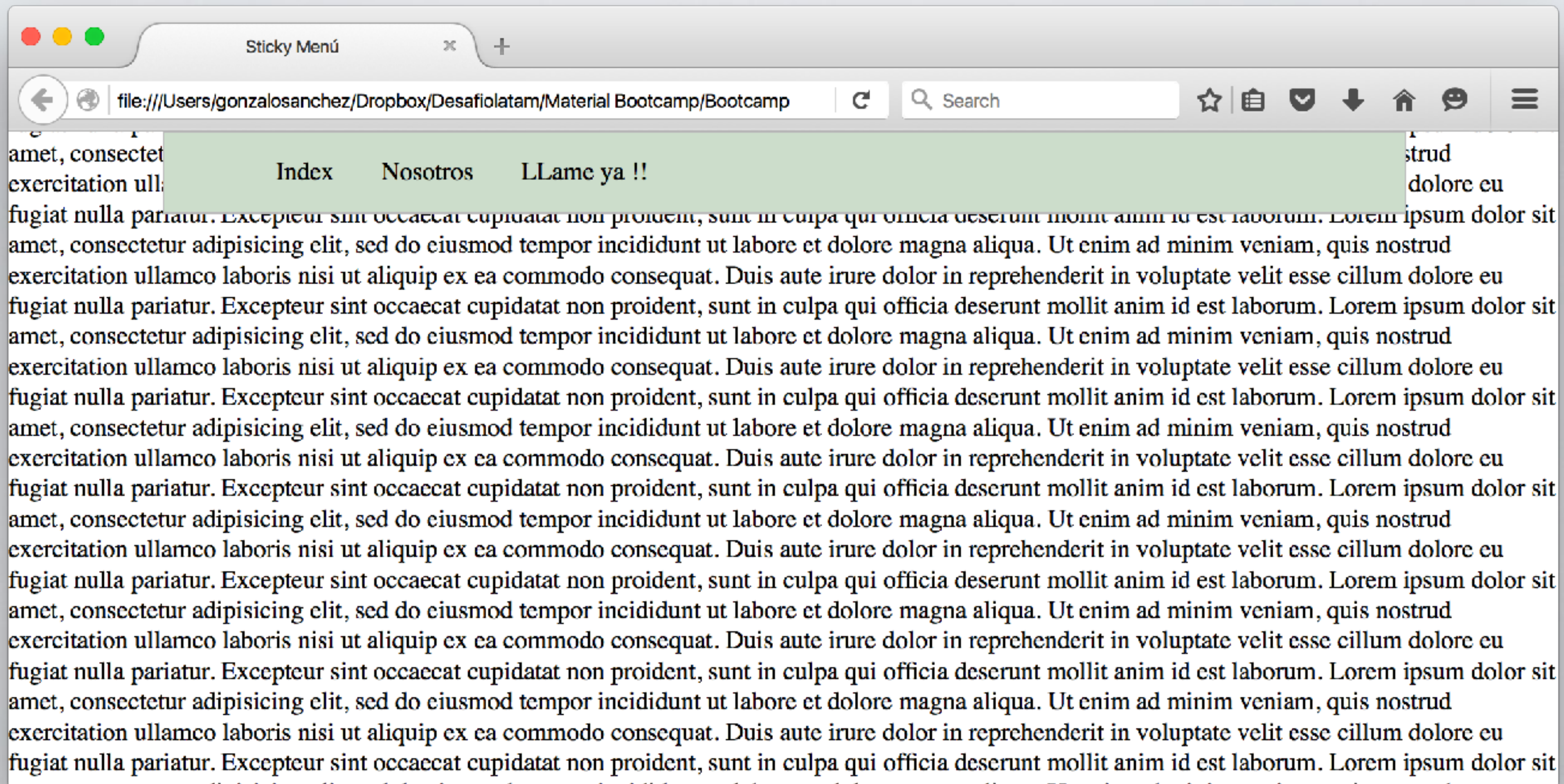
# POSITION: ABSOLUTE

- Permite **disponer contenido en función a su padre**, siempre y cuando este tenga una posición distinta de static. En ese caso tomará como referencia la posición del documento.
- Para situar el contenido ocuparemos las propiedades top, left, right y bottom.
- Podemos ocupar valores negativos para esas propiedades.  
Ejemplo right: -50px es a la derecha pero 50 pixeles menos.



# EJEMPLO POSITION FIXED

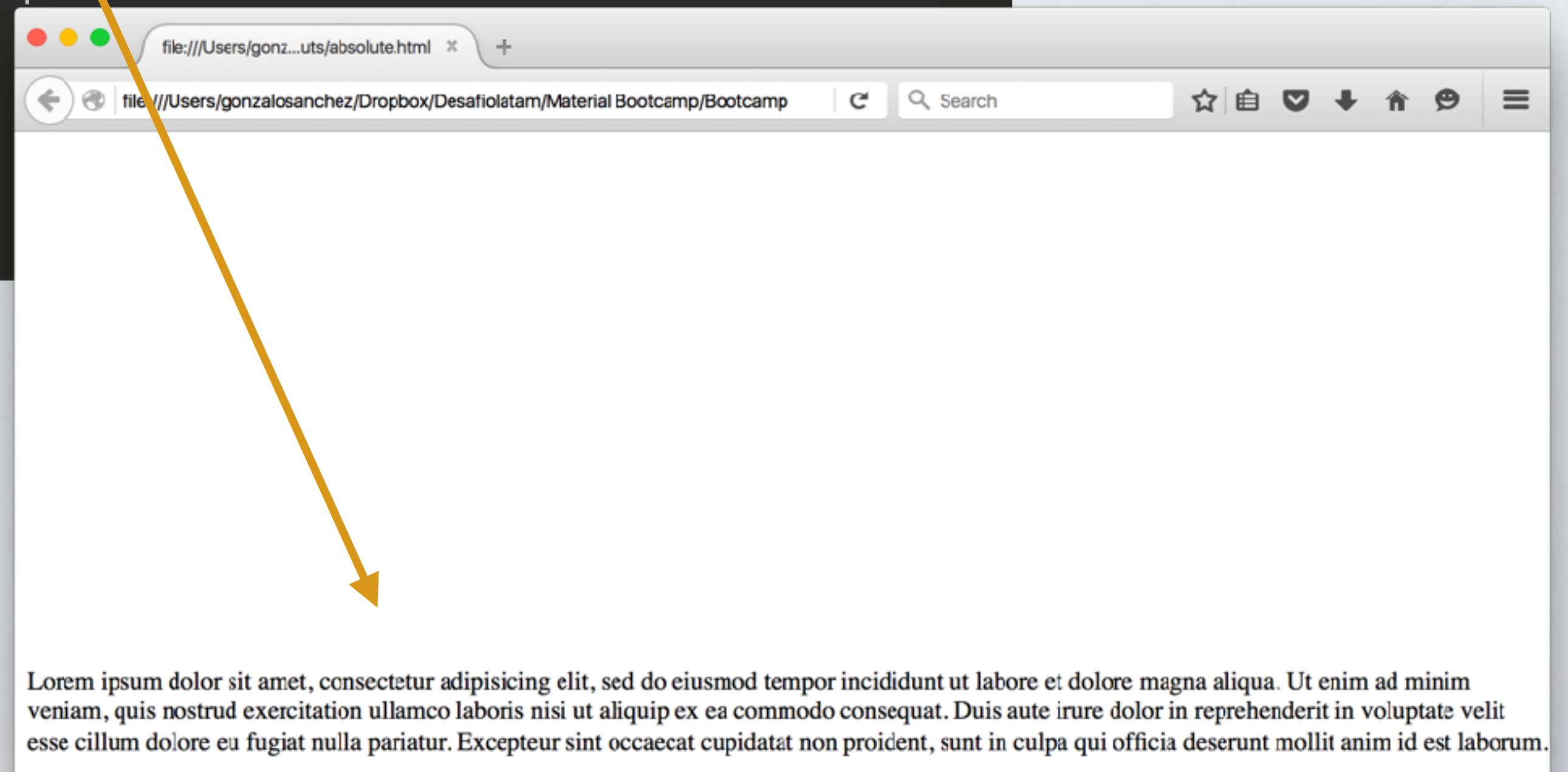
Un sticky navbar te sigue independiente de que hagas scroll





# POSITION: ABSOLUTE

```
<p style="position:absolute; bottom:0px;">Lorem ipsum dolor sit amet,  
consectetur adipisicing elit, sed do eiusmod tempor incididunt ut  
labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud  
exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.  
Duis aute irure dolor in reprehenderit in voluptate velit esse cillum  
dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat  
non proident, sunt in culpa qui officia deserunt mollit anim id est  
laborum.  
</p>
```



# FIJANDO UN FOOTER AL FINAL

<http://matthewjamestaylor.com/blog/keeping-footers-at-the-bottom-of-the-page>

LA POSICIÓN ABSOLUTA SE PUEDE OCUPAR EN  
CONJUNTO CON LA RELATIVE O FIXED, PARA  
HACER MÁS FÁCIL LA DISPOSICIÓN DE CONTENIDO