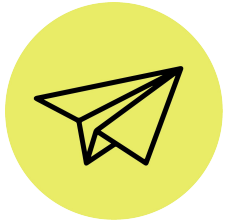
CÓDIGO PA' LANTE

EidoS
ALWAYS LEARNING

accenture

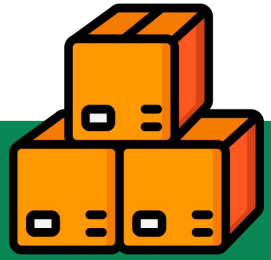


¡Les damos la bienvenida!

CLASE 5

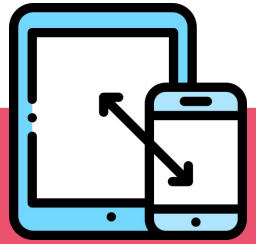
CÓDIGO
PA' LANTE

CÓDIGO PA' LANTE



Box Model

Flexbox



Responsive

CÓDIGO
PA' LANTE

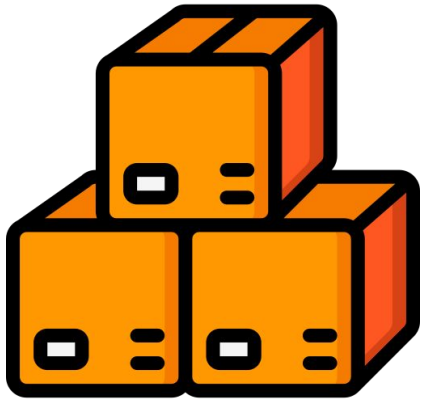
CSS Box Model (modelo de caja)

CÓDIGO PA' LANTE

¿Cómo separamos los elementos HTML?

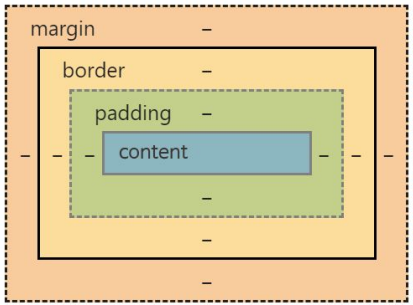
Hasta ahora no vimos cómo poner espaciados con CSS, para poder separar a los elementos HTML unos de otros, o separar el contenido de un elemento de su borde. En esta clase conoceremos con qué propiedades podemos hacer esto, pero antes es clave que comprendamos **cómo son representados los elementos HTML para CSS.**

Spoiler: son cajas.



¿Qué es el modelo de cajas?

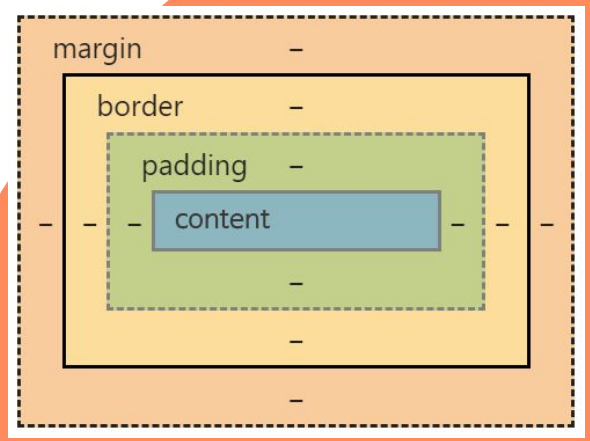
En CSS, **cada elemento HTML es representado como una caja compuesta de varias capas que recubren su contenido.**
Podemos observar en el inspector del navegador, debajo de todo en la pestaña de estilos.
A continuación, conoceremos las capas que componen estas "cajas" y con qué propiedades modificarlas.



Capas de un elemento

- **Content:** es el espacio que ocupa aquello que contiene el elemento. Es lo más interno de la "caja". Ajustamos su tamaño con **width** y **height**.

- **Padding:** es el espacio que se coloca alrededor del contenido (entre el contenido y el borde). Modificamos su tamaño con la propiedad **padding**.

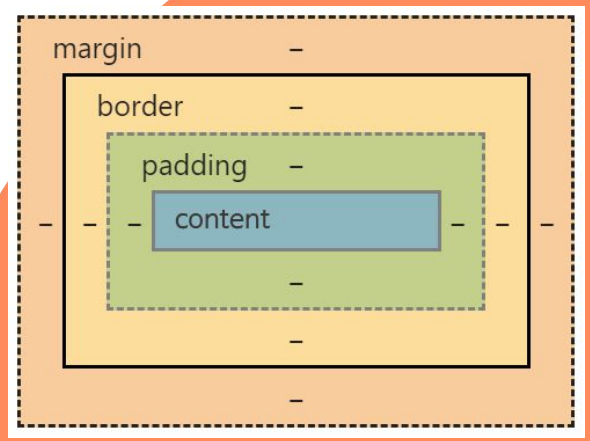


Capas de un elemento

- **Border:** luego, entre el relleno (*padding*) y el margen (*margin*), está el **borde**.
  Puede tener su propio estilo como ser de un color específico y de un grosor determinado.
  Para modificarlo usamos la propiedad **border** que abrevia las propiedades border-style, border-width y border-color.

- **Margin:** por último, está la capa del **márgen**, la más externa. Representa el espacio vacío que separa a este elemento de los que lo rodean. Usaremos la propiedad **margin** para modificarla.

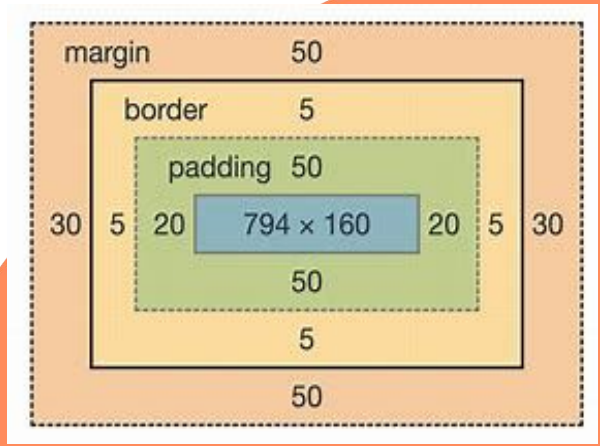
Padding y Margin son espacios. Son transparentes y por tanto solo podemos modificar su tamaño.



Ejemplo

style.css

```
div {
    width: 794px;
    height: 160px;
    padding: 50px 20px;
    border: 5px solid #000000;
    margin: 50px 30px;
}
```



CÓDIGO PA' LANTE

Propiedades abreviadas

Cuando trabajamos el **padding**, **border** o **margin** de un elemento, podemos:

- Especificar valores **lado por lado**, con el nombre de la propiedad y **–top, -right, -bottom, -left**.
- O usar las **propiedades abreviadas** e indicar valores para varios lados en una sola línea.

```
/* Lado por lado */
margin-top: 10px;
margin-right: 40px;
margin-bottom: 30px;
margin-left: 20px;
```

```
/* Propiedad abreviada */
/* Mismo valor a los 4 lados */
margin: 10px;
/* Arriba|Abajo - Izq|Der */
margin: 10px 20px;
/* Arriba - Izq|Der - Abajo */
margin: 10px 20px 30px;
/* Valor diferente para cada lado */
margin: 10px 20px 30px 40px;
```

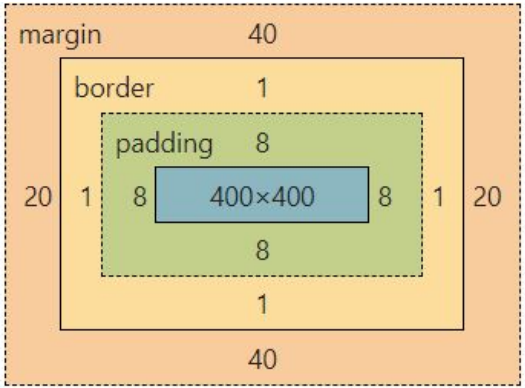
CÓDIGO
PA' LANTE

Desafío:

¿Cuánto espacio ocupa un elemento?

CÓDIGO PA' LANTE

Calcular el espacio que ocupa cada elemento no es tan sencillo en CSS, para tí ¿cuál es el **tamaño que ocupa a lo ancho** el siguiente elemento?



400px

418px

429px

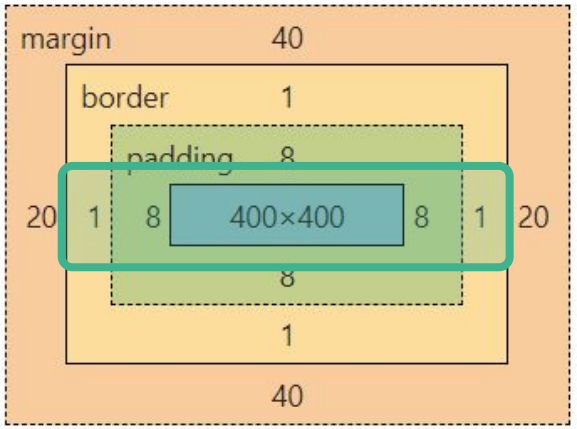
CÓDIGO PA' LANTE

El tamaño que ocupa a lo ancho este elemento es la sumatoria de (width + padding + border) pero ojo, no siempre se calcula así... 😮

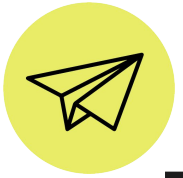
400px

418px ✓

429px



Además, estarán esos 20px de margin que lo separarán de otros elementos



CÓDIGO PA' LANTE

Box-sizing

Con la propiedad box-sizing podemos configurar **cómo queremos que se calcule el ancho y alto total de los elementos en pantalla.** Veremos dos valores posibles y cuál nos conviene utilizar.

`box-sizing: content-box;`

`box-sizing: border-box;`

CÓDIGO PA' LANTE

Box-sizing: content-box;

Al utilizarse "content-box" como valor de la propiedad "box-sizing", el ancho y alto total de cada elemento se calculan de la siguiente manera:

Ancho del elemento = width + border + padding
Alto del elemento = height + border + padding

Este es el valor por defecto que tienen todos los elementos: es confuso y difícil de calcular. 😢

Box-sizing: border-box;

En cambio, al utilizarse "**border-box**" como valor de la propiedad box-sizing, **el ancho total de cada elemento va a ser siempre igual al valor de la propiedad "width" y su alto al "height"**. El tamaño del **"border" y "padding" van a estar incluidos**. Lo que sucede es que el contenido se va a reducir para hacer lugar al borde y al padding.

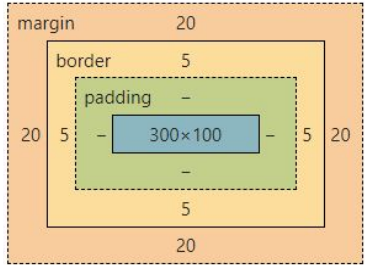
Ancho del elemento = width
Alto del elemento = height

Este valor hace las cosas más sencillas, por eso es conveniente aplicarlo a todos los elementos antes de empezar a escribir el resto del CSS. Lo haremos con el selector universal (*). 👍

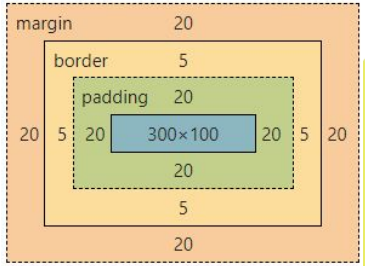
Box-sizing: content-box;

Ambos tienen un tamaño de 300 x 100, pero el padding hace que el segundo elemento se vea más grande:

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Doloremque consequatur ducimus vitae sit quas culpa mollitia iusto dolorem voluptatem corrupti?



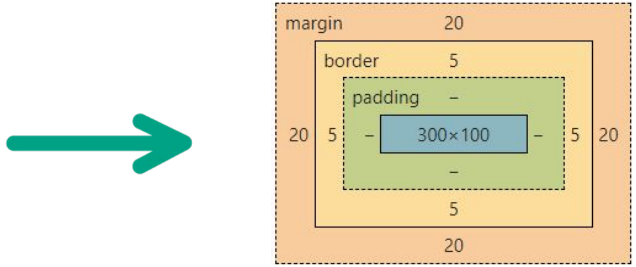
Lorem ipsum dolor sit, amet consectetur adipisicing elit. Doloremque consequatur ducimus vitae sit quas culpa mollitia iusto dolorem voluptatem corrupti?



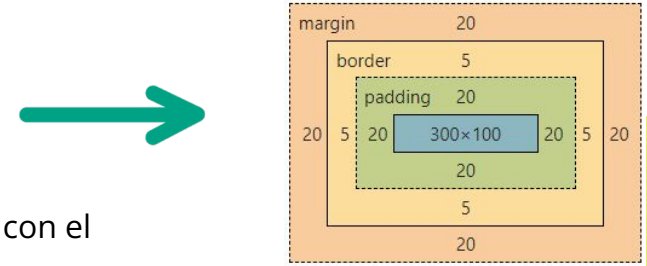
Box-sizing: border-box;

Aplicando border-box, ambos respetan el mismo width y height.
Se comprime el contenido.

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Doloremque consequatur ducimus vitae sit quas culpa mollitia iusto dolorem voluptatem corrupti?



Lorem ipsum dolor sit, amet consectetur adipisicing elit. Doloremque consequatur ducimus vitae sit quas culpa mollitia iusto



Recordatorio: usen border-box pero cuidado con el padding y con ponerle un height en px a los elementos porque pueden quedar así 🤣

Reset

Lo primero que escribiremos al comenzar una hoja de estilos (después de importar las fonts) son algunos ajustes sobre propiedades que ya vienen con un valor por defecto para todos los elementos.
box-sizing: seteamos border-box para todos los elementos.
margin y padding: indicamos el valor 0 para quitar los espaciados que tienen por defecto algunos elementos.

styles.css

```css
/* usamos el selector universal para afectar a todos los elementos */
* {
  box-sizing: border-box;
  margin: 0;
  padding: 0;
  scroll-behavior: smooth;
}
```

Bonus track: con scroll-behavior logramos que, al navegar con anclas por la página, esta se desplace suavemente.

CÓDIGO
PA' LANTE

Propiedad "Display"

CÓDIGO PA' LANTE

Cómo ocupan el espacio las "cajas"

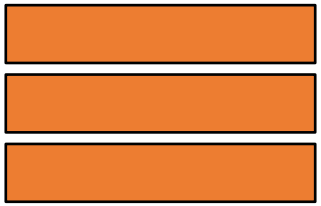
Seguramente ya habrán notado que algunos elementos HTML ocupan todo el espacio disponible a lo ancho (como los <p>, <section>, <h1>) y otros no (<a>, <img>). Esto se debe a la propiedad **display y el valor que reciben por defecto algunos elementos.**
Podemos modificar este comportamiento de los elementos HTML cambiando el valor de la propiedad.

CÓDIGO PA' LANTE

Tipos de display

- **block**: el elemento ocupa el 100% del espacio horizontal disponible, generando un salto de línea con respecto a los elementos que lo rodean. Se le pueden aplicar width, height, padding y margin. Valor por defecto en **<p>, <h1>...<h6>, **
- **inline**: ocupa solo el espacio de su contenido, no genera un salto de línea. No se le aplican width o height, y padding o margin solo se le aplican en sentido horizontal. Valor por defecto en **<a>**
- **inline-block**: funcionan como el display inline, pero se le pueden aplicar width, height, padding y margin. Valor por defecto en **<img>**

display: block;



display: inline;

display: inline-block;

Además, si aplicamos el valor **none** en esta propiedad, el elemento no se va a mostrar.

CÓDIGO PA' LANTE

Problemas comunes

Tener en cuenta el valor por defecto de display en los elementos es importante por casos como este: Estoy maquetando enlaces (que tienen por defecto inline-block) y quiero ponerles width o un margin para separarlos entre sí. Si no les modifico primero el valor de display, **¡no me van a hacer caso!**

Enlace 1 Enlace 2 Enlace 3

```
a {
    display: inline-block;
    margin: 20px;
}
```

CÓDIGO
PA' LANTE

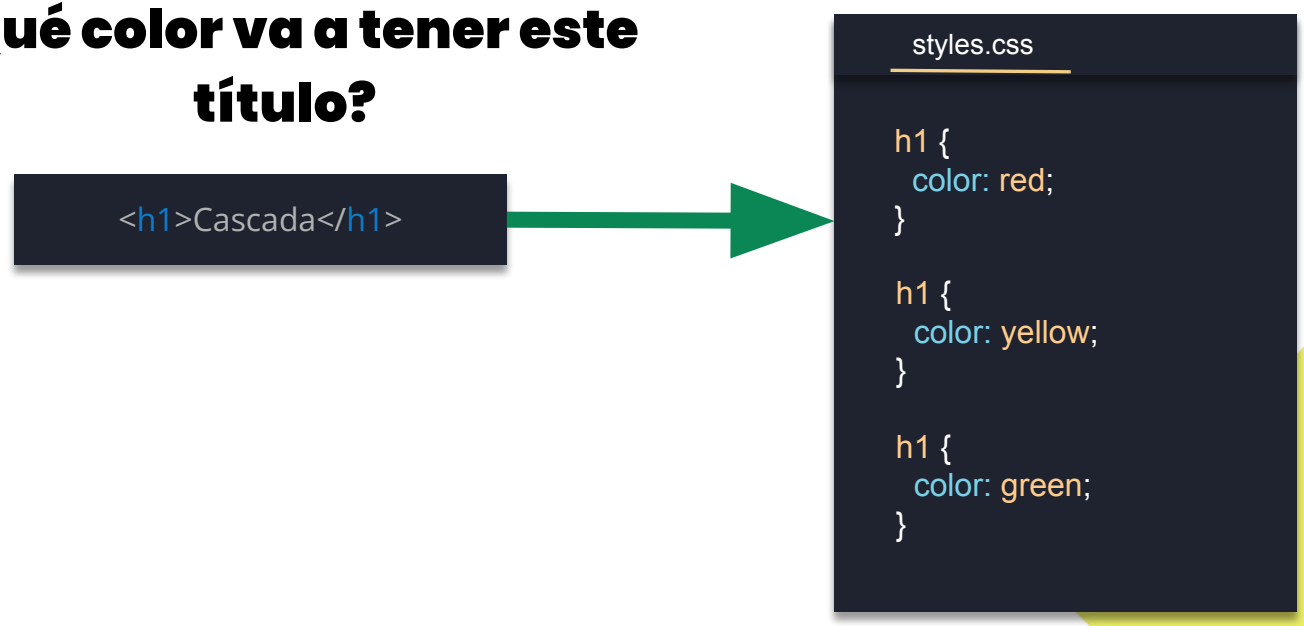
Cascada, especificidad y herencia

CÓDIGO PA' LANTE

¡Juguemos!

¿Qué color va a tener este título?

```
<h1>Cascada</h1>
```

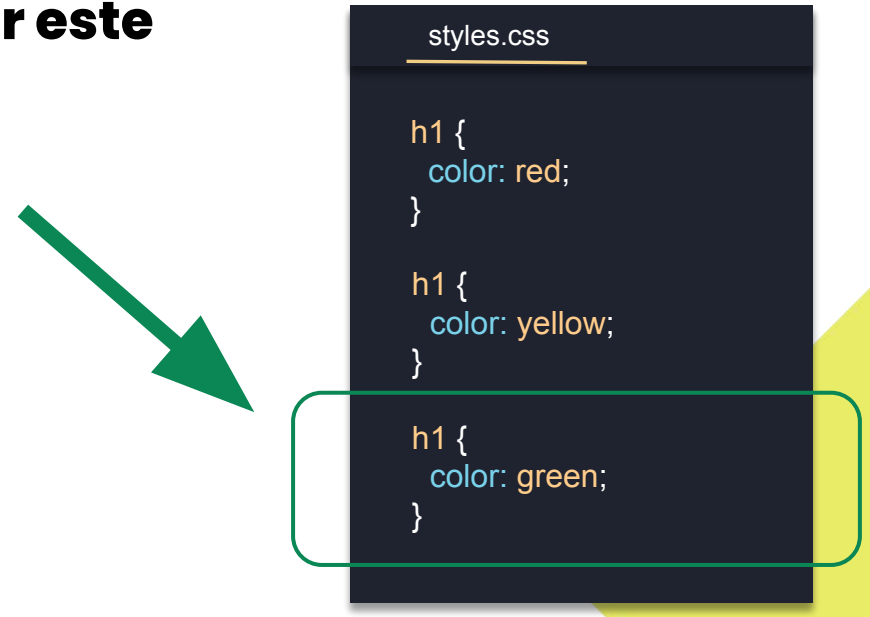


CÓDIGO PA' LANTE

¡Juguemos!

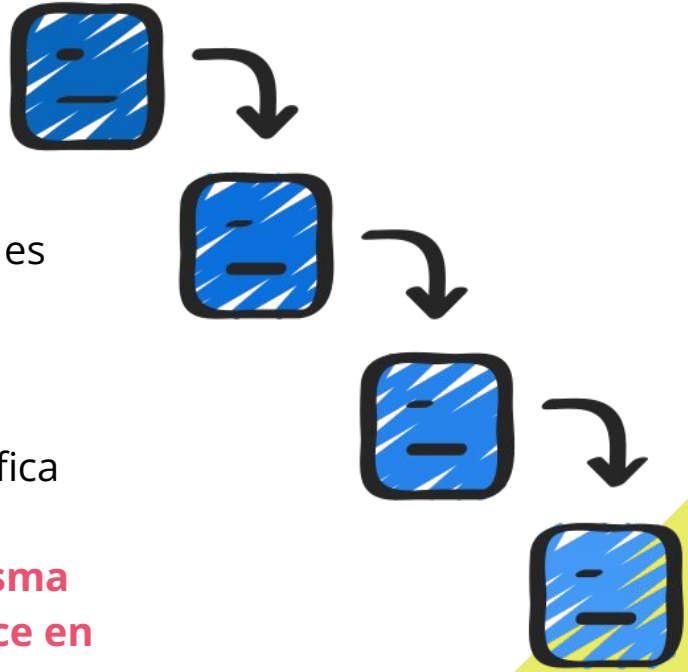
¿Qué color va a tener este título?

Cascada



CÓDIGO PA' LANTE

Cascada



CSS significa **hojas de estilo en cascada** (cascading style sheets), y es muy importante entender la palabra *cascada*.

La **cascada** en las hojas de estilo significa que el orden de las reglas importa en CSS: **cuando dos reglas tienen la misma *especificidad*, se aplica la que aparece en último lugar en el CSS.**

CÓDIGO PA' LANTE

Continúa el desafío...

¿Qué font va a tener este párrafo?

```html
<p class="parrafo-ejercicio"
id="parrafo-ejercicio">
    Lorem ipsum dolor sit amet consectetur,
     adipisicing elit. Illum, sed!
</p>
```

styles.css

```css
.parrafo-ejercicio {
    font-family: 'Courier New', monospace;
}

#parrafo-ejercicio {
    font-family: Arial, sans-serif;
}

p {
    font-family: 'Times New Roman', serif;
}
```

CÓDIGO PA' LANTE

Continúa el desafío...

¿Qué font va a tener este párrafo?

Lorem ipsum dolor sit amet consectetur, adipisicing elit. Illum, sed!

styles.css

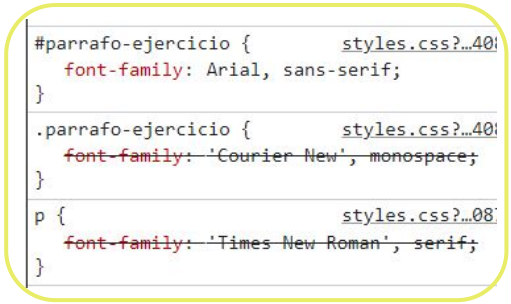
```
.parrafo-ejercicio {
    font-family: 'Courier New', monospace;
}

#parrafo-ejercicio {
    font-family: Arial, sans-serif;
}

p {
    font-family: 'Times New Roman', serif;
}
```

CÓDIGO PA' LANTE

Especificidad



En el inspector podemos ver que ganó el selector por id, luego seguía el selector por **clase**, y finalmente el menos específico, el selector por etiqueta.

La especificidad es el modo que tiene el navegador de decidir qué regla se aplica si diversas reglas tienen selectores diferentes pero podrían aplicarse a un mismo elemento.

Los distintos selectores tienen diferente jerarquía, aquí vemos un ejemplo con selectores simples.

CÓDIGO PA' LANTE

Última prueba...

¿Qué estilos va a tener el primer <p>?

¿ Y el segundo?

```
<article>
  <p>
    Lorem ipsum dolor sit amet consectetur,
    adipisicing elit. Illum, sed!
  </p>
  <p class="ejemplo">
    Lorem ipsum dolor sit amet consectetur,
    adipisicing elit. Illum, sed!
  </p>
</article>
```

styles.css

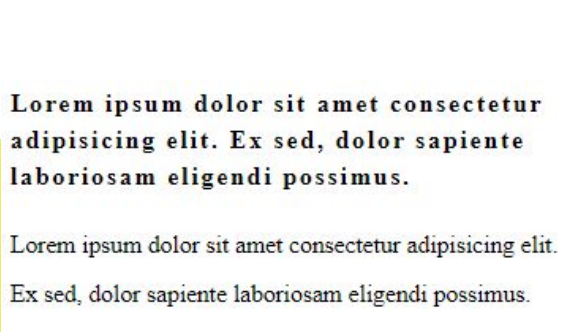
```
article {
  line-height: 1.5;
  font-weight: 700;
  letter-spacing: 2px;
}

p.ejemplo {
  line-height: 2;
  font-weight: 400;
  letter-spacing: initial;
}
```

CÓDIGO PA' LANTE

Última prueba...

¡Ambas son correctas... Pero ¿Por qué?🤔



styles.css

```
article {
    line-height: 1.5;
    font-weight: 700;
    letter-spacing: 2px;
}

p.ejemplo {
    line-height: 2;
    font-weight: 400;
    letter-spacing: initial;
}
```

primer <p>

segundo <p>

CÓDIGO PA' LANTE

Herencia

Algunos valores de propiedades CSS pueden "heredarse" de elementos padre a elementos hijos. Por ejemplo, si para un elemento se define un color de letra y una font, los elementos que se encuentren dentro de éste se van a mostrar con esos valores, a menos que tengan un valor diferente en estas propiedades.

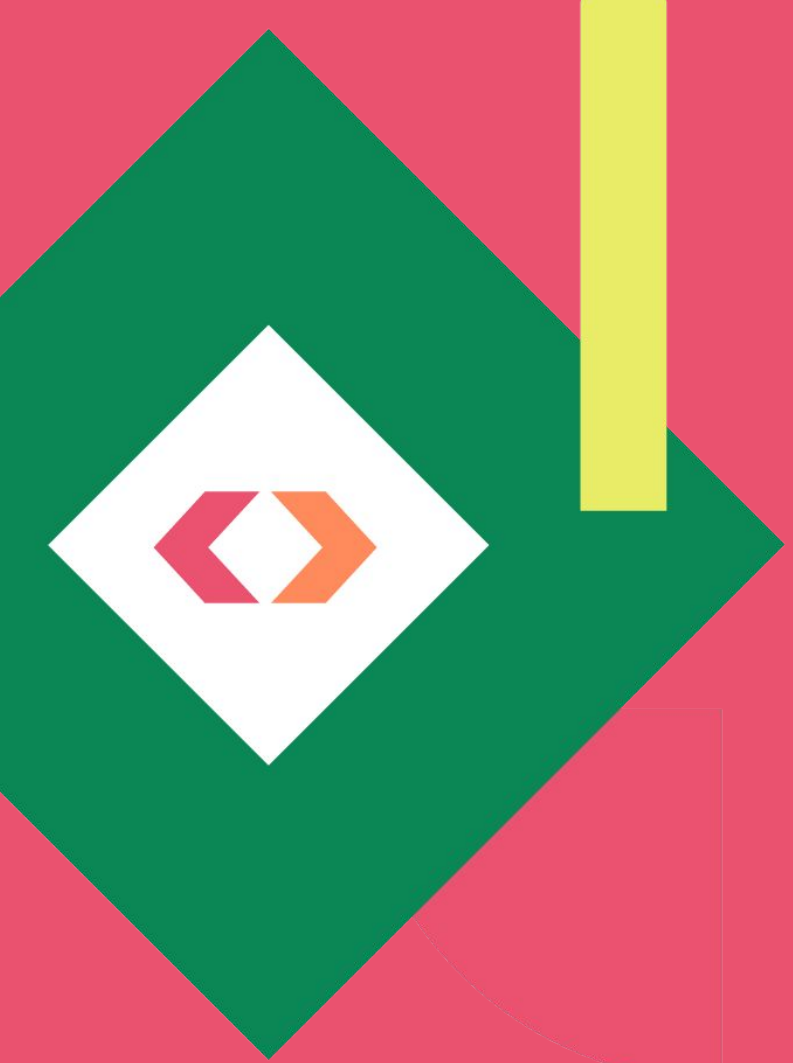
👉¿Qué propiedades se heredan y cuáles no?

El primer párrafo hereda los valores del article en esas propiedades, sin embargo, el segundo p tiene otros valores especificados!

styles.css

```
article {
    line-height: 1.5;
    font-weight: 700;
    letter-spacing: 2px;
}


p.ejemplo {
    line-height: 2;
    font-weight: 400;
    letter-spacing: 0;
}
```



CÓDIGO
PA' LANTE

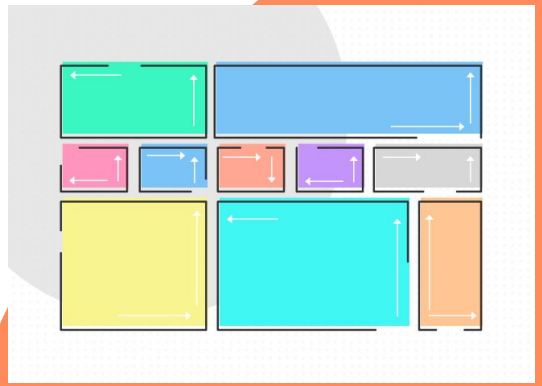
Flexbox

CÓDIGO PA' LANTE

¿Cómo centrar un elemento?

Algo que debería ser muy sencillo, en realidad era bastante complejo de hacer en CSS... hasta que llego Flexbox. 🙌

Se trata de un conjunto de propiedades CSS que nos facilitan la tarea de alinear, distribuir y organizar visualmente los elementos HTML en nuestra página.



CÓDIGO PA' LANTE

Propiedades Flexbox

Las propiedades y valores que conoceremos son:

Propiedad	**Valores**
display	flex
flex-direction	row ó column
justify-content	center / flex-start / flex-end / space-around / space-between / space-evenly
align-items	center / flex-start / flex-end

CÓDIGO PA' LANTE

¿Cómo aplicamos flexbox?

1

Definir como **"contenedor flex"** a un elemento, para luego poder manipular a los elementos en su interior.

```
div {
    display: flex;
}
```

1

2

Establecer la dirección en la cual se organizaran los ítems en el interior del contenedor (fila o columna). Esto va a determinar cuál es el eje principal y cuál el eje secundario.

Si el valor de **flex-direction es row** el **eje principal es el horizontal** y el secundario el vertical.
Si el valor de **flex-direction es columna**, el **eje principal es el vertical** y el secundario es el horizontal.

```
div {
    display: flex;
    /* row o column */
    flex-direction: row;
}
```

2

CÓDIGO PA' LANTE

¿Cómo aplicamos flexbox?

3

Usaremos **justify-content** y **align-items** para distribuir los elementos.
Es importante tener en cuenta si el eje principal es el horizontal o el vertical y lo mismo sobre el eje secundario.
Con la propiedad **justify-content** organizaremos los elementos en el **eje principal** y con la propiedad **align-items** sobre el **eje secundario.**

3

```
div {
    display: flex;
    /* El eje principal es horizontal */
    flex-direction: row;
    /* Eje principal */
    justify-content: center;
    /* Eje secundario */
    align-items: center;
}
```

Para ver en detalle qué valores podemos usar con estas propiedades, consultar el material complementario en el campus.

CÓDIGO PA' LANTE

Dos ejemplos, dos desafíos

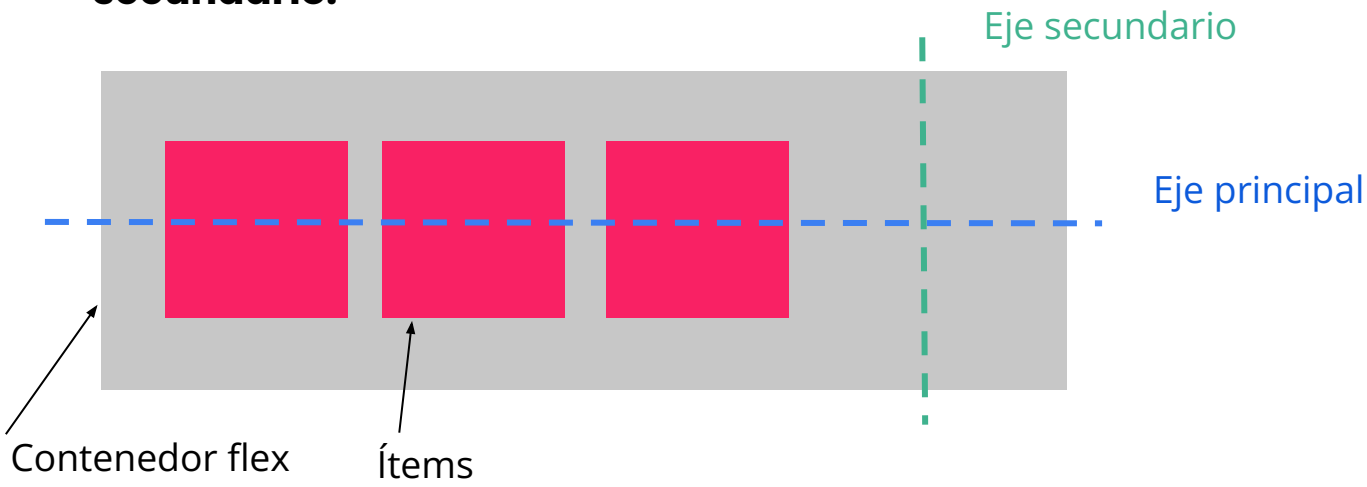
Acompáñennos a estos CodePen donde aplicaremos flexbox para resolver problemas cotidianos en desarrollo Front End.💪

- Primer encuentro con Flexbox, centramos un div: CodePen
- Segundo round: colocamos texto al lado de una imagen: CodePen

CÓDIGO PA' LANTE

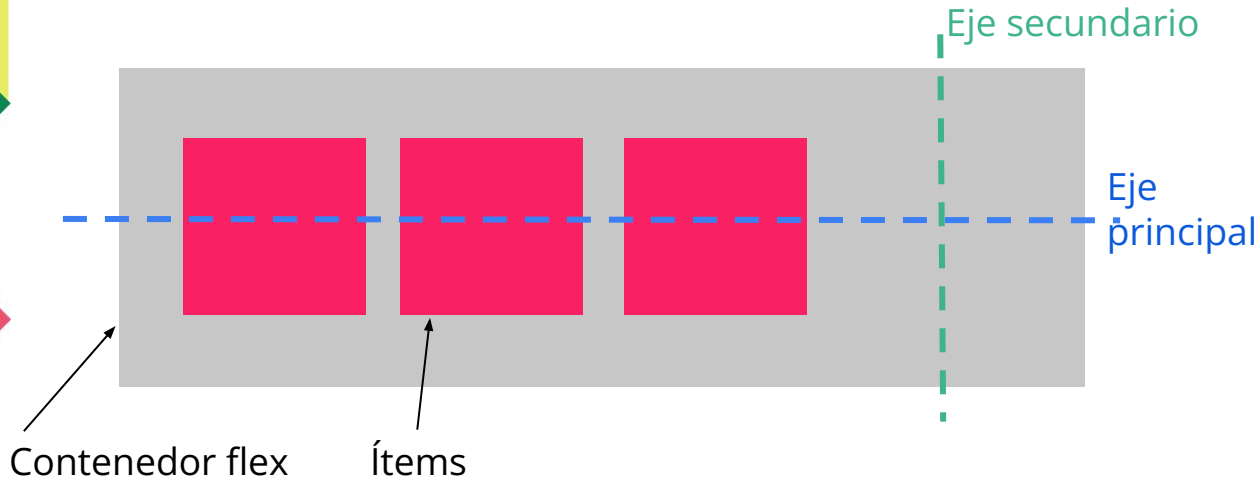
Flexbox: conceptos clave

Para poder usar flexbox tendremos que tener en claro algunos conceptos como: **contenedor flex, ítems, ejes principal y secundario.**



Flex-direction: row

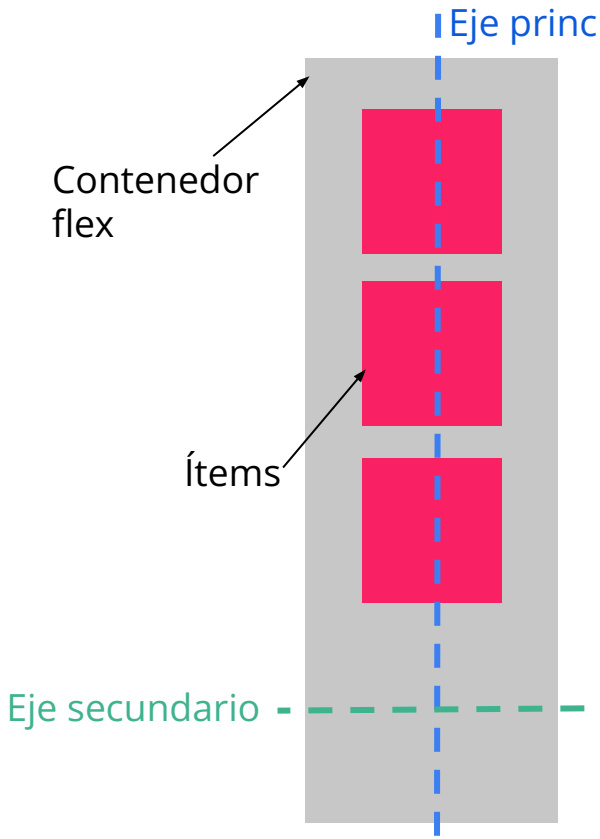
El eje principal es el horizontal, el secundario es el vertical.



CÓDIGO PA' LANTE

flex-direction: column

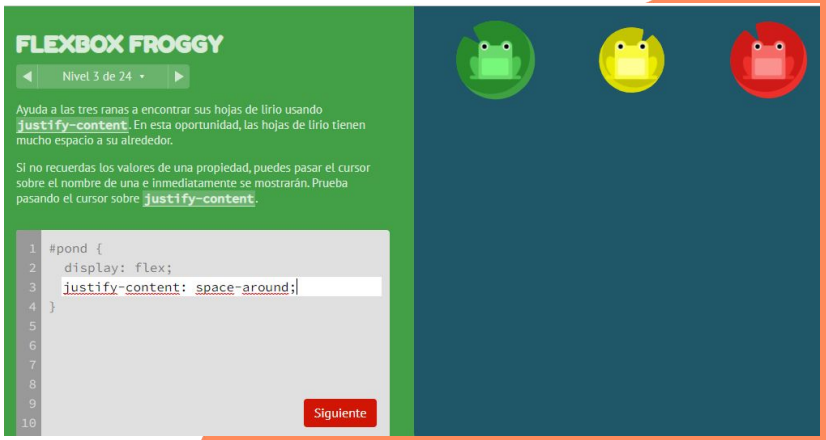
El eje principal es el vertical, el secundario es el horizontal.



CÓDIGO PA' LANTE

Aprender jugando

Flexbox es algo complejo para comprender.
Sin embargo, muchas personas terminamos de entenderlo gracias a este juego 😅😅



CÓDIGO
PA' LANTE

CSS: Responsive

Parte 1

CÓDIGO PA' LANTE

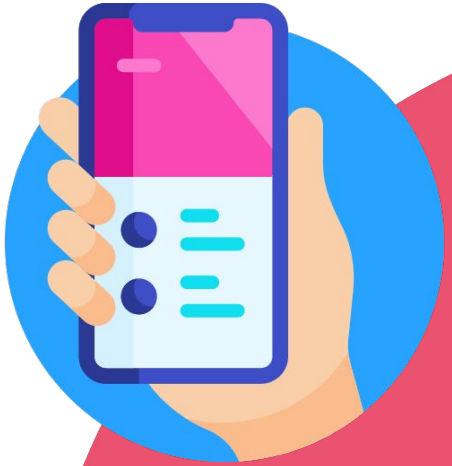
Responsive

Los sitios web pueden ser visualizados desde dispositivos de diferentes tamaños, por lo cual es necesario programarlos para que se vean y funcionen bien en todos los casos.

Podemos ingresar desde una computadora y un celular a los sitios que visitamos habitualmente y comprobar que se ven distintos.

Mobile first

Actualmente, este es el enfoque que se da al diseño y desarrollo web: **primero se diseña y desarrolla para mobile (celular y tablets) y luego desktop (notebooks, escritorio).**
Esto se debe a que los sitios web son mayormente visitados desde dispositivos móviles.



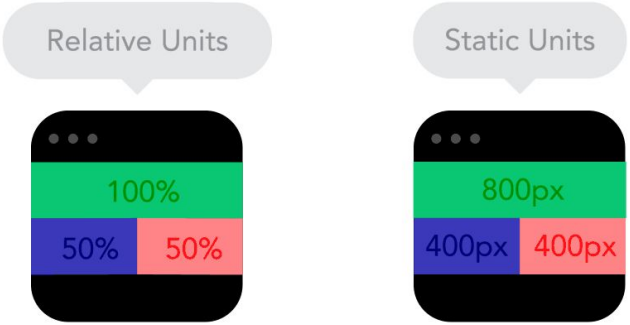
CÓDIGO PA' LANTE

Técnicas responsive

A continuación, veremos algunas técnicas que podemos aplicar en CSS para que el contenido de nuestra web pueda verse lo mejor posible tanto en mobile como en desktop.

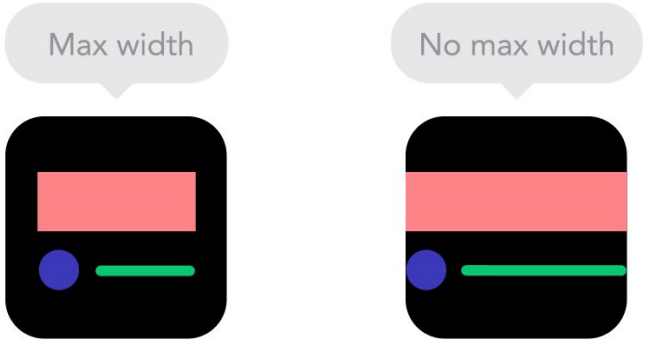
Unidades fijas vs unidades relativas

El uso de unidades relativas(%, vw) en el width de los elementos ayuda a que respondan a los cambios de tamaño.



Uso de la propiedad max-width

El uso de width 100% combinado con un max-width en pixel, previene que el contenido se "estire demasiado" en tamaños desktop grandes.



width: 100%
max-width: 800px

CÓDIGO PA' LANTE

Imágenes responsive

Para que una imagen tenga un ancho máximo, pero no se rompa en caso de que el tamaño de la pantalla sea más chico, podemos combinar también las propiedades **width y max-width.**

styles.css

```
img {
  width: 100%;
  max-width: 500px;
}
```

CÓDIGO PA' LANTE

Imágenes responsive 2

El enfoque anterior es útil, sin embargo, genera que la imagen al aumentar su ancho también automáticamente aumente en alto, lo cual puede no funcionar en ciertos casos.
Esta segunda opción usando las propiedades **height y object-fit** resuelve ese problema.

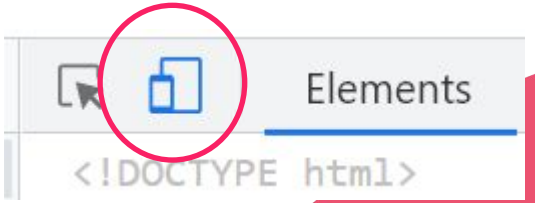
styles.css

```
img {
  width: 100%;
  height: 300px;
  object-fit: cover;
}
```

CÓDIGO PA' LANTE

Siempre a mano las devtools

Las herramientas para desarrolladores del navegador nos permiten visualizar el sitio simulando distintos tamaños de dispositivos para poder trabajar nuestro CSS.
Recuerden que las abrimos presionando **ctrl+shift+i** o **click derecho -> inspeccionar.**

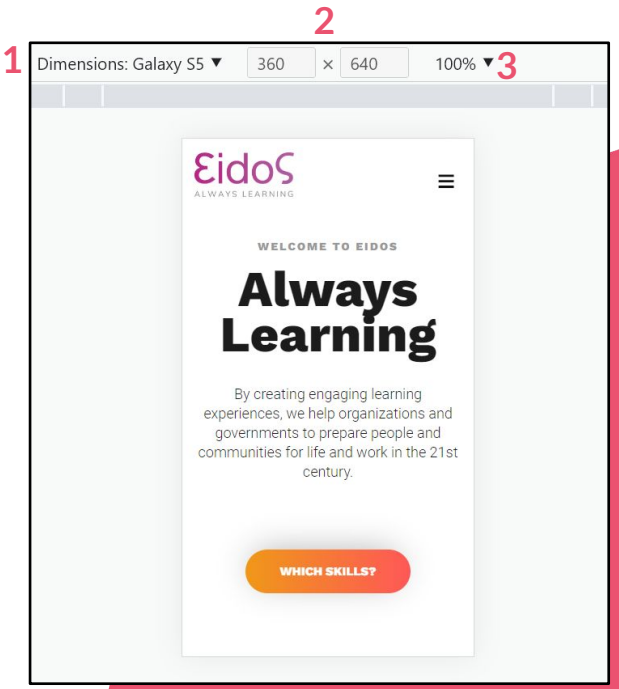


Tienen que clickear en este ícono en la esquina superior izquierda del inspector para ver el sitio en versión mobile.

CÓDIGO PA' LANTE

Herramientas para hacer CSS responsive

1- Lista desplegable con diferentes dispositivos desde celulares hasta tablets
2- Ancho y alto en px
3- Zoom



CÓDIGO
PA' LANTE

Recursos

CÓDIGO PA' LANTE

Enlaces útiles para consultar

- Sobre unidades de medida
- Cascada, especificidad y herencia
- Selectores
- Box model
- Flexbox

CÓDIGO
PA' LANTE

Aprender a aprender

CÓDIGO PA' LANTE

Estudiar código

¿Cómo se estudia en programación?
Probablemente no haya una sola respuesta, pero aquí les compartimos algunas estrategias útiles, en particular, para tomar apuntes:

1. **Armar tutoriales** de cómo hacer algo específico que me sirvan para fijar los pasos: por ejemplo, cómo iniciar un proyecto web en el VSC.

2. Armar **"machetes" de código con comentarios**, por ejemplo, qué estamos escribiendo en cada línea.

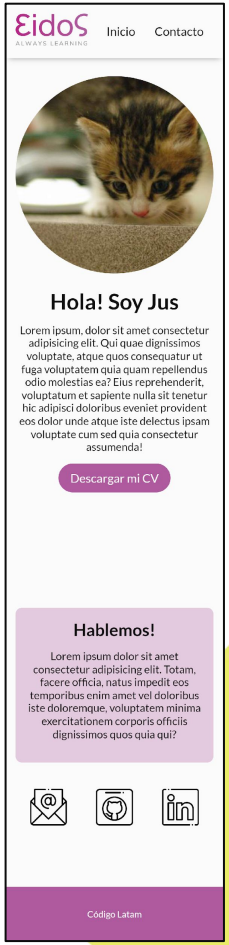
CÓDIGO PA' LANTE

Proyecto 1

¡Hemos llegado al fin de la primera etapa del curso! Les proponemos un pequeño proyecto para aplicar todo lo que aprendimos sobre HTML y CSS en estas semanas.
La idea es construir una página muy similar a la página de "presentación personal" pero, esta vez, a partir de un diseño que van a tener que replicar. En esta clase les presentaremos la consigna.

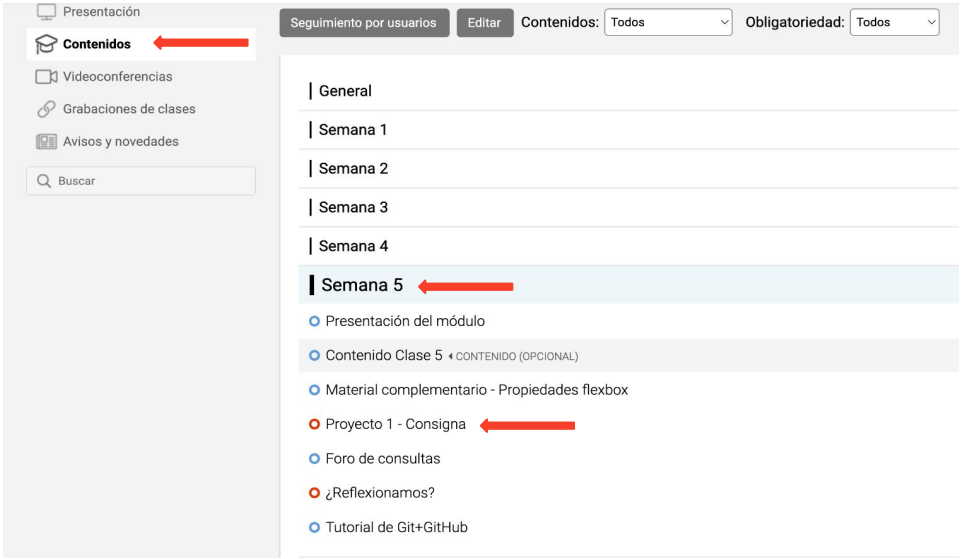
Cómo debe verse el proyecto

Este es el diseño del proyecto, se trata de una **One Page** con dos secciones: Inicio y Contacto.
En la consigna que encontrarán en el campus pueden encontrar especificaciones de colores, fonts, tamaños, etc.
En este primer proyecto realizaremos la versión mobile.



¿Cómo empiezo?

Descargar del campus la consigna del **"Proyecto 1"**



Git + GitHub: en la sección “Semana 5” el campus encontrarán el tutorial para instalar y utilizar estas tecnologías.

Sobre la consigna

En la consigna encontrarán:

- La consigna detallada en PDF
- Imagen del diseño
- Una carpeta llamada "proyecto-1-cod-latam" con los archivos HTML y CSS listos para que comiencen a trabajar. Úsenlos como punto de partida.

¿Tengo que comenzar a codear ya?

No es necesario que comiencen a codear el trabajo.

Lo que deberán hacer esta semana es:

Tener descargada la carpeta con la consigna y los archivos de base
Realizar el tutorial de Git+GitHub
Ver el contenido de clase 6 en el campus

Clase 6: tutoría

La Clase 6 la dedicaremos al proyecto,
con livecodings y consejos para empezar a trabajar.
Si ya descargaron todo, leyeron la consigna, vieron el material y no saben por dónde comenzar, pueden esperar a la Clase 6, en la cual resolveremos todas sus dudas. 💪

Calendario de cursada

Completar esta slide con la fechas correspondientes:

- Semana del 17/4: Clase de tutoría (Clase 6)
- **Fecha de entrega del proyecto: 30 de abril de 2023**