# Que es wireframe

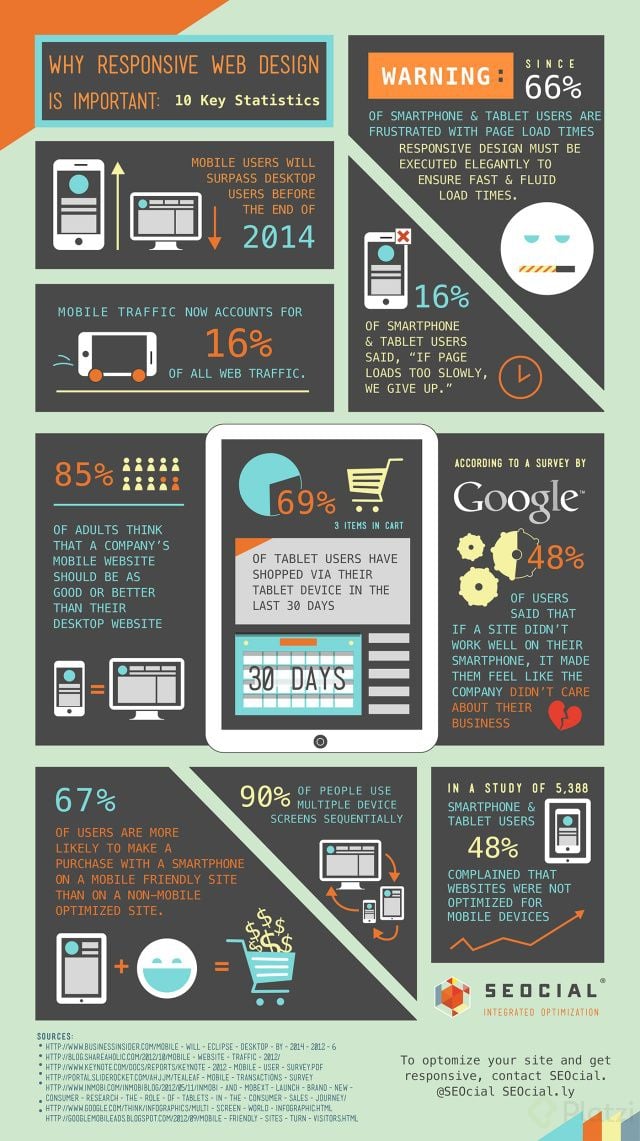
También llamado plan de pantalla por algunos profesionales, el wireframe es un prototipo de página web o de una aplicación. Esto significa que, antes de la elaboración del layout, se crea una especie de esquema. De esta manera, conseguimos entender cómo quedará el producto final

# Benefiaciarios

* **Para desarrolladoras y desarrolladores: escalar es más sencillo**
* **Para usuarios: menos es más y para más personas**
* **Para negocios: mejor posicionamiento en buscadores**

Google comenzó trabajar a inicios de 2018 con un algoritmo que otorga de mayor relevancia a aquellos sitios optimizados para móviles. Esto no afectará a aquellos sitios que tengan una versión de escritorio y móvil, pero sí penalizará a los que carezcan de una alternativa móvil.

Para fines de SEO esto puede significar un menor rebote de usuarios si el contenido es lo suficientemente atractivo para retener la atención de los usuarios.



La arquitectura web se puede definir como la forma en que las páginas de un sitio web están estructuradas y enlazadas entre sí (de manera lógica y coherente). Una arquitectura web ideal ayuda a los usuarios y a los motores de búsqueda a encontrar fácilmente lo que están buscando en un sitio.

Los assets son todos los **recursos estáticos** que utilizaremos en nuestro proyecto, tales como:

Imágenes  
Logotipos/Isotipos (Si el logotipo y el nombre de la empresa/marca se encuentran en elementos separados, es preferible descargarlos agrupados)  
Íconos  
**PRO TIP**: podemos descargar varios assets simultáneamente. Solo debemos seleccionar varios elementos, presionando CTRL + Clic sobre el elemento, y luego presionamos Export seleccionamos el formato, ¡y Listo! nuestros assets se descargaran en un archivo .zip

Clasificamos los assets según el tipo de recurso (imagen, ícono, etc…), además es recomendble que sigamos un estándar para nombrar nuestros archivos (nombre corto, desciptivo, en minúsculas, sin espacios) .  
Finalmente la estructura de carpetas de nuestro proyecto debería quedar de esta forma:



# Fuentes

Por buenas prácticas se debe manejar máximo dos fuentes en el proyecto.

Hay diversos tipos de fuentes que componen el diseño un proyecto. Recuerda que como parte de las buenas prácticas en el desarrollo web, no debes tener más de dos fuentes. En caso de encontrar más de dos, tendrás que comunicarte con el equipo de diseño para llegar a una conclusión sobre cuáles elegir.

## Cómo insertar fuentes en un proyecto

Para linkear el enlace que copiamos en el paso anterior, abrimos nuestro archivo index.html y vamos hasta hasta la sección del head. Pegamos el enlace justo debajo de la etiqueta title.

El enlace generado para el proyecto en que estamos trabajando es el siguiente:

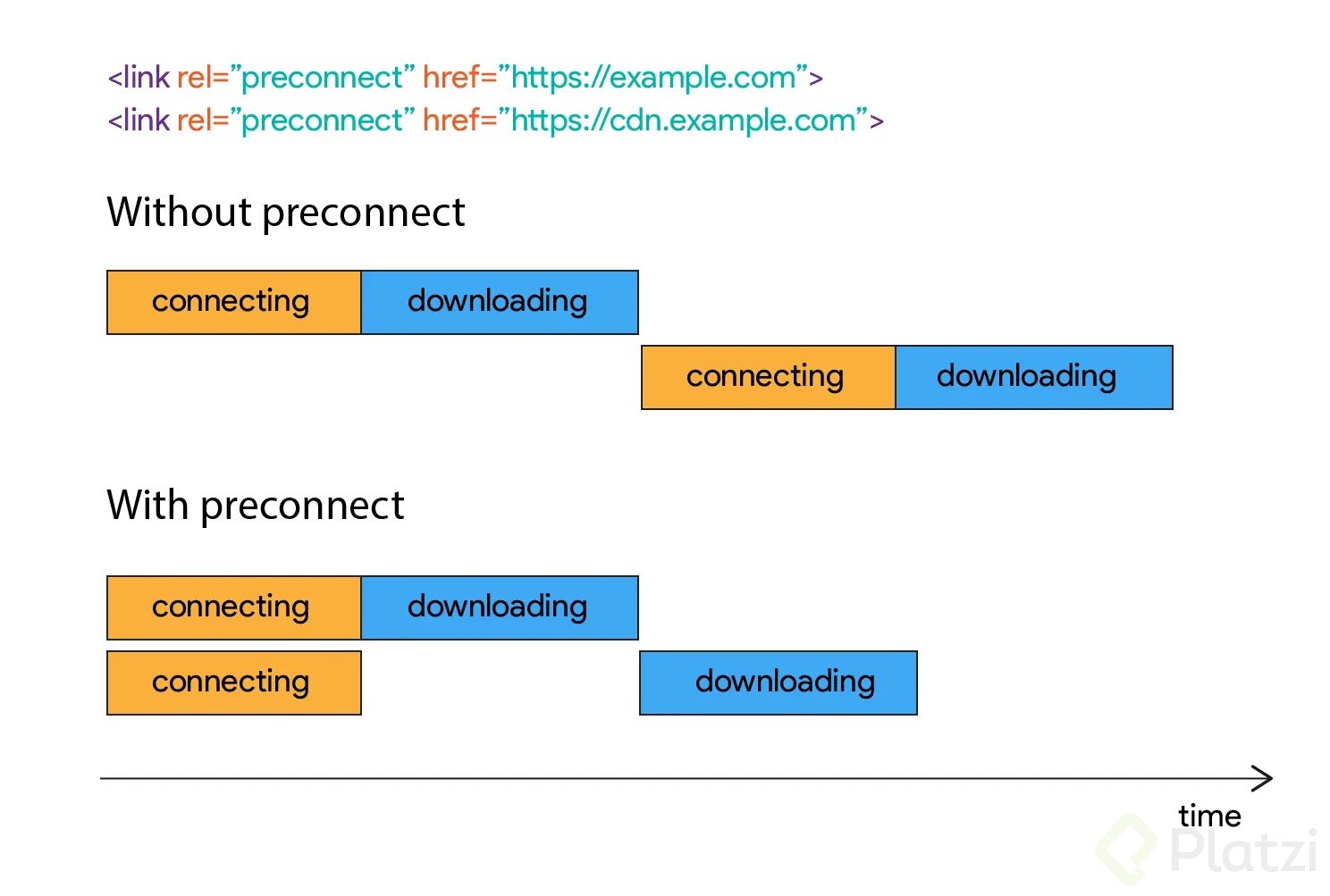
<**link** rel="preconnect" href="https://fonts.googleapis.com">

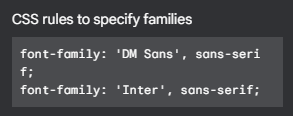
<**link** rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<**link** href="https://fonts.googleapis.com/css2?family=DM+Sans:wght@400;500;700&family=Inter:wght@300;500&display=swap" rel="stylesheet">

**Nota**: El tag de link con el valor de atributo **rel=“proconnect”** permite notificarle al navegador que la página requiere establecer una conexión a otro dominio y que esta debe realizarse de la forma más rápida posible. De esta forma cuando el navegador requiera hacer uso de los recursos (en este caso los tipos de fuente), la descarga de recursos será más rápido porque la conexión ya existirá de forma previa. Esto permite mejorar el performance de la página.

Recuerda que Google mismo te indica, justo debajo del link que te da, cómo llamar la familia de la fuente que seleccionaste.





¡Y listo! Ya podemos empezar empezar a crear nuestros primeros estilos.

Hey amigo dev…vengo del futuro🤖

Vengo a hacer un par de recomendaciones:

•Aprende a utilizar muy bien BEM (metodología explicada en este curso ya que lo necesitaras en tu futuro empleo👷)

[Sitio para aprender bem](https://es.accentsconagua.com/articles/webdesign/an-introduction-to-the-bem-methodology.html)

•📱Cuando este en un proyecto y estes prototipando, empieza desde la versión movile, (mi error fue ese, no comenzar por ahí) así te ahorraras mucho tiempo.

💣Extra: Aprende CSS grid, eso será la Bomba para acelerar y crear sitios increíbles 😃 💣

[Curso de CSS GRID 2021](https://platzi.com/cursos/css-grid-layout/)

PD: Te sirvió?



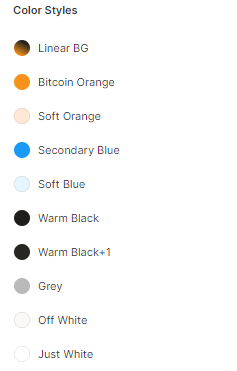
**Primeros estilos**

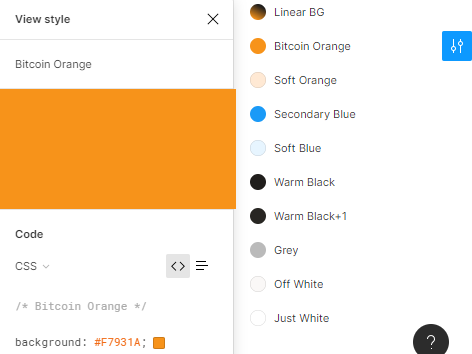
## Cómo hacer el posicionamiento de contenido

Como parte de las buenas prácticas tenemos un orden en que es recomendable escribir el código dentro de CSS. Te ayudarán a tener todas las secciones organizadas y saber dónde regresar cuando necesites hacer un cambio o solucionar un problema.

1. **Posicionamiento** --> static, absolute, relative, fixed.
2. **Modelo de caja** (Box model) --> margin, border, padding, content.
3. **Tipografía** --> tipos, tamaños de fuente, etc.
4. **Estilos visuales** --> box-shadow, border-radius, gradient, etc.
5. **Otros** --> misceláneos, reglas CSS y más.

## Cómo crear variables

En la sección de Branding del diseño en Figma podemos ver todos los colores que se usan. Por ahora, las variables que vamos a crear van a ser de estos colores.  


Al hacer clic sobre el ícono de ajuste de cada color, se abre la información sobre el mismo. Ahí tenemos el código hexadecimal que necesitamos.  


Recordemos que para declarar una variable en CSS debemos usar la función :root y añadir un nombre a la variable que contendrá el valor que vamos a usar repetidas veces en nuestro código. Seleccionando todos los colores, tendríamos:

:root {

--bitcoin-orange: #f7931a;

--soft-orange: #ffe9d5;

--secondary-blue: #1a9af7;

--soft-blue: #e7f5ff;

--warm-black: #282623;

--black: #201e1c;

--grey: #bababa;

--off-white: #faf8f7;

--just-white: #fff;

}

Convenciones a la hora de acomodar los estilos

Orden de los estilos, estilos de:

/\*

1. Posicionamiento

2. Modelo de caja(Box model)

3. Tipografía(tamaño, familia,...)

4. Temas visuales(shadow box, border radius,...)

5. Otros(Miscelaneos)

\*/

## Nombrar variables en CSS

**El nombre no debe ser igual al valor:**  
Cuando el nombre de la variable está ligada a su valor perdemos una de sus funciones que es la versatilidad de cambiar su valor y que todo el diseño cambie con ella por ejemplo:  
Si tenemos una variable que se llama

--bitcoin-orange: #f7931a;

si luego quieres cambiar el color a un **azul**

--bitcoin-orange: #0000ff;

la variable pierde todo el sentido porque tiene en el nombre **naranja**, pero su valor es **azul**

**El orden:**

Cuando nombramos una variable es importante el orden porque cuando estamos escribiendo el código nuestro editor tratara de autocompletarnos por eso lo mejor es nombrar la variable desde su característica más general hasta la más específica para aprovechar esta característica del editor

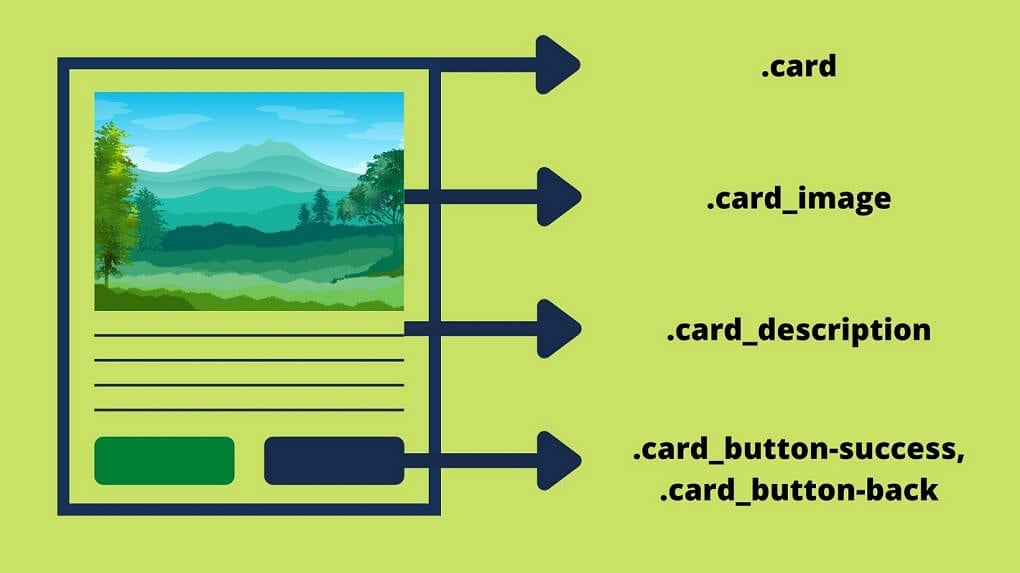
**Meh 😕**

Por buenas prácticas, una pantalla o página solo debe tener un h1 por temas de SEO. Exacto, es por SEO. Además, se recomienda que no tenga más de 50 o 60 caracteres.

**Metodología BEM**

Para nombrar las clases de los contenedores y contenidos, vamos a usar la metodología BEM. Pero para tenerlo más claro, hagamos un repaso de lo que es BEM.

## ¿Qué es BEM?

BEM (Block, Element, Modifier o Bloque, Elemento, Modificador) es una metodología ágil de desarrollo basada en componentes. Divide toda la interfaz en bloques que se pueden reutilizar y escalar.  


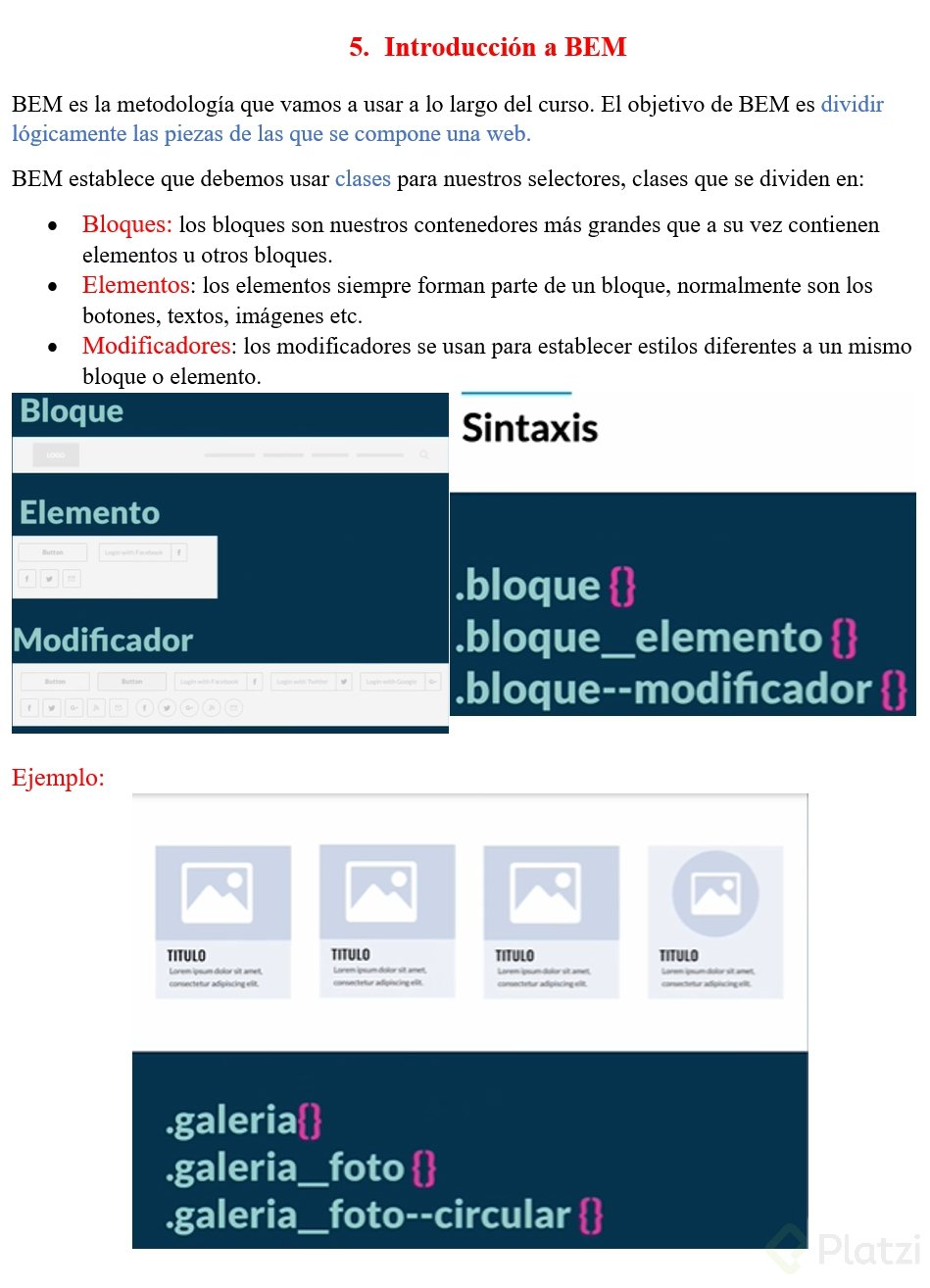
De esta manera, podemos separar una sección en múltiples secciones pequeñas y darles un nombre en este orden y función.

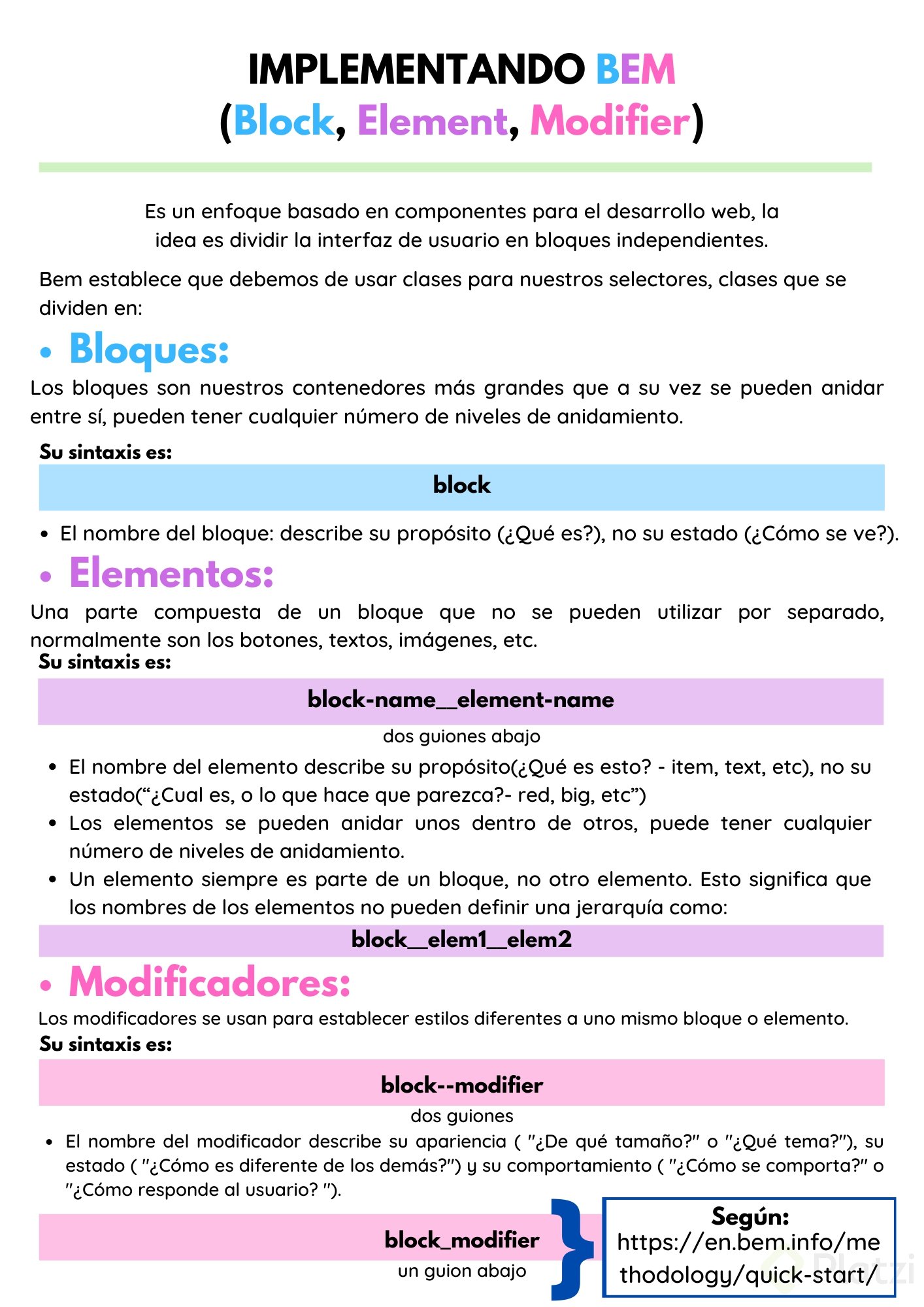
## Consejos para el nombre de las clases

* Como no va a haber otra etiqueta **img** dentro del **header**, podemos no colocarle una clase y llamarla siendo específicos.
* La etiqueta **div** es muy común, por lo que una clase es necesario. La podemos llamar así: **header–title-container** en el que **header** es el contenedor principal y **title-container** el contenido.
* Como la etiqueta **h1** es única en toda la página, tampoco es necesario colocarle una clase.
* Los botones también son comunes, así que aplicamos la clase: **header–button**.

El profesor al cambiarle la dirrección al flex con flex-direction:column hizo que a su vez el justify-content cambiara su direecion de centrado. Por este motivo en las clases de mas adelante le cuesta centrar algunos contenedores. Para evitar esto añadan en el header align-items: center con esto solucionan algunas fallas en el centrado de los contenedores.

Un dato a tener en cuenta:  
Según la metodología BEM, los **Elements** deben llevar el nombre como: block\_\_element, usando dos guiones bajos y luego el nombre. Por otro lado, los **Modifiers** deben llevar el nombre como: block--modifier, usando dos guiones y luego el nombre.  
En la clase se nombran los Elements como Modifiers, un pequeño detalle a la hora de hacer el naming de las clases.





Hola compañeros! me surge una pregunta y es  
¿Por qué usa un position relative en este caso para el header?

Lo usa para que nos sea mas fácil posicionar elementos de forma relativa. En este caso, nos sirve mucho tener al header con **position relative** para poder ajustar el enlace de “Conoce nuestros planes” de forma que quede “sobresaliendo” del header.  
.  
Si le damos a nuestro enlace un **position absolute** lo podríamos posicionar de forma relativa al elemento mas cercano que tenga algún tipo de position, en este caso, el header (con position relative)  
.  
Si quieres profundizar mas sobre el tema, te recomiendo esta página donde vienen explicados todos los positions de manera corta y entretenida: <https://learnlayout.com/position.html>

# Uso de linear gradient

    background: linear-gradient(207.8deg, #201E1C 16.69%, #F7931A 100%)



line-height: 2.6rem;

es el interlineado.

Para usar la funcion calc de css hay que separa por un espacio el signo, en este caso el menos.

left: calc(50% - 100px);