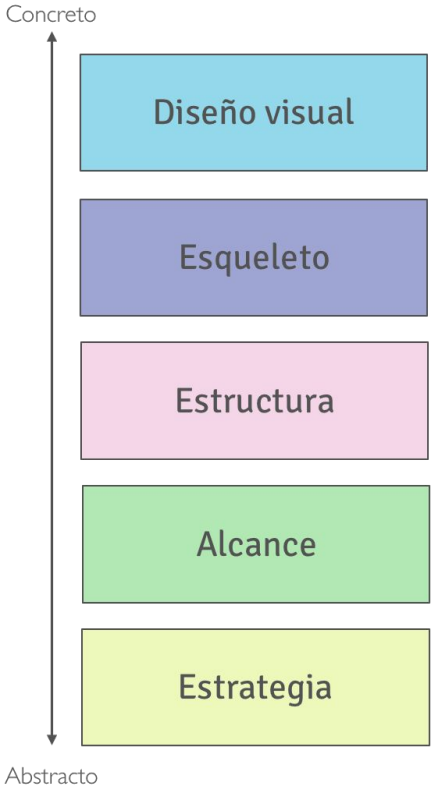
Flujo de trabajo _

Crear la experiencia de usuario



Para crear una experiencia realmente útil para un usuario es necesario pasar por un proceso que va de los más abstracto a lo más concreto.

Según Garret en el libro "Los elementos de la experiencia del usuario", la experiencia de usuario se puede entender en 5 planos primarios.

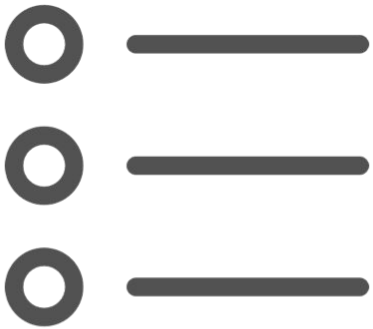
La estrategia es donde todo comienza

ESTRATEGIA

¿Qué queremos obtener del sitio?

¿Qué quieren nuestros usuarios?

El alcance transforma la estrategia en requisitos

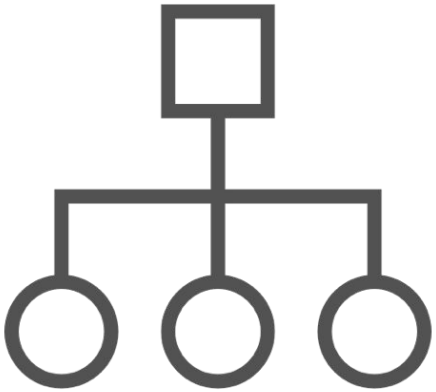


ALCANCE

¿Qué funcionalidades debe incluir el sitio?

¿Qué contenido requiere el sitio?

La estructura da forma al alcance



ESTRUCTURA

¿Cómo organizar el contenido?

¿Cómo navegará el usuario?

El esqueleto hace que la estructura sea concreta

ESQUELETO

¿Qué componentes permitirán que las personas usen el sitio?

La superficie une todo visualmente

DISEÑO VISUAL

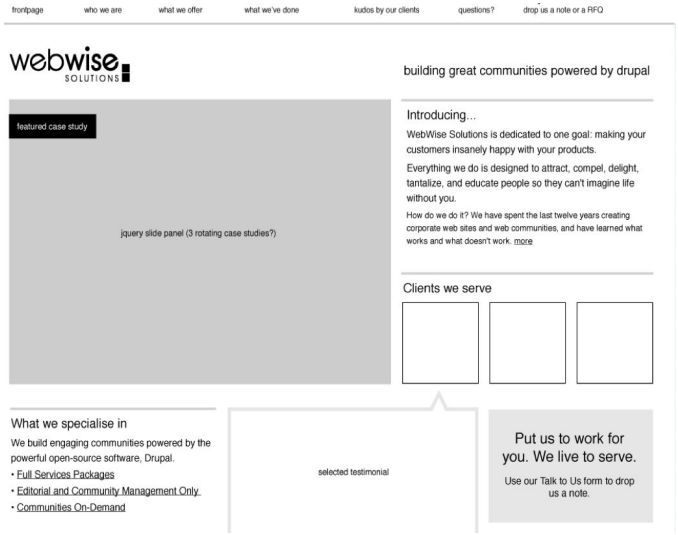
¿Cómo se verá la página terminada?

Diseñador UX

Un diseñador UX es quién investiga y analiza cómo los usuarios se sienten al usar un producto.

- Investiga la conducta del usuario.
- Analiza los resultados de la investigación del usuario.
- Informa a los miembros del equipo sus hallazgos.
- Monitorea el desarrollo del proyecto.
- Asegura que los resultados sean implementados en el proyecto.

¿Cuáles son los entregables que realiza el UX?



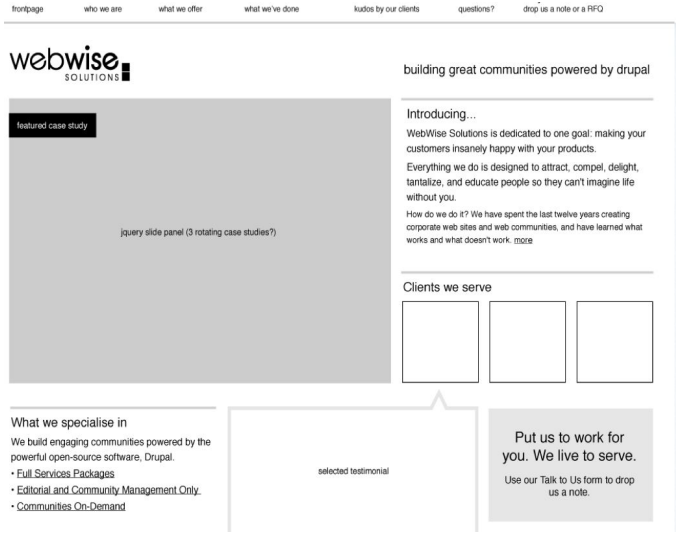
Los diseñadores UX, por lo general, entregan a su equipo toda la investigación realizada usando un representación visual llamada Wireframe.

Diseñador UI

Un diseñador UI es el encargado de diseñar las interfaces con las cuales el usuario va a interactuar.

- El diseño de la interacción (cómo responde el sistema al usuario).
- El diseño de elementos o componentes del sitio (botones, formularios, barras de navegación, etc).
- El diseño visual (iconos, imágenes, etc).
- Las guías de estilos (paletas de color, fuentes, tamaños De Fuentes, componentes UI, entre otras cosas).

¿Cuáles son los entregables que realiza el UI?



Los diseñadores UI, por lo general, entregan a su equipo una representación visual de mediana y alta fidelidad, junto con una guía de estilos.

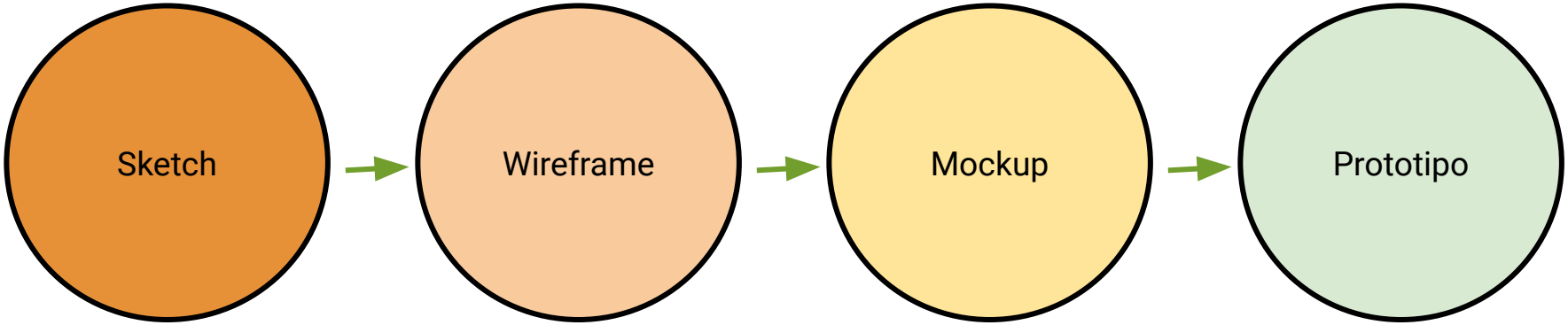
Representaciones visuales

¿Qué es una representación visual?

Las representaciones visuales podemos definirlas como la aplicación de una idea o concepto en un diseño visual.

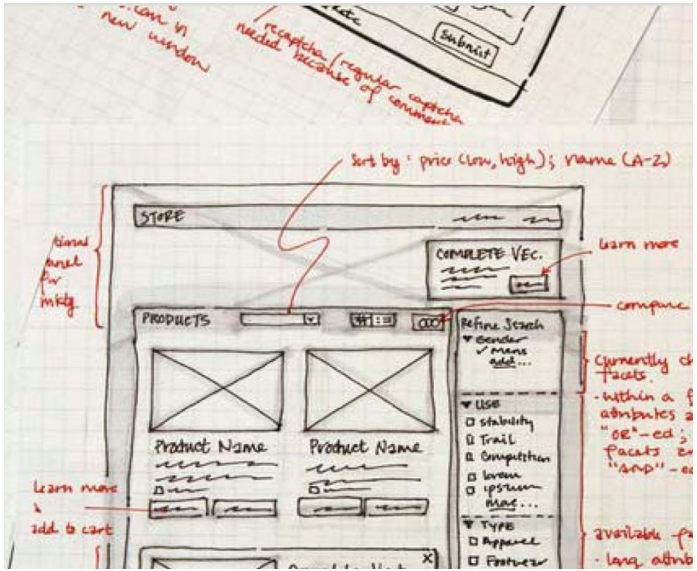
Estas se pueden segmentar dependiendo del grado de fidelidad con la cual están diseñados.

Tipos de representaciones visuales



Baja fidelidad **Media fidelidad** **Alta fidelidad**

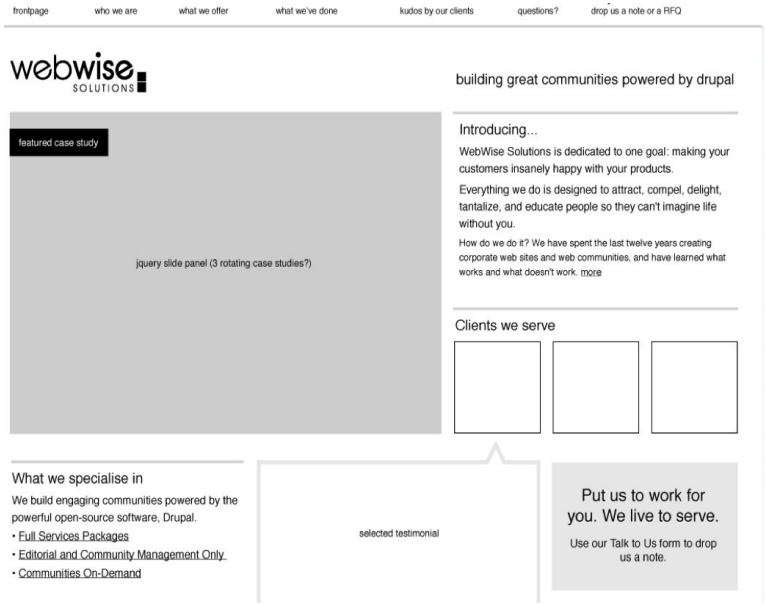
Sketch



Representación en papel de una página web o aplicación.

Es la forma más rápida de visualizar una página web.

Wireframe

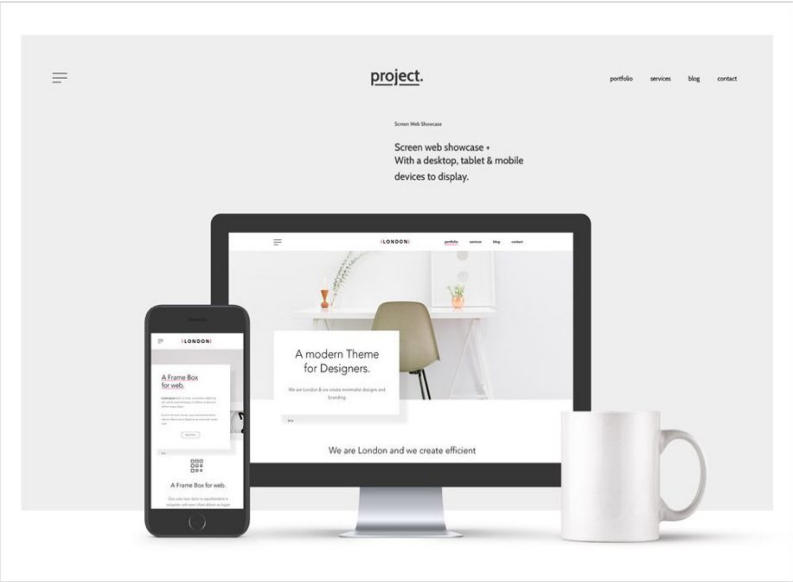


Es el equivalente a un esqueleto de una web o aplicación.

Es usado para describir estructura, contenidos y características de una web o aplicación.

Un wireframe no contiene diseño.

Mockup



Es una representación con diseño de una página web.

Junta todas las características de un wireframes, con características propias de un diseño visual (colores, fuentes, imágenes, logos, iconos, etc).

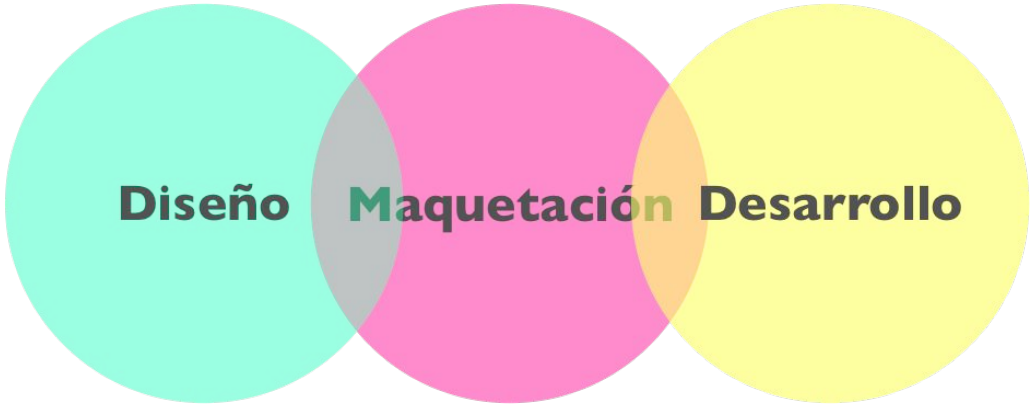
Prototipo

Un prototipo ofrece una representación de alta fidelidad de una página web o aplicación.

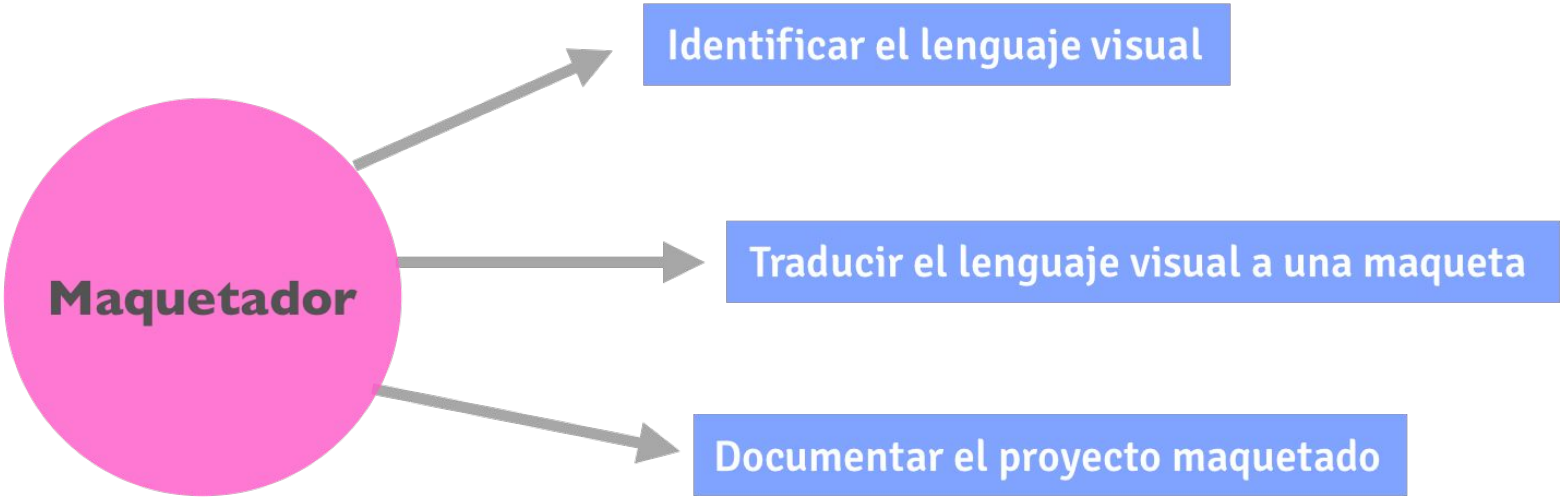
Es un mockup enriquecido con piezas de experiencia de usuario como interacciones y animaciones.

¿Qué es la maquetación?

La maquetación es la acción de traducir representaciones visuales de manera fiel en código HTML, CSS y JavaScript, es decir que deberá contener todas las características visuales y funcionales reunidas por los diseñadores UX/UI.



¿Cuál es el rol del maquetador?



Esto significa que deberemos conocer todos los elementos que componen la interfaz propuesta por el diseñador, de manera que sea más fácil decidir qué herramientas usar para crear el código de la interfaz.

¿Qué es una guía de estilos?

La guía de estilos es un entregable creado por un diseñador, el cual contiene las líneas, reglas, uso los elementos visuales incluidos dentro de una interfaz de usuario.

Traducir el lenguaje visual a una maqueta

Esto tiene que ver con algunas habilidades particulares que harán de este trabajo más organizado y fácil de escalar como:

- Separar los elementos de una interfaz en pequeñas piezas.
- Crear un sistema o seguir una metodología de trabajo.
- Usar herramientas que nos ayuden a trabajar de manera consistente.
- Seguir buenas prácticas al escribir código HTML y CSS.

Documentar el proyecto maquetado

- Este último punto es importante para poder guiar al desarrollador que implementará la maqueta al backend.
- Esta documentación deberá contener una descripción de los elementos incluidos dentro del proyecto, además de presentar una guía con algunas recomendaciones relacionadas a la estructura y estilos aplicados dentro del proyecto realizado.

Sass

¿Qué es Sass?

- Es el acrónimo de **Syntactically Awesome Style Sheets**
- Poderosa herramienta que nos ayuda a crear hojas de estilo claras, consistentes y fáciles de mantener
- Preprocesador de CSS

¿Qué es un preprocesador CSS?

- Es el acrónimo de **Syntactically Awesome Style Sheets**
- Poderosa herramienta que nos ayuda a crear hojas de estilo claras, consistentes y fáciles de mantener
- Preprocesador de CSS

Ventajas de usar un preprocesador CSS

- Ahorrar tiempo
- Separar lógica visual del sitio
- Crear hojas de estilo más claras y mantenibles

¿Por qué usar Sass?

- Compatible con todas las versiones de CSS
- Preprocesador CSS maduro
- Popularidad en la industria

https://sass-lang.com/

¿Cómo instalar Sass?

Para instalar Sass se necesita:

Instalar
node JS y NPM

Instalar
Sass vía NPM

Revisar la
instalación de Sass

Antes de instalar

Previamente debemos revisar si tenemos instalado Node JS y NPM, para ello:

1. Ingresa al terminal
2. Escribe el comando **node --version**
3. Escribe el comando **npm --version**

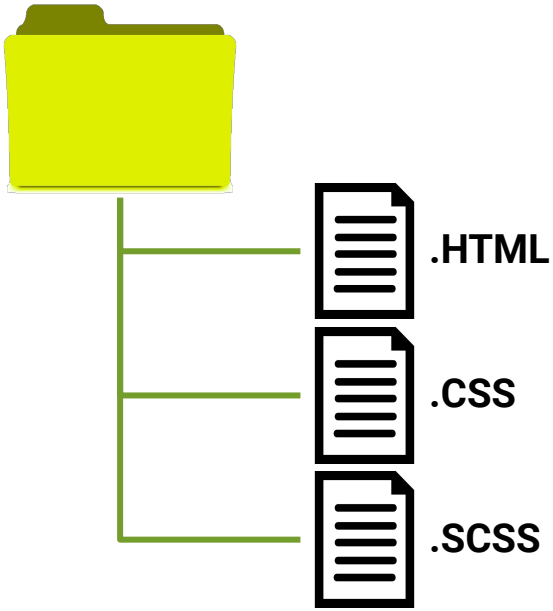
Instalar Node JS

1. Ingresar a https://nodejs.org/es/
2. Descargar Node JS v.8.12.0 o superior (versión estable)
3. Seguir los pasos de instalación
4. Verificar la correcta instalación de Node JS y NPM: https://www.npmjs.com/get-npm

Instalar Sass

1. Ingresar al terminal y escribir el comando **npm install -g sass**
2. Esperar la instalación del paquete en el sistema
3. Verificar la instalación del paquete escribiendo en la terminal **sass --version**

Estructura básica de un proyecto con Sass



1. Crear el directorio raíz del proyecto

2. Dentro de la carpeta creada anteriormente:
   - 2.1. Crear archivo **HTML**
   - 2.2. Crear archivo **CSS**
   - 2.3. Agregar un archivo compatible con **Sass**

Tipos de archivos Sass

Sass tiene 2 tipos de archivos disponibles, para este taller utilizaremos la extensión **.SCSS**

.SASS

.SCSS

Iniciar Sass

```
$ sass --watch archivo_entrada.scss:archivo_salida.css
```

Para comenzar a trabajar con Sass es necesario conocer un comando que compila Sass en CSS.

Pasos para usar sass --watch

1. Abrir el terminal
2. Buscar la carpeta del proyecto
3. Escribir el comando **sass --watch archivo_sass.scc:archivo_css.css**
4. Testear el compilado del archivo CSS

Estructurar un proyecto con Sass

¿Por qué estructurar proyectos con Sass?

- Orden
- Mantenible
- Separación lógica visual
- Sass posee características especiales para organizar proyectos

¿Qué características de Sass se pueden utilizar?

ARCHIVOS PARCIALES

- Archivos que contienen pequeñas partes de código CSS
- Es cualquier archivo con un guión bajo (_)
- Se deben importar a otro archivo para poder generarlo en CSS

¿Qué características de Sass se pueden utilizar?

MANIFIESTOS

- Archivo que contiene un grupo de archivos importados que juntos forman una unidad
- Administra todos los archivos parciales importados y los ensambla en una sola hoja de estilo
- Los parciales son importados al manifiesto utilizando la propiedad @import

¿Qué características de Sass se pueden utilizar?

IMPORT

- Extensión de la regla @import la cual puede importar archivos SCSS y Sass
- Todos los archivos que son importados con @import soin combinados en un único archivo CSS
- Esta directiva es crucial cuando utilizamos archivos de manifiesto

¿Qué características de Sass se pueden utilizar?

COMENTARIOS

- Soporta dos tipos de comentarios de múltiples líneas (**/* */**) y de línea simple (**//**)
- Los comentarios de línea simple solo se verán en Sass
- Utilizar comentarios es una buena implementación para documentar, definir, ordenar o declarar el código en un archivo Sass

Estructura de directorio de Sass

Existe una gran cantidad de métodos y técnicas para organizar un proyecto Sass los cuales son útiles dependiendo del proyecto a trabajar.

Nosotros utilizaremos el patrón 7-1

https://sass-guidelin.es/#the-7-1-pattern

¿Cómo implementar el patrón 7-1?

```
sass/
    base/
    components/
    layout/
    pages/
    themes/
    abstracts/
    vendors/
    main.scss
```

Para implementar el patrón 7-1 debemos crear 7 directorios dentro de una carpeta raíz.

En la raíz irá el manifiesto y en el interior de los directorios los parciales del proyecto Sass.

¿Qué son las clases semánticas?

- Las clases semánticas hacen referencia al acto de escribir clases que describen objetivamente el contenido y no a lo que pensemos que es el elemento.
- Crear clases semánticas es clave para el correcto uso de estos selectores en CSS y JavaScript.
- Escribir clases, de buena manera, hará que nuestros código sea escalable y fácil de entender por otros desarrolladores.

¿Qué son las metodologías de la arquitectura CSS?

Son unas guías de estilo que nos ayudarán a organizar y estructurar nuestro CSS de manera que sea fácil escalarlos y mantenerlos por otros desarrolladores.

Las metodologías que podremos usar para esta tarea son:

- OOCSS
- SMACSS
- Atomic Design
- SUITCSS
- BEM

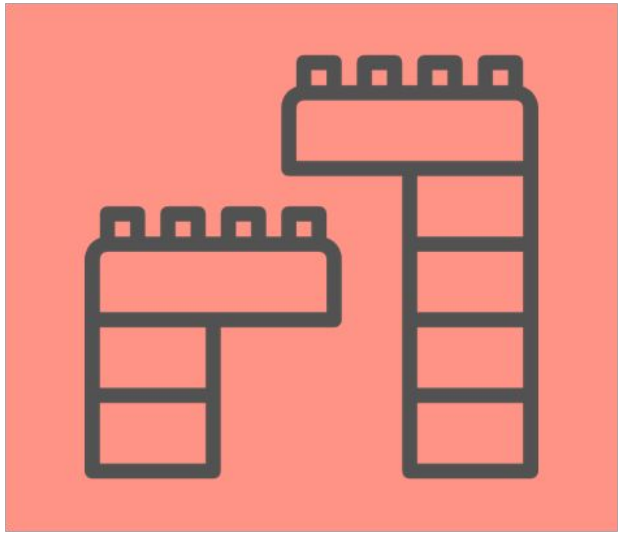
entre otras.

BEM

¿Qué es BEM?

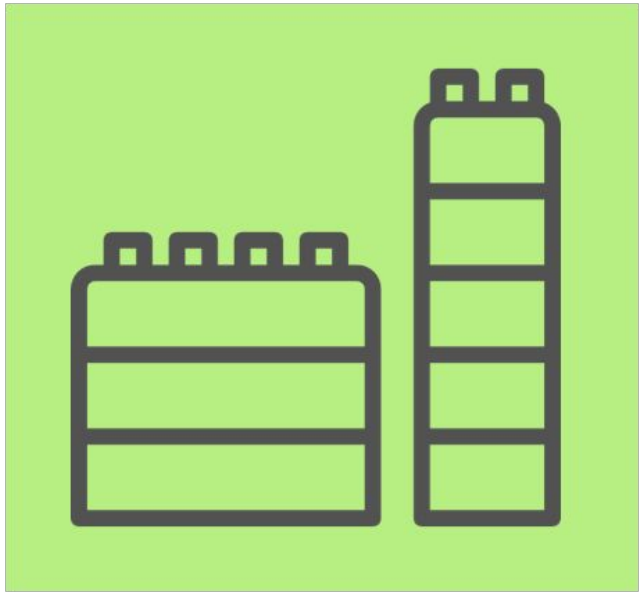
Bem significa bloque, elemento y modificador y como sabemos es una metodología que nos ayudará a separar los elementos de nuestra interfaz en bloques.

¿Por qué elegir BEM?



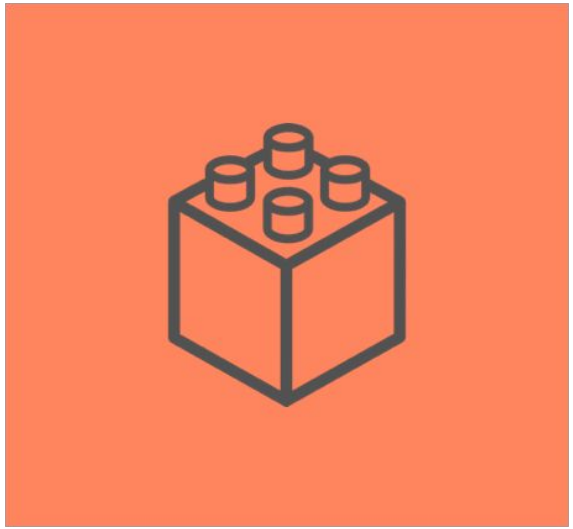
- La nomenclatura usada para escribir clases evita la repetición de estilos previniendo conflictos asociados a cascada CSS.
- Tiene gran aceptación y popularidad por parte de los desarrolladores.
- Aprender las bases de esta metodología no conlleva gran dificultad.

BEM: Bloques



Los bloques son entidades o funcionalidades independientes de un componente. Estos en HTML son representados con el atributo de clase.

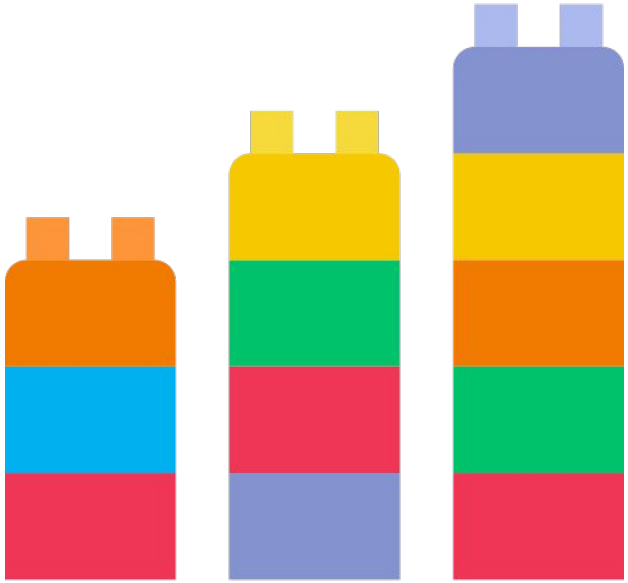
BEM: Elementos



- Los elementos son entidades que están ligadas a un bloque y que usadas por sí sola no tendrán una connotación semántica.

- Para identificarlos como elementos de un bloque deberemos nombrar a la clase usando doble guión bajo (__).

BEM: Modificaciones



Los modificadores son entidades que definen la apariencia, estado o comportamiento de un bloque o elemento.

Utilizar Sass en proyectos web

Dentro de los archivos SCSS se pueden utilizar muchas reglas y funcionalidades que facilitarán el trabajo a la hora de organizar y reutilizar códigos CSS.

Dentro de ellas se encuentran:

- Variables
- Nesting
- Mixin

Variables

```
// Colors
$white: #fff;
$primary-color: $white;

// Fonts
$serif: “Lora”, “Playfair Display”, Georgia, serif;
$primary-font: $serif;

// Spacing
$min-space: 10px;
```

- Ayudan a almacenar la información que se desea reutilizar en las hojas de estilo
- En ellas se pueden almacenar colores, fuentes o cualquier valor de CSS
- Éstas deben comenzar con el signo $

Nesting

```
.hero-title {
     display: flex;
     flex-direction: column;
     color: $primary-color;
     text-align: center;
     font-family: $primary-font-family;
     span {
          font-size: $font-size-large;
          font-family: $secondary-font;
          font-weight: $bold;
          }
```

- Permiten anidar los electores CSS de manera que sigan una jerarquía visual similar a HTML
- Ayudan a prevenir conflictos de especificidad entre reglas CSS evitando que se cancelen entre sí
- Es importante NO anidar excesivamente ya que el proyecto se vuelve inmantenible

Ampersand (&)

```
.button_positive {
     background-color: $verdigris;
     border-color: $verdigris;

     &:hover {
          background-color: $medium-aquamarine;
          border-color: $medium-aquamarine;
     }
}
```

- La función ampersand nos ayudará a crear clases más específicas usando nesting.
- Al lado del ampersand podrán ir clases, pseudo clases, pseudo elementos, entre otros.

Mixin

- Ayudan a crear un grupo de declaraciones CSS que se pueden reutilizar dentro de las hojas de estilo.
- Dentro de ellas, se pueden pasar argumentos para hacer más flexible su uso.

```
@mixin background($color, $image, $position, $size, $repeat) {
     background-color: $color;
     background-image: $image;
     background-position: $position;
     background-size: $size;
     background-repeat: $repeat;
     }
.hero-container {
     @include(#e94f37, url(“../images/background.png”), 80%, cover, no-repeat);
}
```