sustantiva

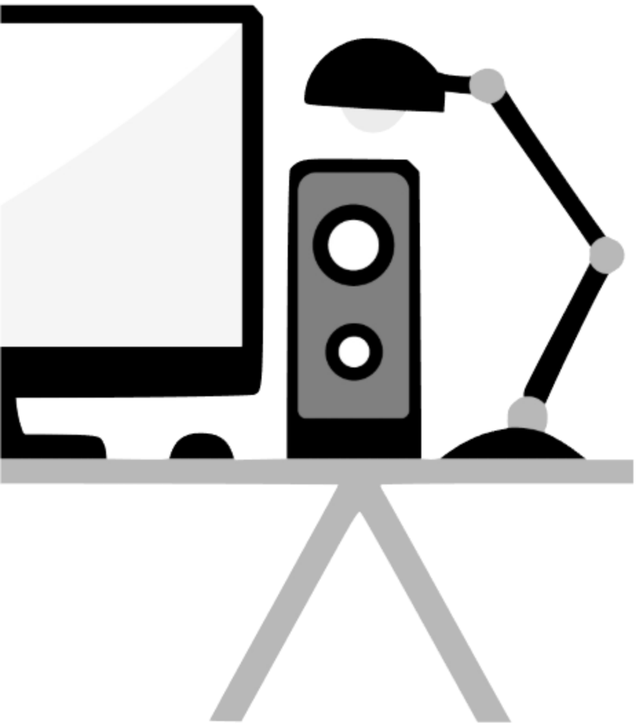
CON•SENTIDO

DESARROLLO DE LA INTERFAZ DE USUARIO WEB

Lección 01

sustantiva
CON•SENTIDO

Contenidos



1. Del diseño a la implementación de un producto digital
2. El proceso de desarrollo de un producto visual
3. El proceso de ideación de un producto digital
4. El proceso de implementación de un producto digital
5. El rol del diseñador UX/UI
6. El rol del desarrollador Front-End
7. Importancia de las guías de estilos y representaciones visuales en la maquetación
8. Metodologías para la organización y modularización de estilos
9. Ventajas de utilizar una metodología
10. Metodologías más utilizadas (BEM, OOCSS, SMACCS)
11. Preprocesadores CSS
12. Qué es un procesador CSS
13. Importancia del preprocesador en el desarrollo front
14. Preprocesadores más utilizados (SASS, LESS)

Diseño e implementación de un producto digital

¿Qué es el desarrollo de producto digital?

El desarrollo de producto digital se refiere al diseño de experiencias de usuario basadas en software, que mejoran el viaje del usuario dentro de una organización de forma parcial o total. Por lo general, emplea metodologías de desarrollo agile para agilizar la entrega de productos mientras se prueban y se rehacen constantemente en base al feedback de los usuarios.

Diseño e implementación de un producto digital

¿Cuáles son los beneficios del desarrollo de producto digital?

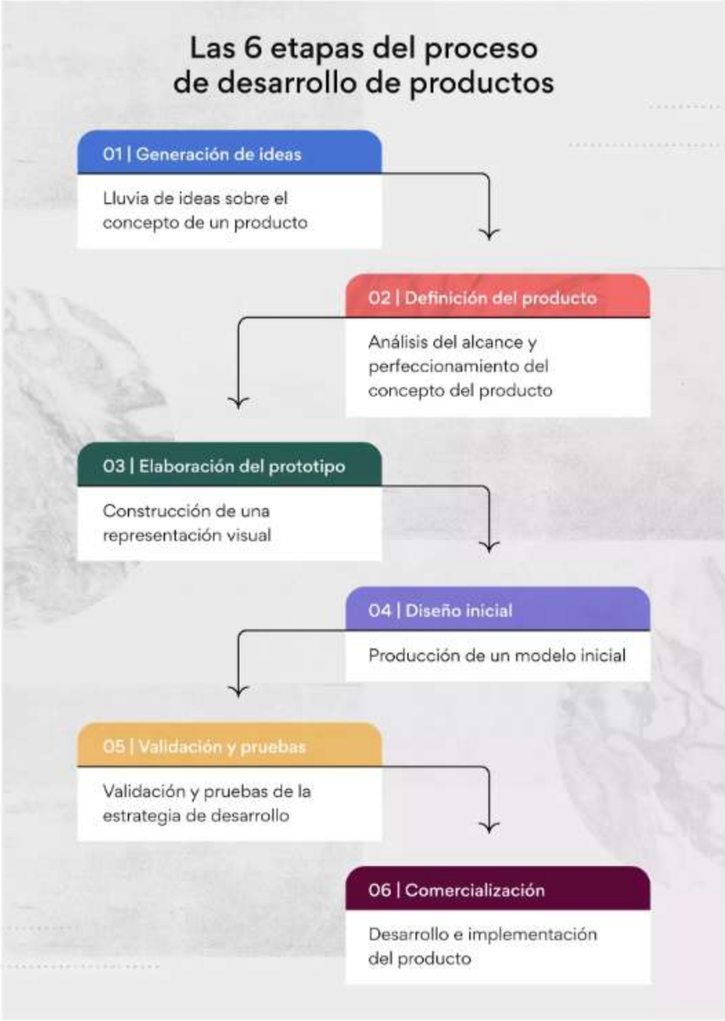
El principal es la entrega de un producto digital que utiliza la tecnología más innovadora y apropiada para maximizar la experiencia de usuario. Un segundo beneficio es la escalabilidad. A medida que las necesidades del usuario y la disponibilidad del software evolucionan, también lo hacen los productos digitales, que se mejoran continuamente para dar respuesta a los cambios en la expectativas de los usuarios.

Desarrollo de un producto visual y digital

El desarrollo de un producto nuevo es algo que entusiasma pero que también presenta un gran desafío. Desde la ideación, pasando por las investigaciones y la elaboración del prototipo, ningún lanzamiento es exactamente igual a otro. Sin embargo, hay un sistema general que puede ayudarte a iniciar el proceso de desarrollo de un producto.

El proceso de desarrollo de productos cuenta con seis pasos necesarios para llevar a un producto desde el concepto inicial al lanzamiento al mercado. Se incluyen la identificación de las necesidades del mercado, la investigación sobre la competencia, la ideación de una solución, el desarrollo de una hoja de ruta del producto y la elaboración de un producto mínimo viable (MVP).

Desarrollo de un producto visual y digital



Durante la fase de diseño inicial, los participantes trabajan juntos para producir un modelo (*mockup*) del producto basado en el prototipo MVP.

El diseño se debería crear con una audiencia objetivo en mente y habría que complementarlo con las funciones clave del producto nuevo.

Un buen diseño del producto puede requerir de varias iteraciones hasta que quede bien. Además, también puede hacer falta comunicarse con los distribuidores para conseguir los materiales necesarios para el diseño.

Implementación de un producto digital

Tras las distintas fases para desarrollar un producto digital, el siguiente paso consiste en implementarlo. Para esto, es necesario que dos personas clave trabajen e interactúen entre ellos: El diseñador de UI/UX y el desarrollador Frontend.

Es necesario que estos dos roles trabajen juntos y se comuniquen, ya que lo diseñado por la persona a cargo del UI/UX, debe ser convertido a Código por el desarrollador frontend.

El rol del diseñador UX/UI

Un diseñador de interfaz de usuario (UI) **se enfoca en esquematizar cada una de las pantallas o páginas con las cuales debe interactuar un usuario**. Su principal trabajo consiste en definir y organizar los elementos que el usuario debe emplear para completar una tarea.

Por su parte, **un diseñador UI o User Interface es el especialista en 'dar vida' al trabajo del UX Designer**. El diseño UI se encarga de implementar esos wireframes o prototipos de webs o aplicaciones, de una forma visual.

El rol d UX/UI

Un diseñador de interfaz de usuario (UI) **se enfoca en esquematizar cada una de las pantallas o páginas con las cuales debe interactuar un usuario**. Su principal trabajo consiste en definir y organizar los elementos que el usuario debe emplear para completar una tarea.

Por su parte, **un diseñador UI o User Interface es el especialista en 'dar vida' al trabajo del UX Designer**. El diseño UI se encarga de implementar esos wireframes o prototipos de webs o aplicaciones, de una forma visual.

El rol del Frontend Developer

Un desarrollador front-end **trabaja la interfaz de usuario desde el punto de vista del código, para que la interacción con el sistema sea posible**. Por lo tanto, se encarga de la parte visual de la web (de todo aquello que puedes ver en tu explorador) haciendo que su diseño sea intuitivo y atractivo.

Pero, ojo, porque, aunque lo parezca, el **front-end no es un diseñador**, sino que **recoge los documentos y directrices del equipo de diseño para trasladarlas a código** hace su magia convirtiendo esas ideas en realidad. Por lo tanto, que el desarrollador conozca lo básico del buen diseño gráfico e interactivo será un plus para su desempeño.

Importancia de las guías de estilos

Las guías de estilo son muy importantes a la hora de desarrollar, puesto que poseen una serie de reglas orientadas a hacer el diseño más consistente.

Grandes compañías han hecho público y documentado sus guías de estilo, por lo que es posible acceder a estas y aprender como lo crearon.

Importancia de las guías de estilos

Qué es una Guía de estilo (UI)

Es un documento que define criterios visuales y de maquetación para asegurar consistencia en una interfaz: **paleta de colores, tipografías, grid/espaciados, iconografía, componentes UI reutilizables (botones, tarjetas, formularios), estados y accesibilidad**. Se usa como referencia al construir nuevas pantallas o secciones.

Fuente: Google. (s. f.). *Material Design 3*. https://m3.material.io/

Importancia de las guías de estilos

Conceptualmente, el interfaz de usuario descansa en 3 puntos:

- **Significado (qué)**: es la base del interfaz. Recoge el contenido o información de la pantalla. Textos, campos de formularios, botones, menús…

- **Comportamiento**: trata el funcionamiento del interfaz. Cómo se comporta cuando un usuario envía un formulario (validaciones), hace clic en un enlace...

- **Aspecto**: apariencia final de un sistema: colores, tipografía, disposición de los elementos en pantalla (layout).

Las Guías de Estilo, generalmente se centran en el "Aspecto". Puntos como diseño y maquetación (colores, tipografías y píxeles), y apenas incluye criterios o casuística para aplicar en el proceso de diseño de interfaz ("Significado").

sustantiva
CON•SENTIDO

Metodologías para la organización y modularización

Antes de empezar aclarar que cuando hablamos de metodología nos referimos a un sistema de métodos. Donde un método es simplemente una forma de hacer algo de una manera sistemática, un cierto modo preestablecido de hacer las cosas para lograr el resultado que queremos.

Se hace difícil de manejar el css en sistemas grandes y complejos. A medida que la hoja de estilo crece nos damos cuenta que no paramos de repetir reglas para diferentes elementos y cualquier cambio puede afectar algún elemento no deseado.

sustantiva
CON•SENTIDO

Ventajas

- **Añade especificidad:** Usa un selector único para cada regla, lo que permite reducirla y hacer menos repeticiones.
- **Aumenta la independencia:** Los bloques se pueden mover a cualquier parte del documento, sin afectar los estilos.
- **Mejora la herencia múltiple:** Se puede cambiar cualquiera de las tres partidas sin afectar a las demás.
- **Permite la reutilización:** Es posible reciclar ciertas áreas de código desde un proyecto hacia otro, esto debido a la no existencia de dependencias mayores en cuanto a la implementación de cada uno de los bloques estructurados.
- **Entrega simplicidad:** Permite un fácil entendimiento una vez conocido el proceso lógico sobre el cual se basa. A su vez, las convenciones a la hora de nombrar las clases permiten tener un control absoluto al saber a qué, quién y hacia dónde hacemos referencia dentro de una estructura.

Metodologías más utilizadas

Se pueden destacar al menos 3 metodologías entre las más utilizadas, estas son BEM, SMACSS OOCSS

BEM.css

```css
.menu {
  background: #0B2027;
}

.menu__trigger {
  background: #16414f;
  float: left;
  padding: 1.3rem 0;
  width: 10%;
  text-align: center;
  color: white;
  font-size: 1.5rem;
  -webkit-transition: .3s;
          transition: .3s;
}
@media (min-width: 30em) and (max-width: 40em) {
  .menu__trigger {
    width: 15%;
  }
}
@media (max-width: 30em) {
  .menu__trigger {
    width: 15%;
```

BEM

OOCSS.css

```css
.sidebar {
    padding: 2px;
    left: 0;
    margin: 3px;
    position: absolute;
    width: 140px;
}

.list {
    margin: 3px;
}

.list-header {
    font-size: 16px;
    color: red
}

.list-body {
    font-size: 12px;
    color: #FFF;
    background-color: red;
}
```

OOCSS

SMACSS.css

```css
#article {
    width: 80%;
    float: left;
}

#sidebar {
    width: 20%;
    float: right;
}

.l-fixed #article {
    width: 600px;
}

.l-fixed #sidebar {
    width: 200px;
}
```

SMACSS

sustantiva
CON•SENTIDO

Metodologías más utilizadas

Blocks, Elements and Modifiers (BEM)

BEM es una abreviatura de los elementos clave de la metodología: Bloque, Elemento y Modificador. BEM significa *Modificador de Bloques de Elementos (Block Element Modifier)* por sus siglas en inglés. Sugiere una manera estructurada de nombrar nuestras clases, basado en las propiedades del elemento en cuestión. Cuando utilizamos la metodología BEM, hay que tomar en cuenta que solamente podemos usar nombres de clases (no IDs). Los nombres de clases permiten repetir el nombre BEM si es necesario, y crear una estructura de código más consistente (en ambos archivos el HTML y CSS/Saas).

Metodologías más utilizadas

sustantiva
CON•SENTIDO

Bloque: Entidad independiente que es significativa por sí sola. *(header, container, menu, checkbox, input,* button*...)*

```
.button {
        display: inline-block;
        border-radius: 3px;
        padding: 7px 12px;
        border: 1px solid #D5D5D5;
        background-image: linear-gradient(#EEE, #DDD);
        font: 700 13px/18px Helvetica, arial;
}
```

Elemento: Una parte de un bloque que no tiene un significado independiente y está semánticamente vinculado a su bloque. (menu item, list item, checkbox caption, header title, state success...)

```
.button--state-success {
        color: #FFF;
        background: #569E3D linear-gradient(#79D858, #569E3D) repeat-
x;
        border-color: #4A993E;
}
```

Modificador: Una bandera en un bloque o elemento. Úsalos para cambiar la apariencia o el comportamiento. *(disabled, highlighted, checked, fixed, size big, color yellow,* state danger*...)*

```
.button--state-danger {
        color: #900;
}
```

En el ejemplo vemos que podemos tener un botón normal para los casos habituales y dos estados más para los diferentes.

Las reglas de nomenclatura nos dicen que usemos la sintaxis block--modifier-value

sustantiva
CON•SENTIDO

Metodologías más utilizadas

Scalable and Modular Architecture for CSS (SMACSS).
En el núcleo de SMACSS (Arquitectura en CSS Escalable y Modular) está la categorización. Al clasificar las reglas CSS, comenzamos a ver patrones y podemos definir mejores prácticas en torno a cada uno de estos patrones.
Cada categoría tiene ciertas pautas que se aplican a ella. Esta separación algo sucinta nos permite hacernos preguntas durante el proceso de desarrollo. ¿Cómo vamos a codificar las cosas y *por* qué las vamos a codificar de esa manera?
Gran parte del propósito de categorizar cosas es codificar patrones, cosas que se repiten dentro de nuestro diseño. La repetición resulta en menos código, mantenimiento más fácil y mayor consistencia en la experiencia del usuario. Estas son todas las victorias. Las excepciones a la regla pueden ser ventajosas, pero deben justificarse.

Metodologías más utilizadas

Al separar las reglas en las cinco categorías, la convención de nomenclatura es beneficiosa para comprender de inmediato a qué categoría pertenece un estilo en particular y su función dentro del alcance general de la página.

En proyectos grandes, es más probable que los estilos se dividan en varios archivos. En estos casos, la convención de nomenclatura también facilita encontrar a que archivo pertenece un estilo.

```
html, body, form { margin: 0; padding: 0; } input[type=text] { border:
1px solid #999; } a { color: #039; } a:hover { color: #03C; }
```

sustantiva
CON•SENTIDO

Metodologías más utilizadas

CSS orientado a objetos (OOCSS) nos permite separar el contenedor y el contenido con “objetos” CSS

Al igual que con cualquier método de codificación basado en objetos, el objetivo de OOCSS es fomentar la reutilización del código y, en última instancia, hojas de estilo más rápidas y eficientes que son más fáciles de agregar y mantener.

Se basa en dos principios:

Estructura y piel separadas

Contenedor separado y contenido

sustantiva
CON•SENTIDO

Metodologías más utilizadas

```
.button {
  width: 200px;
  height: 50px;
}

.box {
  width: 400px;
  overflow: hidden;
}

.widget {
  width: 500px;
  min-height: 200px;
  overflow: auto;
}

.skin {
  border: solid 1px #ccc;
  background: linear-gradient(#ccc, #222);
  box-shadow: rgba(0, 0, 0, .5) 2px 2px 5px;
}
```

Todos los elementos están usando clases, los estilos comunes se combinan en una “máscara” reutilizable y nada se repite innecesariamente. Solo tenemos que aplicar la clase “skin” a todos los elementos y el resultado será el mismo que el que produciría el primer ejemplo, excepto con **menos código y una posibilidad de reutilización adicional** .

sustantiva
CON•SENTIDO

Preprocesadores CSS

Preprocesador CSS

Un preprocesador CSS es un programa que te permite generar CSS a partir de la syntax única del preprocesador. Existen varios preprocesadores CSS de los cuales escoger, sin embargo la mayoría de preprocesadores CSS añadirán algunas características que no existen en CSS puro, como variable, mixins, selectores anidados, entre otros. Estas características hacen la estructura de CSS más legible y fácil de mantener.

Para utilizar un preprocesador CSS debes instalar un compilador CSS en tu web server.

sustantiva
CON•SENTIDO

Importancia preprocesadores en frontend

Ventajas de utilizar Preprocesadores

- Reduce el tiempo para crear y mantener el CSS.
- Permite tener una organización modular de los estilos, lo cual es vital para proyectos grandes.
- Proporciona estructuras avanzadas propias de los lenguajes de programación, como variables, listas, funciones y estructuras de control.
- Permite generar distintos tipos de salida, comprimida, normal o minimizada, trabajando tanto en desarrollo como en producción, además se hace una forma muy fácil.
- Permite vigilar los ficheros, de tal manera que, si ha habido un cambio en la hoja de estilos, se regenera automáticamente (modo watch).

sustantiva
CON•SENTIDO

Preprocesadores más utilizados

Sass y LESS son dos populares **Preprocesadores CSS**. Una de estas herramientas es un superconjunto de secuencias de comandos de CSS que hace que escribir código CSS sea más cómodo.

Ambos preprocesadores CSS: **SASS vs LESS** son expansiones de plantilla únicas que hacen que la planificación y el desarrollo sean más simples y efectivos.

sustantiva
CON•SENTIDO

Preprocesadores más utilizados

Ventajas de Sass frente a Less

- **Es mejor para CSS3**, sobre todo si la asociamos con herramientas como Compass o Bourbon, herramientas que permiten desarrollar CSS3 de manera mucho más rápida y con muchas librerías ya estandarizadas.

- **Tiene mejores estructuras de control**, como las estructuras de control condicionales como if/else, bucles como for y while, o la estructura de control para recorrer mapas each.

- De manera nativa, **permite minimizar la salida de ficheros CSS**, algo que Less no permite.

- **Es más usado**, si atendemos a los repositorios que están en GitHub, que es un estándar a la hora de medir el uso de tecnologías, e incluso hay estudios de Developers Survey que así lo indican.

sustantiva
CON•SENTIDO

Preprocesadores más utilizados

Desventajas de Sass frente a Less

- Al tener más estructuras de control, **Sass tiene una sintaxis más compleja**, por lo que es un poco más complejo de aprender que Less.

- En cuanto a la documentación, y esto es un aspecto un poco subjetivo, **la de Less está mejor que la de Sass**, que tiene todo bien documentado, pero a veces es difícil encontrar la explicación.

sustantiva

CON•SENTIDO