

15/02/2022

## Misión-FrontEnd

### Introducción a la programación FrontEnd

## ¿Qué es la programación FRONTEND?

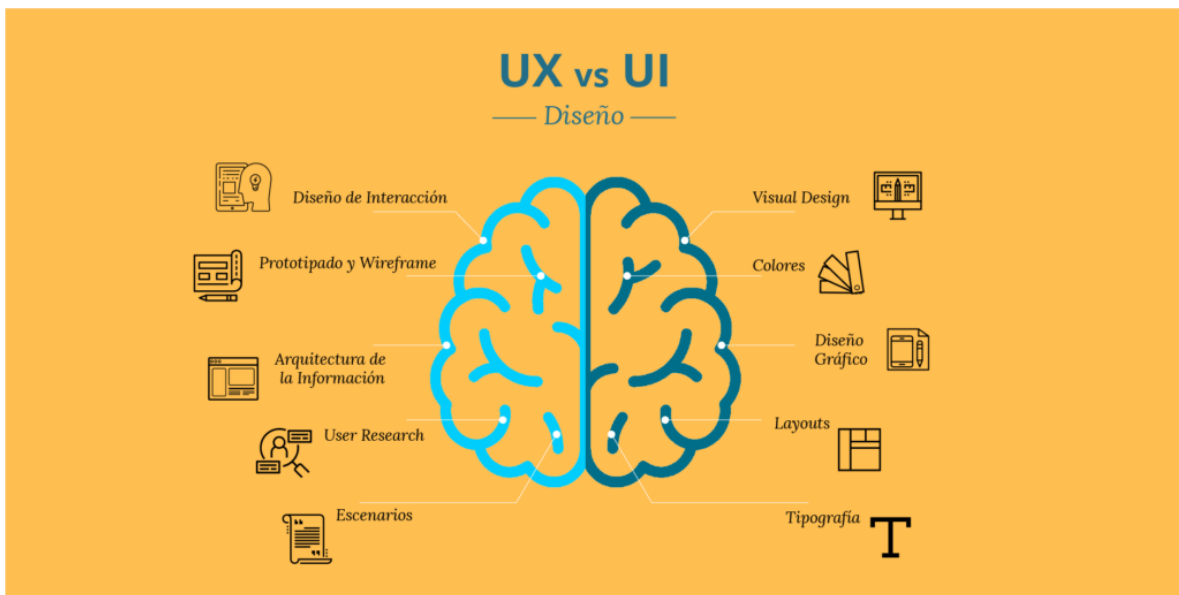
### ¿Qué significa Front?

Cuando hablamos de front, hablamos de la parte de enfrente de la página web, ósea la parte con la que nosotros como usuarios interactuamos, esta parte es la que manda toda la información a la parte backend. Esta parte se encarga del diseño de la pagina web; desde la estructura, el acomodo, la distribución de contenido, los estilos y los flujos de interacción. En pocas palabras de todo el diseño de la página web.

### Usuario

Debemos de siempre estar consientes de todo lo que hacemos será usado por algún usuario que use la pagina, y en esta persona es en la que nos debemos de enfocar.

Existe el UI/UX, los cuales se refieren a User Interface y User Experience, estas partes son las mas importantes del diseño web, y siempre se deben de realizar antes de empezar a programar.

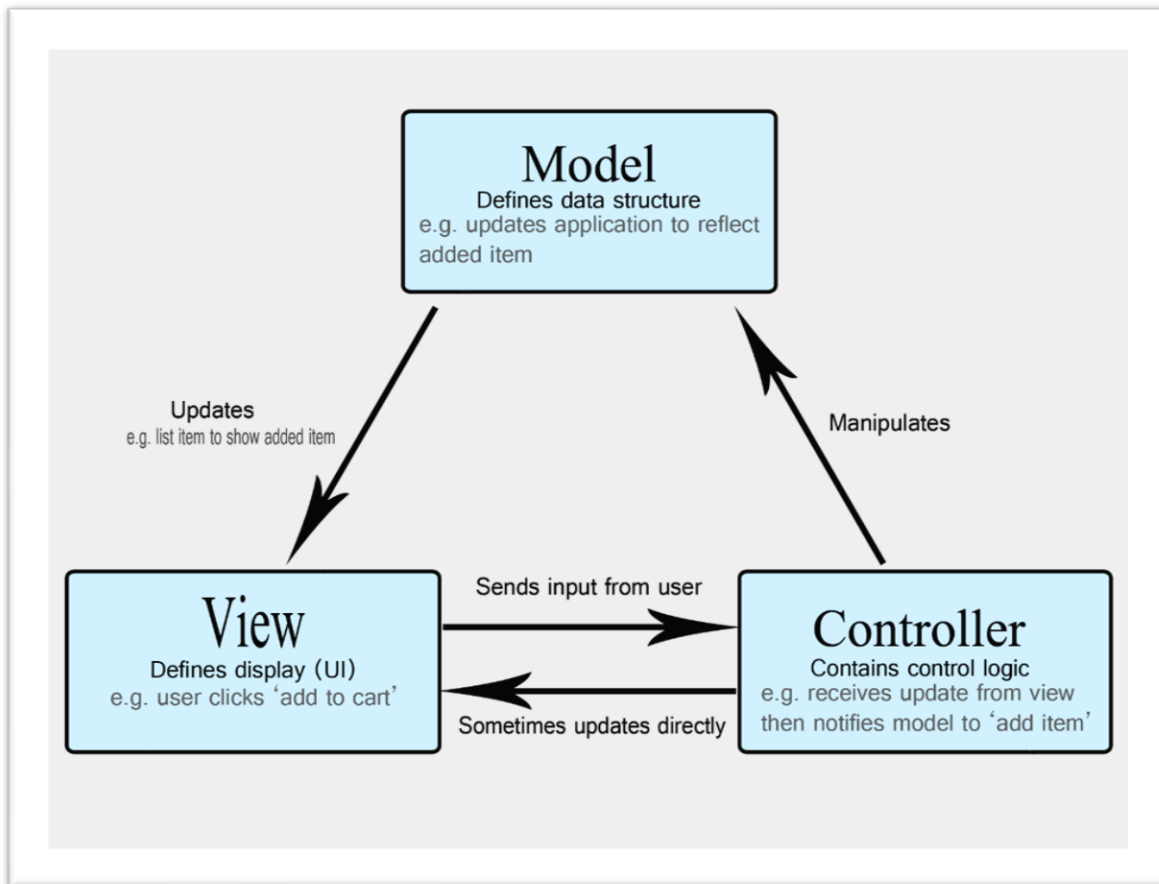


UX- viene de User experience (Experiencia de usuario). Es la experiencia que el usuario sentirá al usar la pagina web. Con ella nos enfocamos a dar la mejor experiencia posible al usuario, para tener mas oportunidades con los usuarios.

UI- viene de User Interface (Interface de usuario). Esta consiste en guiar al usuario por la página web o aplicación mientras la esté usando. Un buen diseño de UI nos permitirá guiar a los usuarios por la navegación y los llevará a tomar acciones naturalmente de manera intuitiva.

## Patrones de diseño en el desarrollo web

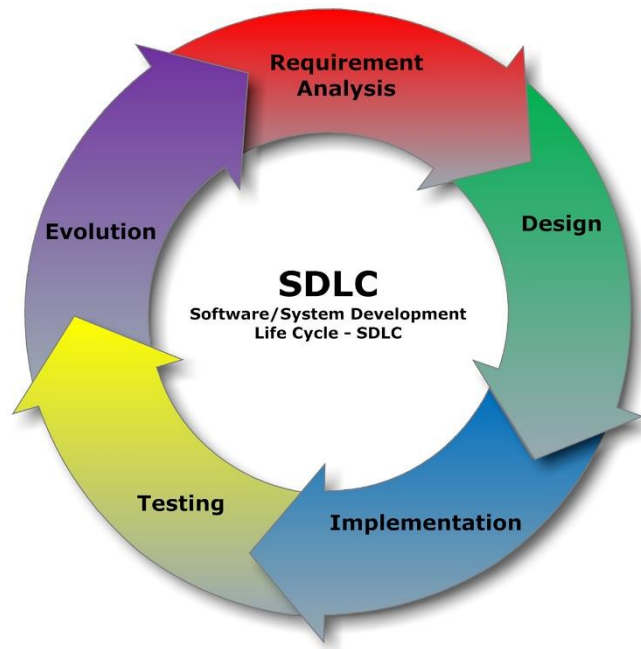
### MVC (Model View Controller)



MVC se refiere a la separación de la interfaz de usuario(Vista) de la parte de datos y negocio(Modelo), la cual esta intermediada por un controlador. El controlador se encarga de realizar validaciones, sanitizaciones de datos, manejo de errores, entre algunas otras mas cosas que hace que permiten que los datos que se mandan desde la vista, llegue de forma mas ordenada y limpia a su procesamiento en backEnd que es donde se encuentra el modelo.

### SDLC ( Software Development Life Cycle)

El desarrollo de software tiene un ciclo de vida que debemos tomar en cuenta siempre que vayamos a hacer una aplicación, así sea proyecto personal, un trabajo o cualquier momento en el que desarrollemos software.



En el SDLC existes diferentes fases que harán que nuestro desarrollo seas mucho mas controlado y sobre todo escalable y mantenible.

- Fase 1: Requerimientos --> Fase donde se presentan las necesidades de la aplicación.
- Fase 2: Diseño --> En esta fase los requerimientos se convierten en un plan y en lo que debería de parecer la aplicación o producto final.
- Fase 3: Desarrollo --> En esta fase se hace la programación de las aplicaciones, aquí es donde metemos el código con las mejores prácticas y con las reglas de las guías de desarrollo seguro.
- Fase 4: Verificación --> En esta fase revisarás y confirmarás que las buenas prácticas se aplicaron en el código. En esta parte se integran las pruebas de CI/CD e integración de pruebas unitarias.
- Fase 5: Mantenimiento y evaluación --> Los sistemas son un ente vivo y por lo tanto tiene que mantenerse en continuo movimiento.

## ¿Qué tecnologías se usan en FrontEnd?

En el FrontEnd existen cientos de formas de hacer que tengamos una página o aplicación web, y eso sin contar que también podríamos contar como FrontEnd las aplicaciones móviles y de escritorio en toda su secciones de interfaces.

En el caso de este curso utilizaremos principalmente 3 tecnologías y sus variantes, estas son: HTML, CSS y JavaScript.



### HTML

Este lenguaje sirve para hacer el esqueleto de nuestra aplicación, define la estructura del sitio y lo que nos da la pauta y el inicio de nuestra aplicación web.

### CSS

Este lenguaje nos sirve para hacer el diseño de toda la parte visual y estética de nuestra aplicación web.

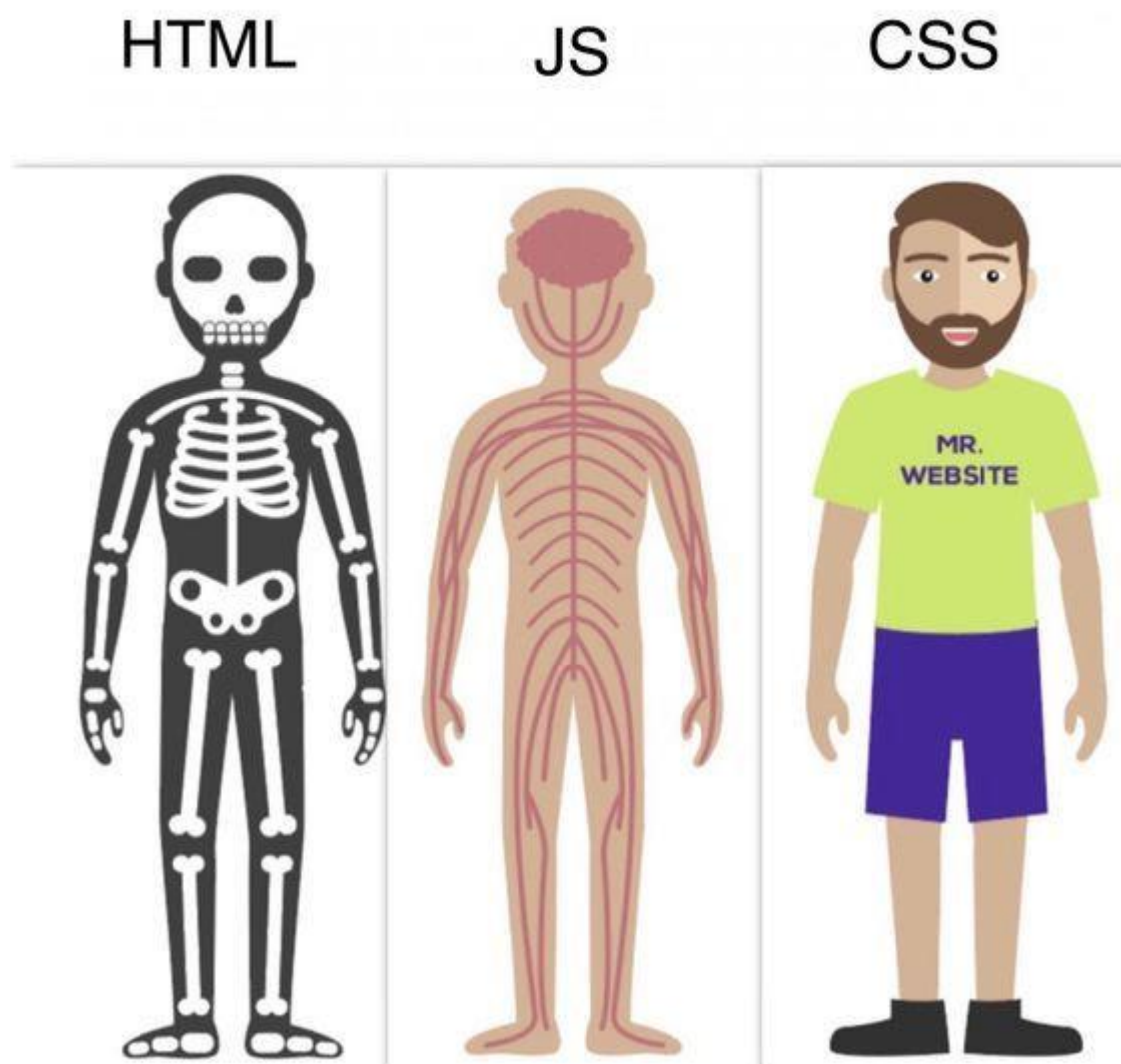
Se utilizan clases y selectores para poder definir las propiedades y características de cada uno de los elementos que definamos en la aplicación web.

### JavaScript

A JavaScript lo podemos llamar el cerebro de nuestra plataforma, ya que una vez que utilizamos JS en el sitio le damos la capacidad de escalar funcionalidades de forma exponencial, ya que pasamos

de las propiedades que tienen las etiquetas a tener una cantidad virtualmente infinita de posibilidades.

Podemos considerar a JS como el sistema nervioso que controla toda nuestra aplicación web y la que maneja todos los músculos y huesos de nuestro sitio.



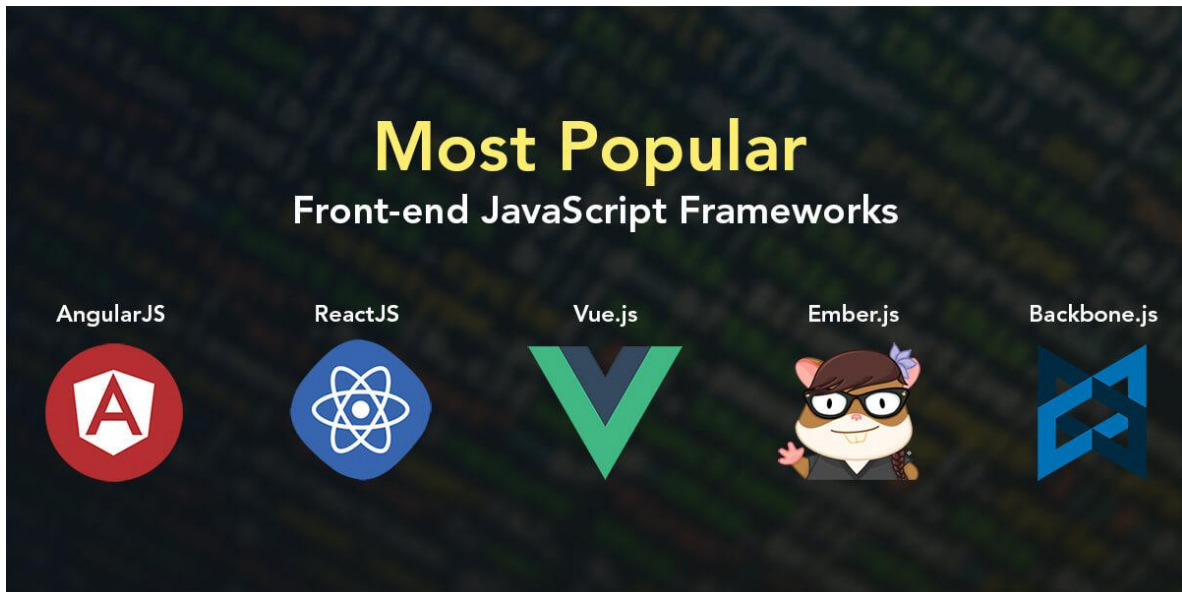
Esta imagen es una manera muy acertada de como funcionan estos lenguajes en nuestra aplicación web.

## Frameworks

Los frameworks son variantes del JavaScript que nos ayudan a que nuestra programación pueda llegar a ser mucho más rápida o con algunas funcionalidades adicionales a vanilla JS, esta parte de frameworks aplica tanto a FrontEnd como para BackEnd.

Aunque tengamos un framework que permita programar de forma más sencilla NO SIGNIFICA que podamos saltarnos las partes fundamentales de JS, esto porque al ser la base de muchos frameworks web, es bastante útil conocer como funciona desde el fondo.

Existen muchos frameworks y librerías muy famosas, pero por mencionar algunos de FrontEnd están React JS, Vue JS, Angular, Ember Js, Svelte, entre muchos otros.



## Setup de un programador FrontEnd

En el caso específico de programación web y específicamente FrontEnd lo que utilizaremos es lo siguiente:

- Herramienta de diseño de aplicación
- IDE de programación
- Navegador web
- Diferentes opciones de "Developer tools"
- Herramienta de documentación

Algunas de las herramientas podrían ser opcionales o en un ámbito profesional podrían ser utilizadas por otras áreas de la empresa, sin embargo, es importante que ustedes las conozcan para que, en el flujo operativo de la empresa, su aporte sea mucho más valioso.

## Herramienta de diseño

Esta herramienta es la que nos da la capacidad de tener un prototipo sin poner NI UNA línea de código.

En el flujo de programación, basándonos en el SDLC (Software Development Life Cycle) siempre es necesario empezar con el diseño de la solución antes de irnos directo a programarla, esto se hace así porque en caso de haber un cambio o modificación es mucho más fácil y rápido cambiarla en el lado de diseño que volver a programar el módulo completo.

El diseño tiene como objetivo poder aterrizar los requerimientos de quien nos esté pidiendo el software y para eso existen diferentes técnicas como por ejemplo: Design Thinking, Visual Thinking, diagramas de flujo, o la que mejor se te acomode para poder entender lo más cercano posible a lo que tu cliente esté pidiendo.

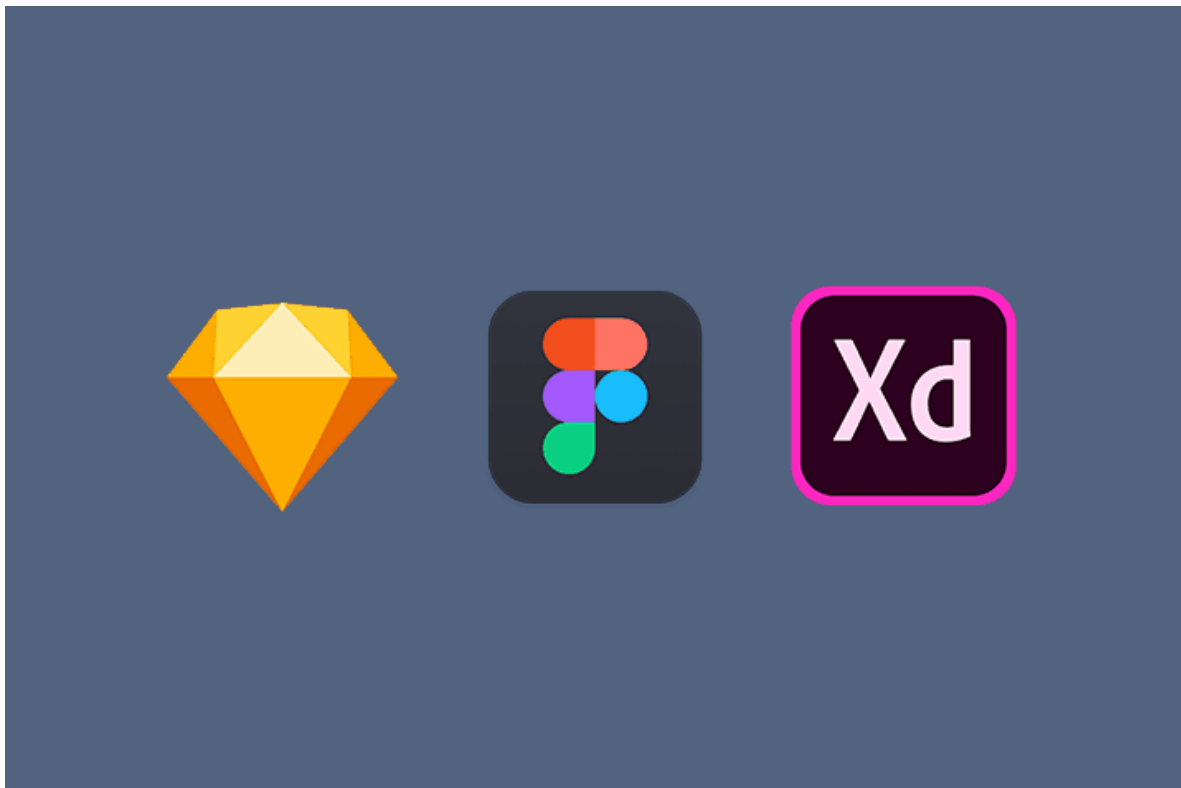
En cuanto a las opciones, existen muchas herramientas de diseño; desde Paint, Canva, Photoshop, Illustrator, Publisher o cualquier cuaderno que tengamos en casa, pero existen herramientas especializadas y enfocadas al diseño de wireframes e interfaces gráficas las cuales idealmente deberíamos usar para este caso.

Antes de adentrarnos en las herramientas recordemos que primero necesitamos poder aterrizar la idea y los requerimientos, para esto existe toda una fase previa a la estética de nuestra aplicación, esta fase es fundamental, ya que si no la realizamos podríamos encontrarnos obstáculos en el camino después. Esta es la fase de diseño de flujo y de WireFrame, lo que haremos en esta fase es entender las necesidades y empezar a dibujar como funciona la aplicación sin darle diseño gráfico aún, esta es una fase completamente funcional para darnos cuenta de las interacciones, los botones necesarios, el número de clicks y los diferentes movimientos de la aplicación y nos dará muchísima información para que nuestra aplicación sea lo más fluida posible.

Para esta fase de flujos y Wireframe existen diferentes herramientas como las siguientes:

- Miro (<https://miro.com/es/>) Esta herramienta te permitirá crear flujos de información y flujos de negocio.
- Balsamic mockups (<https://balsamiq.com/wireframes/>) Nos da elementos para poder crear interfaces rápidas y que nos dejan representar la idea.
- Dibujos a mano alzada --> a veces un wireframe también puede ser en una servilleta o en cualquier forma escrita.

Una vez realizado tu wireframe pasaremos a la parte de UI/UX, en este momento comenzaremos con toda la parte gráfica, de identidad y de uso de colores, formas e interacciones a lo largo de nuestra aplicación, para eso podemos usar cualquiera de las siguientes herramientas.



- Sketch (<https://www.sketch.com/>) Esta herramienta solo aplica si tienes una computadora MacOS, pero cuenta con una amplia gama de características enfocadas a diseñadores, lo que hace que se puedan crear cosas muy grandes y escalables.
- Figma (<https://www.figma.com/>) Esta herramienta es gratuita en muchas de sus características y permite colaboración entre usuarios, lo que es muy útil cuando se trabaja en equipo, tanto en equipos de diseño como en equipos de desarrollo.
- Adobe XD (<https://www.adobe.com/mx/products/xd.html>) Existe su versión de prueba gratuita que te dejará ocupar la herramienta completa para crear todos tus diseños y el costo para tener un plan pro no es muy alto, tiene la ventaja de estar basado en las



herramientas de Adobe, por lo que si ya tienes experiencia con alguna de estas herramientas, la curva de aprendizaje podría ser más corta.

Cualquiera de estas 3 últimas nos sirven también para la fase de wireframe y diseños de flujo, pero están más enfocadas hacia la parte de agregar diseño gráfico a los flujos realizado

## IDE de programación

Un IDE (Integrated Development Environment) es un software que nos ayuda a los desarrolladores a poder construir aplicaciones de forma más sencilla, esto es porque combina diferentes herramientas de desarrollo en una sola interfaz gráfica.

- Editor de código: Un simple editor de texto que te ayuda al escribir software pues tiene características como resaltar la sintaxis, encontrar algunos problemas dependiendo del lenguaje o autocompletando para que tu experiencia sea mejor.
- Automatización: Algunos IDEs tienen la posibilidad de automatizar tareas repetitivas como el despliegue de tu programa, la compilación o incluso la actualización de versiones.
- Depuración: Al tener toda la experiencia con lenguajes de programación, hay algunos IDEs que te ayudan a identificar errores o bugs antes de que se compile, e incluso hay algunos que te dan la solución o recomendaciones para que eso no pase.
- Integración: Existen herramientas que nos permiten integrar nuestro IDE de forma directa con su aplicación como lo son las interfaces, versionamientos de código o herramientas de despliegue.

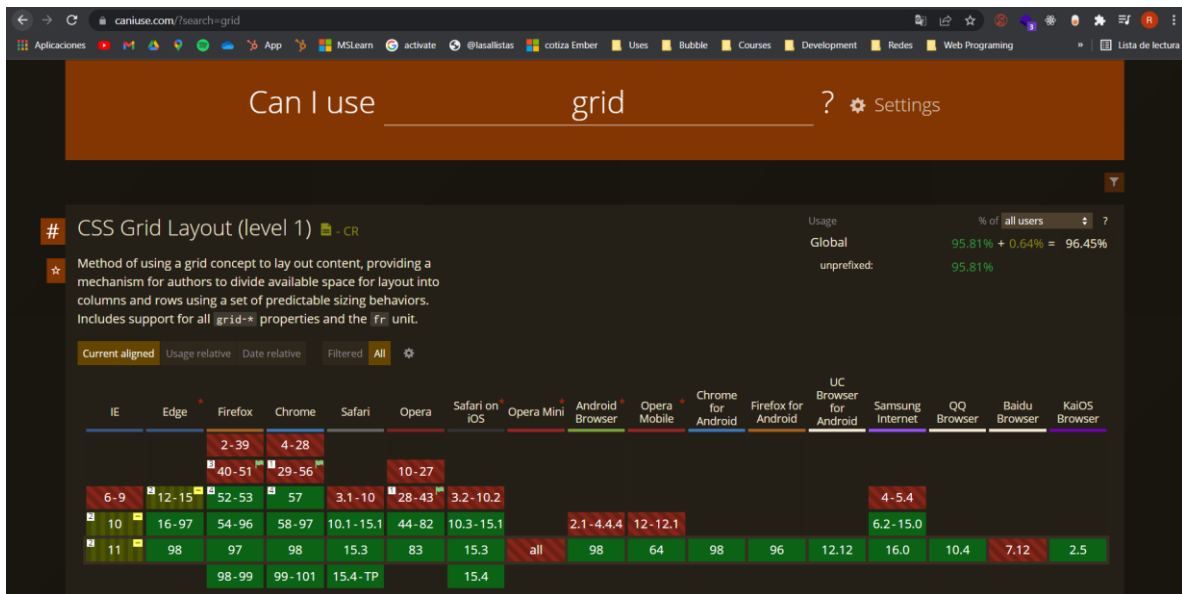
## Navegador Web

Al ser programadores web todo lo que programemos se desplegará en un navegador web, aquí hay que tener mucho cuidado y revisar que la tecnología que estemos utilizando es compatible y se despliega de forma igual o similar en todos los navegadores.

Recordemos que todo lo que hacemos es para nuestro usuario y si a nuestro usuario le gusta usar un navegador u otro, nuestra aplicación deberá de ser capaz de brindarle los servicios, independientemente del navegador en el que se encuentre.

Tenemos Navegadores desde Internet Explorer, Chrome, Brave, Safari, Firefox, Edge, Opera y muchísimos más que hacen diferentes interpretaciones al código que nosotros programemos, es por eso que siempre es importante revisar la compatibilidad.

Para eso existen plataformas como "Can I use" <https://caniuse.com/>, la cual nos dice en que navegadores podemos utilizar la tecnología de HTML, CSS, JS que queramos utilizar.



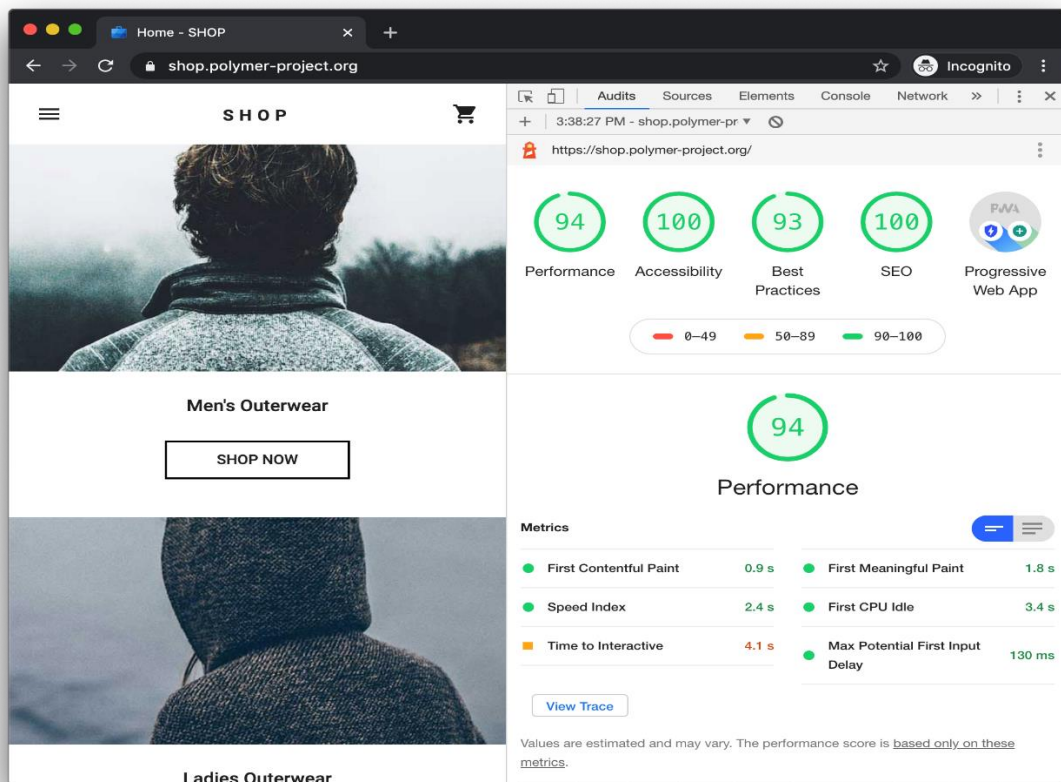
Antes de decidir que tecnología vamos a implementar es importante saber a que usuarios vamos dirigidos y poder conocer los navegadores más utilizados por ellos, tanto en ambientes de escritorio como móviles.

## Developer tools

Cuando hablamos de las Developer tools ya *no* nos referimos a todo lo que estamos viendo como parte del SetUp que necesitamos para programar y desarrollar nuestras aplicaciones.

En este caso nos referimos a herramientas adicionales que nos dan información sobre nuestra aplicación y mejoras que podemos implementar en nuestra aplicación.

Una de las más conocidas y usadas es Lighthouse de Google (<https://developers.google.com/web/tools/lighthouse?hl=es>) , la podemos encontrar directamente en el navegador y nos dará una auditoría del status de nuestro sitio en diferentes áreas como rendimiento, accesibilidad, buenas prácticas y SEO.

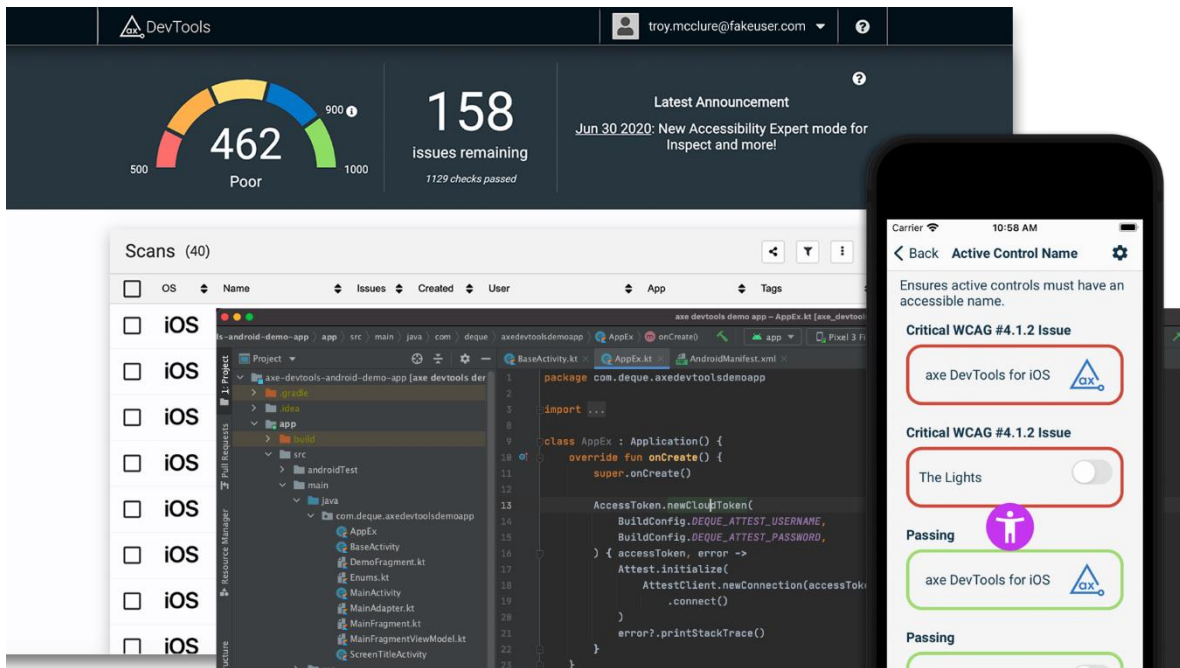


Si bien es de las más conocidas, no es la única y algo que debemos de tomar muy en cuenta es la parte de Accesibilidad en nuestros desarrollos, la democratización de la tecnología no solo significa que todos puedan tener acceso a ella, sino que también la puedas aprovechar independientemente de si tienes alguna condición física que no te permita interactuar con ella.

Pensemos en las personas que tienen debilidad visual o que son completamente ciegas usando nuestra aplicación web o personas que tienen algún problema motriz y que no pueden tocar todos los botones de nuestro sitio.

Estas personas también pueden ser nuestros usuarios y es nuestro deber como desarrolladores integrar las facilidades de accesibilidad a nuestras plataformas ayudando así a la democratización de la tecnología.

Para esto existen herramientas específicas que nos darán todas las guías y usos adecuados de accesibilidad para los usuarios como axe DevTools (<https://www.deque.com/axe/devtools/>) que puedes instalar como una extensión en tu navegador y te dará mucha información sobre cuáles son los problemas en tu desarrollo, pero lo más importante, información sobre cómo solucionarlos.



## Documentación

Una de las fallas más grandes que me ha tocado ver a lo largo de mi carrera es que no se documentan los desarrollos, esto es bastante grave, ya que en caso de que se cambie de desarrollador o de que sea un programa que tiene mucho tiempo que no mueves.

En el ámbito profesional tu proyecto no será considerado a menos que esté bien documentado y cuando eres parte de un equipo de desarrollo la documentación es la forma más sana de comunicarte con todo el equipo.

La falta de documentación hace que la mantenibilidad del código sea mucho más complicada y que muchas veces se tenga que reprogramar los módulos ya que en promedio si no usas el código en aproximadamente 6 meses, dejarás de comprender su funcionalidad.

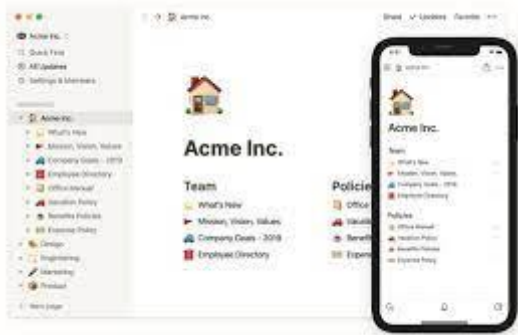
Para esto existen librerías como API Doc (<https://apidocjs.com/>) o como rails Admin ([https://github.com/railsadminteam/rails\\_admin](https://github.com/railsadminteam/rails_admin)) que nos dan facilidades para documentar mientras programamos.

Lo que es importante considerar al momento de documentar tu código es lo siguiente:

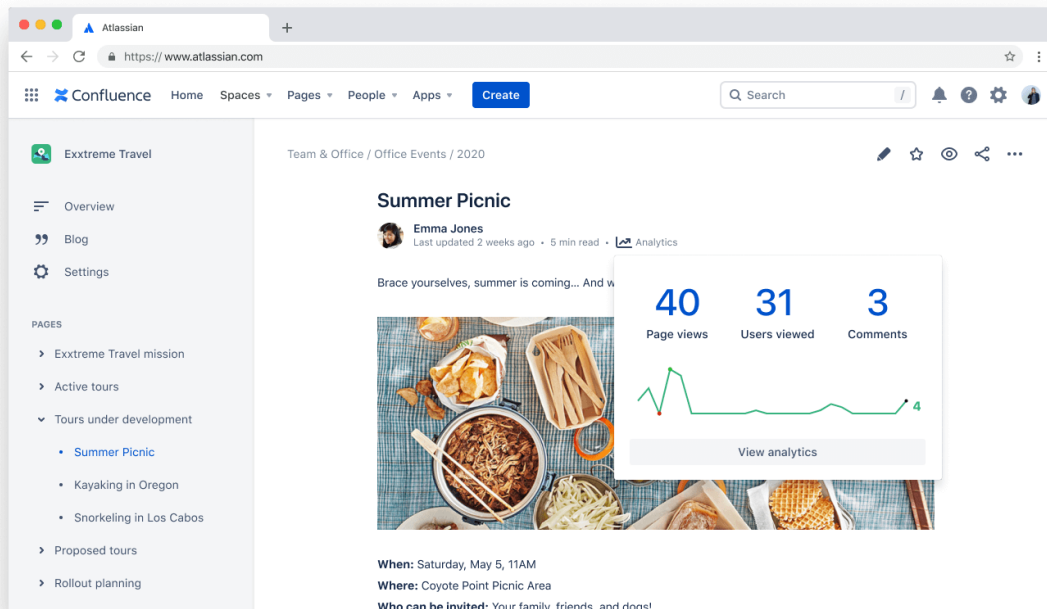
- Definir tu audiencia: A quien va a ir dirigida tu documentación, si es para otros desarrolladores, de que nivel o si es para alguien administrativo o si incluso es para ti mismo en un futuro.
- Qué problemática soluciona tu código: Todo lo que programamos tiene una funcionalidad por módulo y necesitamos definir que parte de la aplicación afecta, en que forma y cuales son las opciones de entrada y salida de nuestras funciones.

- Estructura de aplicación: Cuando documentamos todo un proyecto tendremos diferentes secciones, funcionalidades, páginas o incluso otros sitios que se conecten al nuestro, para esto es recomendable documentar desde lo más general a lo particular, un ejemplo muy bueno lo podemos encontrar en la documentación de Firebase (<https://firebase.google.com/docs/build>)

Una herramienta muy útil para poder lograr esta estructura es Notion (<https://www.notion.so/>)



Otra herramienta que igualmente es adaptable y está enfocada hacia desarrolladores por la parte de seguimiento de sprints por parte de Atlassian es Confluence (<https://www.atlassian.com/es/software/confluence>)



Estas son herramientas que te pueden servir a darle mucha más formalidad a tu proyecto y que hará que tengas más orden en todos tus proyectos y en lo profesional tu valor aumentará de forma muy relevante.

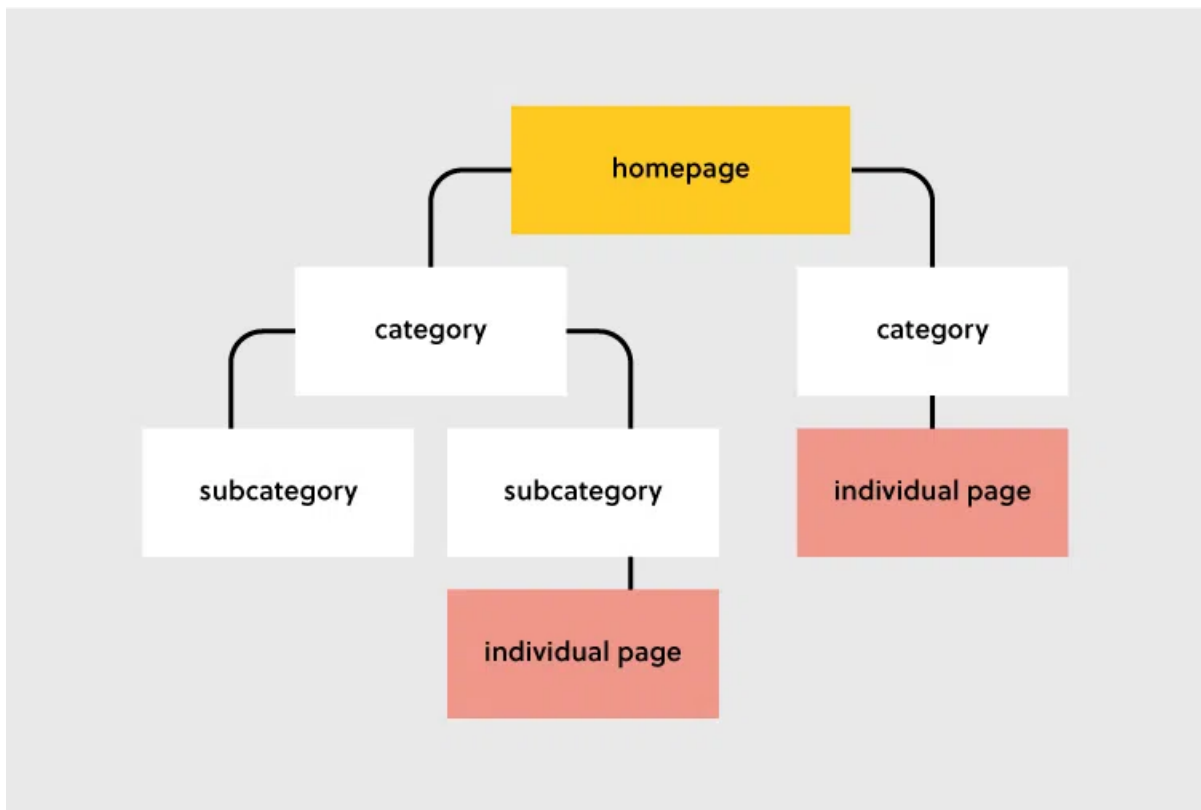
## Estructura Web

La estructura de tu sitio o aplicación web es el cómo están ligadas las diferentes páginas, visto desde como interactúan con los links y llamados internos, así como su jerarquía.

Esto es importante porque define la organización de la información, lo cual afecta tanto a tus usuarios, así como a los navegadores para poder leer bien tu contenido y que eso le ayude tanto a la navegación para las personas, pero también a darte una mejor calificación de SEO.

## Contenido Jerárquico

Esta es una de las estructuras web más utilizadas y conocidas por los diseñadores y desarrolladores, ya que utiliza una categorización de contenido de lo general a lo particular.



Esto se refiere a que tenemos una raíz y de ahí se desprende todo el contenido informativo y de uso en el sitio.

## Homepage

Esta es la página inicial y en donde mostrarás la información más importante, todas las páginas importantes de tu sitio deben de estar ligadas a esta para que la navegación sea mucho más sencilla e intuitiva para los usuarios.

Esta navegación puede ser representada en forma de menú, botones o diferentes tipos de interacciones dentro de la página inicial, aquí podemos ver elementos como sliders, cards, enlaces,

imágenes y más elementos que hagan alusión al movimiento y lo que tenemos que cuidar es que de aquí para el usuario sea muy sencillo encontrar todas las demás páginas.

## Navegación / Menu

Esta es la forma en como tu usuario identificará como está organizada toda tu información y así encontrar lo que esté buscando.

Necesitamos asegurarnos de que la información más importante y las categorías principales estén representadas en el menú principal o como parte de la navegación del sitio considerando lo siguiente:

- Usa frases cortas o en todo caso una sola palabra para cada elemento y poder moverte de un lado a otro.
- Usa lenguaje simple para que seas claro con tus usuarios.
- No revuelvas la información, busca el orden de prioridad y sigue ese orden en tu navegación.
- Las URLs de tu aplicación deben de ser intuitivas y no confusas (iE. tusitio.com/contacto)
- Muestra la jerarquía como parte de tu sitio con las subCategorías desplegadas con su categoría padre.
- Secciones como los términos y condiciones, política de privacidad o secciones que no son muy utilizadas llevalas al footer para que no sea tanta información por procesar para el usuario.

Este menú puede tener diferentes formas e incluso se puede modificar la forma de interacción incluso si cambias de tamaño de pantalla, pero los más comunes son NavBar que es la barra de navegación en la parte de arriba de las páginas o los SideBar que significa que el menú se encuentra a un costado de la página o aplicación.

## Categorías y subcategorías

Las categorías sirven para poder agrupar los contenidos e información pensando en hacerlo fácil para que los usuarios accedan al contenido.

Las categorías puedes considerarlas como tu menú principal y de ahí desplegar la información adicional como submenu, recuerda que las categorías principales deben de darle el "mapa" a tu usuario para poder navegar por todo tu sitio, mientras que las subcategorías le dan la información detallada de lo que está buscando.



Si lo vemos con ejemplo de un blog, las categorías podrían ser los temas principales de tus publicaciones y esas categorías las puedes desglosar para hacerlas más detalladas.

O con otro ejemplo en una tienda en línea de ropa lo podemos ver con las categorías de productos pueden ser "zapatos", "chamarras", etc. y las subcategorías podrían ser "niños", "mujer", "hombre", "Tallas chicas", etc. La idea de la organización hace que el uso de tu aplicación sea más sencilla para el usuario.

### Páginas individuales / finales

Estas son las páginas finales a donde puede llegar tu usuario, de aquí ya no hay más jerarquía y es en donde descansa la mayor parte de la información valiosa de tu plataforma.

Estas páginas deben de tener contenido significativo, ya que puede ser que compartan la publicación, que compren el producto o simplemente que hayan encontrado lo que buscan en tu plataforma.

Alguna información adicional que puedes poner aquí puede ser con respecto a contenido similar como links a otras partes de tu sitio o en algunos casos de productos, un formulario de contacto por si necesitan soporte.

La finalidad de estas páginas es darle al usuario lo que necesita, así que debes de ser muy claro de darle las instrucciones necesarias para que llegue sin problemas a tus páginas finales.



## Otros tipos de estructuras de sitios

### Modelo Secuencial

Este modelo es popular por dirigir al usuario por diferentes pasos secuenciales los cuales tienen un orden de finalización antes de poder pasar al siguiente.


Este modelo se utiliza cuando necesitas que tu usuario vaya por un flujo de información sin salirse del carril.



### Modelo de matriz

Este modelo te permite una navegación general por todo el sitio, en la que normalmente se tienen muchas opciones de información específica, la cual es compleja de agrupar en categorías, estos sitios por lo general utilizan links internos del mismo sitio para poder dirigirte a las secciones deseadas.

Uno de los mejores ejemplos de este tipo de modelo es Wikipedia.



WIKIPÉDIA

L'encyclopédie libre

Accueil

Portails thématiques

Article au hasard

Contact

Contribuer

Debuter sur Wikipédia

Aide

Communauté

Modifications récentes

Faire un don

Outils

Pages liées

Suivi des pages liées

Importer un fichier

Pages spéciales

Adresse permanente

Information sur la page

Élément Wikidata

Citer cette page

Imprimer / exporter

Créer un livre

Télécharger comme PDF

Version imprimable

Dans d'autres langues

Afrikaans

አማርኛ

En attente de fr.wikipedia.org...

Non connectéDiscussionContributionsCréer un compteSe connecter

ArticleDiscussionLireModifierModifier le codeHistoriqueRechercher sur Wikipédia

# Page web

Article connexe : Site web.

La **page web** est l'unité de consultation du World Wide Web. Ce terme a une signification pratique ; il n'a pas de définition technique formelle. Elle est conçue pour être consultée à l'aide d'un navigateur web. Elle a une *adresse web*. Techniquement, une page web est souvent constituée d'un document en *Hypertext Markup Language* (HTML) et d'*images*. Cependant, tout type de ressources ou d'assemblage de ressources, textuelles, visuelles, sonores, logicielles, peuvent constituer une page web.

**Sommaire**

1 Terminologie

2 Consultation

3 Bases techniques


4 Mise en page

5 Création d'une page Web

6 Page d'accueil

7 Notes et références

8 Voir aussi



Une page Web.

## Terminologie

[ modifier ] modifier le code ]

En 1999, la *Commission générale de terminologie et de néologie* de France a recommandé d'écrire une **page sur la toile**<sup>1</sup>. Cette recommandation n'a guère été suivie.

## Consultation

[ modifier ] modifier le code ]

Une page web se consulte soit en entrant directement son *adresse Web* dans un *navigateur Web*, soit en suivant un *hyperlien* se trouvant dans une autre page. Les navigateurs les plus utilisés affichent la page sur *moniteur d'ordinateur*, dans une *fenêtre*, et permettent de l'imprimer. Cependant, le langage HTML à la base de la plupart des pages Web a été conçu pour permettre la consultation avec des *dispositifs* variés : *écrans* de toute taille, terminal en *mode texte*, *imprimante*, dispositif *braille*, *synthétiseur vocal*<sup>2</sup>. Le terme *page* est donc généralement réducteur, de par son potentiel d'interactivité avec l'utilisateur.

Al final del día no hay un modelo que sea mejor a otro, cada sitio es único y se debe tratar como tal para lograr la mejor organización de la información pensando siempre en tu usuario.