# 

TEMAS:

* Prototipado básico
  + Diseño Modular
  + Mockups
  + Prototipo interactivo
* HTML
  + Introducción HTML
  + Anatomía HTML
  + Estructura HTML

OBJETIVO:

MARCO TEÓRICO

## Lectura 1: Mockups y Prototipos

**Autor: Carlos Salgado**

**Web: mosaic.uoc.edu**

**MockUp**

Algunos diseñadores sostienen la opinión de que los MockUp no son necesarios, pero son extremadamente útiles a la hora de explorar las decisiones de diseño visual antes de comenzar con la etapa de desarrollo del código.

**¿Qué es un MockUp?**

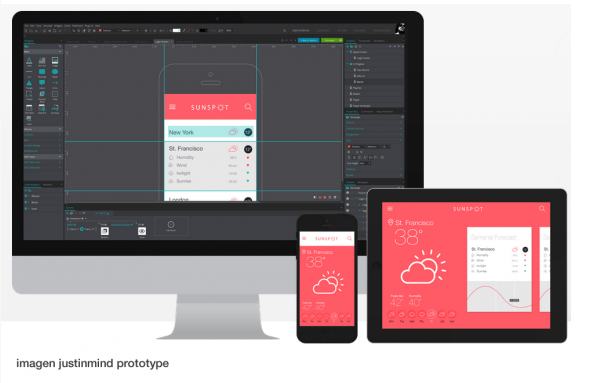
El Mockup o maqueta en castellano es una representación más avanzada del diseño gráfico y comunicativo (desde una visión de navegación y AI) de un proyecto. Hay dos escuelas de pensamiento que se diferencian en como conciben la forma de crear un MockUp, aquellos que creen que la maqueta debe representar el producto final exactamente (alta fidelidad), y aquellos que ven la fase de maqueta como más transitorio y no debe tomar demasiado tiempo (media fidelidad).

Podríamos decir que generalmente es, una composición gráfica completa que ha utilizado el wireframe como plantilla introduciendo todos los elementos gráficos y visuales, convirtiéndose así en un modelo a escala de un producto que se utiliza para demostrar y probar un diseño. El mockup es un medio de representación de la apariencia del producto, y muestra los fundamentos de su funcionalidad. Los MockUp incluyen los detalles visuales, tales como colores, tipografía, etc., y son generalmente estáticas. Al observar un mockup, se debe tener una buena idea de cómo se verá el producto final y una idea aproximada de cómo podría funcionar (incluso si las funciones aún no se han desarrollado).

Es importante distinguir un Mockup de un prototipo. Un prototipo está destinado a funcionar, aunque sea parcialmente, mientras que los Mockup no funcionan. Se componen de las imágenes en pantallas estáticas, sólo “parecen” la interfaz de usuario real.

Se integran elementos con mayor detalle, visualizamos una aproximación de:

* Contenidos (pueden ser imágenes y textos genéricos que no van a ser finalmente usados en el desarrollo)
* Paleta de colores, tomando como referente lo institucional, misional y el público objetivo del proyecto
* Declaraciones CSS
* Dimensiones de áreas de contenido y servicios
* Iconografía



*\*El mockup incluye los elementos del sketch y el wireframe, cada uno es una evolución del anterior*

Cuando presentamos un diseño para el cliente, es necesario tener un Mockup lo más parecido posible al producto final. Los MockUps también se crean para mostrar a nuestro cliente el aspecto general de su nueva página web o programa y es una gran manera de presentar nuestro diseño para que lo apruebe nuestro cliente. Podríamos resumirlo en; “Los MockUp impresionan, dan un aspecto más profesional y venden”.

Por otro lado, pensemos además, que cuando más claro lo vea el cliente, menos quebraderos de cabeza tendremos posteriormente con cambios en la fase de producción con posibles modificaciones. En este momento aun podemos realizar ajustes y mejoras con relativo poco esfuerzo, por todo ello es importante dedicarle el tiempo y la energía necesaria para su creación.

Un MockUp;

* Permite saber qué información existe y donde va;Sabremos dónde va la información antes de empezar a escribir el análisis funcional.
* Puede ser utilizado para la implementación del diseño;Lo que significa que es posible probar su concepto de diseño y tendencia antes de que escribamos una sola línea de código. Los cambios en el concepto de diseño son fáciles y económicamente más bajos en este paso que si lo hacemos posteriormente.
* Es más preciso para los desarrolladores. El desarrollador puede ver cómo debe de ser el producto final, lo que significa que el MockUp de alta fidelidad puede servir como una hoja de especificaciones visual.
* Es una herramienta flexible; Hacer cambios, eliminar o incluso añadir más espacios con nuestras ideas y conceptos es posible más fácilmente
* Es más fácil de presentar a los no diseñadores. Los clientes y partes interesadas prefieren un MockUp de alta fidelidad porque lo que ven es lo que obtendrá.
* Puede servir como parte de nuestra propuesta de presupuesto.Si se trata de un gran cliente, y el trabajo puede resultar lucrativo, de prestigio, o interesante por otras razones, entonces quizás nos puede compensar invertir un tiempo extra y realizar un mockup para presentar junto con el presupuesto como valor añadido para lograr el proyecto. Es importante en este caso que el cliente firme un acuerdo por el cual se compromete a no usar todo o en parte nuestro concepto o diseño sin pagar por ello. No vayamos a pecar de ingenuos y luego llevarnos una sorpresa.

Hemos dedicado bastante tiempo para hablar de beneficios de los mockup para presentar el diseño al cliente y a la importancia que tiene para las ver las relaciones visuales.

Pero hay una tercera cuestión, no menos significativa, por lo que el uso de los mockups es importante, y es que nos ayuda a elaborar el libro de estilo. Tiene su sentido si se piensa en ello, el MockUp es el documento visual del estilo del producto, por ello podemos usar muchos elementos y crear una guía de estilo rápido.

Una guía de estilo es simplemente una recopilación formal de la apariencia de su sitio para que los diseñadores y desarrolladores tengan una guía de referencia rápida y que resalta los puntos más relevantes de nuestro diseño.

Las guías de estilo contienen información de cuál es la norma del diseño, tamaños de márgenes, las fuentes que usamos, sus tamaños dependiendo de su posición, fondos. Debido a que estas son opciones que generalmente incluimos mientras creamos el mockup, la relación entre ambos es fácil de contrastar.

**Algunas cosas que debemos de tener en cuenta.**

* Hay algunos errores que se pueden cometer cuando realizamos un diseño y que al crear nuestro mockup debemos de tener en cuenta;
* Demasiados efectos y detalles: Quizás el error más común, hay que tener en cuenta los detalles que no agreguen valor al diseño y evitarlos, o los detalles que pueden hacer que el código sea luego realmente difícil de crear. Un borde redondeado puede tener cierta dificultad para crearlo mediante CSS en una web, pero si además hay un gradiente de color, o una transparencia mezclada con otros elementos pueden hacer que la fase de desarrollo se complique innecesariamente y aumentar los costes .
* No usar rejillas y no alinear correctamente los elementos: Como comentábamos en el apartado de wireframes, el uso de las rejillas es una de las mejores prácticas que debemos Nuestra aplicación se verá equilibrada y el usuario no se perderá con la vista con un montón de elementos “flotando” en la pantalla. SI hemos creado nuestro wireframe y apoyamos nuestros MockUp en él, no deberíamos tener problemas.
* Si estás realizando una web, no realizar una maqueta para web responsive. Debemos de tener en cuenta, como vimos en la creación de wireframes responsive, que nuestro diseño debe de ajustarse a los distintos tamaños de pantalla, por tanto, deberemos de realizar los MockUps necesarios para representar los distintos diseños de pantalla.
* No usar un esquema de color: Uno de los aspectos más difíciles del diseño para muchos diseñadores, es la selección del color. El esquema de colores de un sitio web o aplicación tiene un gran impacto en el aspecto general de la aplicación y por tanto tendrá también un gran impacto en el usuario. Hay una serie de herramientas que nos pueden ayudar a encontrar la combinación de colores adecuada para un determinado proyecto, muchos de estos recursos los podemos encontrar de forma gratuita en internet.

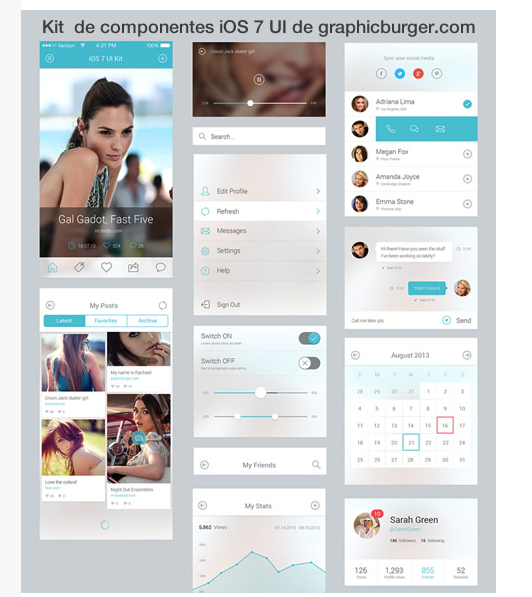
Vamos a enumerar algunas de ellas:

* Kuler (https://color.adobe.com/)
* Color Scheme Designer (http://paletton.com/)
* Colour Lovers (http://www.colourlovers.com/)
* Color Spire (<https://www.colorspire.com/>)
* El cliente no entiende el concepto de diseño en los primeros momentos: Si el cliente no puede entender el concepto de diseño en un par de segundos, es probable que tengamos un problema. Un diseño demasiado innovador o que rompe con muchos de “cánones” habituales en las costumbres de los usuarios a la hora de usar el tipo de aplicación que estemos desarrollando, puede resultar muy creativo, pero quizás poco funcional
* Resulta difícil de leer: Tenemos que asegurarnos de utilizar un contraste alto entre los fondos y las tipografías para facilitar la lectura de los textos. Por lo general, esto significa un color oscuro para la tipografía sobre un fondo muy claro. También, tenemos que asegurarnos de que el tamaño de la letra es adecuado.

**Herramientas para realizar MockUps**

Existen muchas herramientas que nos ayudan a crear nuestros MockUps, podemos usar cualquier programa de dibujo, ya sea vectorial como Illustrator o Inkscape, o programas más orientados al retoque de imágenes como photoshop o fireworks. Uno de los más extendidos es Photoshop, para el que existen gran cantidad de recursos y tutoriales. Además, existen otras herramientas específicamente desarrolladas para el diseño de aplicaciones informáticas o web, que permiten realizar usando una sola aplicación, todo el proceso, creación de wireframes, MockUps, y prototipos que ampliaremos en la sección de herramientas para el prototípico.

Si queremos desarrollar nuestro MockUp mediante programas de dibujo vectorial o de retoque de imágenes podemos encontrar múltiples plantillas o “Toolkits” (cajas de herramientas) llenos de elementos para el diseño de la interfaz de usuario que pueden ahorrar mucho tiempo de creación de nuevos gráficos y que podremos importar en estos programas. Estos kits vienen con botones prefabricados, controles de navegación, patrones de diseño de interfaz de usuario, y otros elementos de interfaz de usuario que nos simplificarán el proceso. La combinación adecuada de los kits de IU, plantillas, fuentes y de otros elementos nos proporcionan los materiales adecuados que modificaremos según resulte necesario.



**Prototipos**

Mientras que los wireframes son un “esbozo” , los MockUps muestran la “sensación”, la textura del diseño, es el prototipo el que da vida a la “experiencia” detrás “de la experiencia del usuario”.

¿Qué es un prototipo?

Un prototipo es un modelo (representación, demostración o simulación) fácilmente ampliable y modificable de un sistema planificado, probablemente incluyendo su interfaz y su funcionalidad de entradas y salidas.

El prototipo es una representación de alto detalle de un proyecto digital. En ella se puede identificar y operar:

* Sistemas de navegación
* Paleta de colores aplicada
* Iconografía
* Experiencia de usuario
* Servicios de ayuda, búsqueda, interacción.
* Otros elementos del proyecto

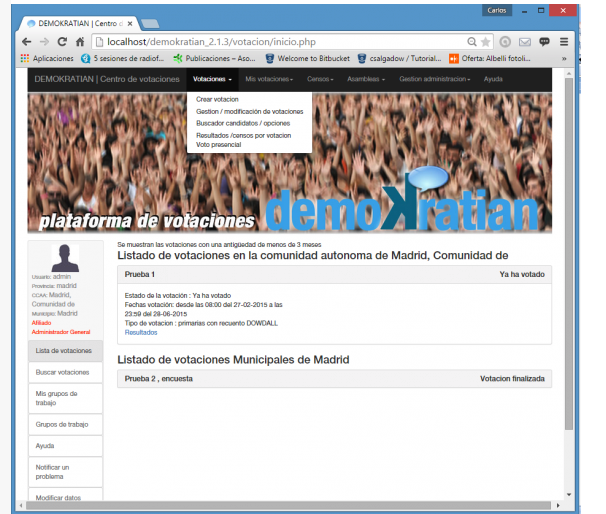
La creación del prototipo debe llevarse a cabo cuando vamos a evaluar la interacción, y sirve para definir aspectos que no quedan claramente reflejados en un boceto de papel o un wireframe no navegable.

Los prototipos son navegables, por lo que sirven para testear elementos de interacción como estados “encima” de botones, validación de formularios, iconos, o cualquier elemento con el que el usuario interactúe. A través de la creación de prototipos, identificamos y solucionamos problemas UX como pueden ser la transición desde la página principal a los resultados de búsqueda sin recargar al usuario con demasiada información. Nos sirve como modelo del comportamiento del sistema que puede ser usado para entenderlo completamente o ciertos aspectos de él y así clarificar los requerimientos.

Durante la fase de creación de prototipos es cuando podemos fusionar nuestras metas conceptuales con nuestra realidad práctica. Es cuando diseño visual y diseño de interacción entran en equilibrio.

Es una fase que puede obviarse en algunos proyectos (si vas a publicar un blog o una sencilla página informativa, probablemente no necesites un prototipo), pero si resulta de gran utilidad en otros, como apps, videojuegos, grandes webs, etc.

Uno de los posibles errores a la hora de lanzar un sistema online o aplicación es pensar sólo en el diseño del modelo de negocio, el desarrollo del producto o las ventas. Antes de que nuestras aplicaciones lleguen a manos de los clientes es importante que los hayamos probado. Invertir en pruebas con usuarios mediante prototipos nos ayudará a ahorrar tiempo y costes y sobre todo a conocer si el producto se adapta a las necesidades de los clientes. Es el paso previo al desarrollo y presentación final del proyecto. Nos sirve para identificar, a partir de pruebas de usuario (beta-tester), las dificultades del proyecto. Esto, en el caso de de la creación de aplicaciones, apps, startups que estén comenzando su proyecto, o grandes webs , es una ventaja inestimable, ya que permite refinar la interfaz en base a datos objetivos.



**Prototipos, UX, usabilidad y accesibilidad.**

Cuando nos dedicamos al desarrollo web o de productos interactivos, vemos lo complicado que resulta lograr crear una web o aplicación informática que cumpla los requisitos de UX, usabilidad y accesibilidad, y que además sea agradable estéticamente al cliente final.

Según la definición ISO, la usabilidad es “la eficacia, la eficiencia y la satisfacción con la que los usuarios alcanzan unos objetivos concretos en un entorno particular” (ISO 9241-11). Un concepto que está ligado a la usabilidad es el de accesibilidad, el cual no se refiere a la facilidad de uso, sino a la posibilidad de acceso. Para algunos clientes es además un requisito el que la aplicación debe de ser “usable” por todos los usuarios potenciales, sin excluir a aquellos con limitaciones individuales. La importancia de ambos términos dependerá de nuestro público objetivo, un diseño accesible implica la necesidad de tener en cuenta otras características para hacerlo más universal y heterogéneo, mientras que el diseño usable se centra más en unos usuarios concretos. Por otro lado, como decíamos anteriormente, la experiencia de usuario (UX) es el conjunto de ideas, sensaciones y valoraciones del usuario resultado de la interacción con un producto; es resultado de los objetivos del usuario, las variables culturales y el diseño del interfaz. Intenta evocar una emoción en el usuario, es decir, intenta que el uso de un determinado objeto o página web sea satisfactorio desde otros puntos de vista más allá de lo utilitario: estética, diversión, identificación, etc.

Estos tres conceptos van muy ligados a la utilidad de crear un prototipo y la realización de pruebas por beta-tester. El objetivo es conseguir la mayor cantidad de información posible de la interacción entre el usuario y el prototipo. Su evaluación asegura que los productos sean fáciles de usar y se ajusten a nuestro público. No hay que perder de vista que el producto será utilizado por personas, individuos que probablemente no se parezcan a nosotros y no tengan el mismo gusto, pero quieren que nuestro producto funcione. Gracias a las pruebas de usuario con prototipos, seremos capaces de detectar problemas y solucionarlos antes del comienzo de la fase de producción, ahorrando así bastantes problemas y costes que habría que de intentar solucionar cuando ya está creada toda la programación, o cuando se ha lanzado ya el producto. Ahorra horas de desarrollo, ya que son necesarias menos versiones de la aplicación.

En esta fase conviene tener muy en cuenta los 10 principios de diseño basados en el usuario, que definió Jakob Nielsen en 1990 , que siguen siendo un referente importantísimo para evaluar la usabilidad de un sitio web.

**Algunos consejos a la hora de crear un prototipo**

No hay una fórmula general que podamos aplicar a la hora de crear un prototipo, ya que este dependerá de muchos factores, pero si hay algunas cosas que podemos tener en cuenta:

Cada sección de una web tiene un objetivo y este debe quedar claro al ver el prototipo.

Dependiendo del tipo de prototipo, si es de una web, o de una aplicación multimedia, el prototipo deberá ser más o menos completo, en una web sencilla puede tener solo las páginas más importantes que sean distintas, y en otro tipo de aplicaciones es posible que debamos desarrollarlo completamente.

Es conveniente que en la creación del prototipo participe también el equipo de diseño. Aunque se puede considerar que es un trabajo más relacionado con los equipos de desarrollo, una buena comunicación con el resto de personas implicadas en el producto puede solventar muchos problemas durante la fase de creación del prototipo.

Diseñar a tamaño real, respetar el tamaño de las imágenes y contenidos e incluir la estructura de navegación.

**Herramientas para la creación de prototipos.**

Existen múltiples formas de crear un prototipo, como comentábamos anteriormente, dependerá mucho del tipo de aplicación que estemos realizando, podemos desde crear una aplicación prácticamente funcional, usando HTML, CSS, etc. si por ejemplo estamos creando una web, o también podemos usar alguna de las aplicaciones que existen en el mercado para realizar un prototipo.

A continuación podéis ver una lista de alguna de las aplicaciones más usadas

**Justinmind:** Herramienta profesional para realizar un prototipo de sitios web, aplicaciones de software y aplicaciones móviles. Puede trabajar con Windows o con Mac.

**Axure RP:** Es una herramienta de realización de prototipos profesional que permite crear wireframes para hacer el pre-diseño de una página web así como MockUps. Disponible tanto para plataforma Windows como Mac. Puedes dibujar tanto wireframes estáticos como interactivos que simulan una experiencia de navegación del usuario real..

**Balsamiq:** Con ella puedes hacer prototipos interactivos de webs. Puedes usar esta herramienta como un servicio web o bien descargarla en tu equipo.

Protoshare Está basado en web, y tiene soporte para twitter bootstrap, además permite hacer simulaciones sobre distinto dispositivos

**Jumpchart:** es una aplicación de planificación de webs basadas en el navegador la cual posibilita esbozar el contenido de la página web. Es posible crear wireframes tanto estático como interactivos que simulan la navegación entre las páginas web de la maqueta.

**FlairBuilder**: herramienta web que te permite hacer bocetos de tus webs y de tus Apps para iPhone. No tiene posibilidad de colaborar con otras personas ni tampoco es posible exportar características.

**iPlotz**: Esta herramienta permite hacer maquetas navegables de sitios web y de aplicaciones. Lo puedes descargar en tu ordenador (Windows/ Mac) o bien puedes usar el servicio vía web.

**MockFlow:** Herramienta web muy sencilla de utilizar para diseñar sitios web y aplicaciones de software.

**Mockingbird:** Se trata de un servicio web gratuito con el que se pueden hacen prototipos de páginas web.

# 

# Lectura 2: HTML

El Lenguaje de Marcado de Hipertexto (HTML) es el código que se utiliza para estructurar y desplegar una página web y sus contenidos. Por ejemplo, sus contenidos podrían ser párrafos, una lista con viñetas o el uso de imágenes y tablas de datos. Como lo sugiere el título, este artículo te dará una comprensión básica de HTML y cúal es su función.

HTML no es realmente un lenguaje de programación; es un *lenguaje de marcado*, y es usado para decirle al navegador cómo desplegar las páginas web que visitas. Puede ser tan complicado o simple como el diseñador web lo desee. HTML consiste en una serie de elementos, que usas para encerrar diferentes partes del contenido para que éstos se muestren o actúen de cierta manera. Las etiquetas de encierre pueden hacer de una palabra o una imagen un hipervínculo hacia otro sitio, pueden convertirpalabras en cursivas, agrandar y achicar las fuentes, y así. Por ejemplo, mira el siguiente contenido:

Mi gato es muy gruñon

Si queremos especificar que se trata de un párrafo, podríamos encerrar el texto con la etiqueta de párrafo (<p>):

<p>Mi gato es muy gruñon</p>

**Anatomía de un elemento HTML**

Exploremos este párrafo en mayor profundidad.



Las partes principales de nuestro elemento son:

1. **La etiqueta de apertura:** Esta consiste en el nombre del elemento (en este caso, p), encerrada en paréntesis angulares de apertura y cierre. Esto indica donde el elemento comienza, o empieza a tener efecto — en este caso donde comienza el párrafo.
2. **La etiqueta de cierre:** Es lo mismo que la etiqueta de apertura, excepto que incluye una barra diagonal antes del nombre del elemento. Esto indica dónde termina el elemento— en este caso dónde finaliza el párrafo. No incluir una etiqueta de cierre es un error común de principiante, y puede conducir a resultados extraños.
3. **El contenido:** Este es el contenido del elemento, que en este caso es sólo texto.
4. **El elemento:** El elemento lo conforman la suma de la etiqueta de apertura, la etiqueta de cierre y el contenido.

**Elementos anidados**

También puedes poner elementos dentro de otros elementos — esto se llama anidamiento. Si quisiéramos decir que nuestro gato es MUY gruñón, podríamos encerrar la palabra "muy" en un elemento <strong>, lo que significa que la palabra debe enfatizarse.

**<p>**Mi gato es **<strong>**muy**</strong>** gruñón.**</p>**

Debes asegurarte de que los elementos estén apropiadamente anidados: en el ejemplo anterior primero abrimos el elemento p y luego el elemento strong, por lo tanto, debemos cerrar primero el elemento strong, y luego el p. Lo siguiente es incorrecto:

**<p>**Mi gato es **<strong>**muy gruñon.**</p></strong>**

Los elementos tienen que abrirse y cerrarse correctamente para que estén claramente dentro o fuera uno del otro. Si se superponen como en el ejemplo anterior, el navegador tratará de averiguar lo que tratas de expresar y es posible que los resultados no sean los esperados. ¡Así que no lo hagas!

**Elementos de bloque y elementos en línea**

Hay dos categorías de elementos en HTML, que es importante que conozcas — elementos de bloque y elementos en línea.

* Los elementos de bloque forman un bloque visible en la página — aparecerán en una nueva línea después de cualquier contenido anterior y cualquier contenido que vaya después también aparecerá en una nueva línea. Los elementos de bloque tienden a ser elementos estructurales en la página que representan por ejemplo párrafos, listas, menús de navegación, pies de página, etc. Un elemento bloque no estaría anidado dentro de un elemento en línea, pero podría estar anidado dentro de otro elemento a nivel de bloque.
* Los elementos en línea son aquellos que están contenidos dentro de elementos de bloque y rodean solo pequeñas partes del contenido del documento, no párrafos enteros ni agrupaciones de contenido. Un elemento en línea no hará que se genere una nueva línea aparezca en el documento y normalmente aparecen dentro de un párrafo o texto, por ejemplo es el caso de un elemento <a> (hipervínculo) o elementos de énfasis como <em> o <strong>.

En los siguientes ejemplos:

**<em>**primero**</em><em>**segundo**</em><em>**tercero**</em>**

**<p>**cuarto**</p><p>**quinto**</p><p>**sexto**</p>**

<em> es un elemento en línea, así como puedes observar, los tres primeros elementos se sitúan en la misma línea uno tras otro sin espacio entre ellos. Sin embargo <p> es un elemento a nivel de bloque, así que cada elemento aparece en una nueva línea, con un espaciado encima y debajo de cada uno (el espaciado es debido al estilo CSS por defecto que aplica el navegador a los párrafos)

**Elementos vacíos**

No todos los elementos siguen el patrón de arriba de etiqueta de apertura, contenido, etiqueta de cierre. Algunos elementos consisten solo en una etiqueta única, que se utiliza generalmente para insertar/incrustar algo en el documento en el lugar donde es incluida. Por ejemplo, el elemento <img> inserta una imagen en la posición de la página donde es incluída:

<img src="<https://raw.githubusercontent.com/mdn/beginner-html-site/gh-pages/images/firefox-icon.png>">

Esto mostraría lo siguiente en tu página:



**Atributos**

Los elementos pueden también tener atributos, que tienen el aspecto siguiente:



Los atributos contienen información extra sobre el elemento que no se mostrará en el contenido. En este caso, el atributo class asigna al elemento un identificador que puede ser utilizarse para dotarlo de información de estilo y otras cosas.

Un atributo debería tener:

1. Un espacio entre él y el nombre del elemento (o atributo previo, si el elemento ya tiene uno o más atributos.)
2. El nombre del atributo, seguido por un signo igual.
3. Un valor del atributo, con comillas de apertura y cierre alrededor de él.

Otro ejemplo de un elemento es [<a>](https://developer.mozilla.org/es/docs/Web/HTML/Elemento/a) — que significa ancla, y creará un hipervínculo en la parte del texto encerrado entre las etiquetas. Puede tener una serie de atributos entre los que se encuentran los siguientes:

* **href:** El valor de este atributo indica la dirección web a la que se quiera que punte el enlace, que será hacia donde nos lleve el navegador cuando se hace un clic sobre el elemento. Por ejemplo, href="https://www.mozilla.org/".
* **title:** El atributo title añade información adicional sobre el enlace, como puede ser el título de la página que estás enlazando. Por ejemplo, title="La página de inicio de Mozilla". Esta información aparecerá cuando se le pase el ratón por encima.
* **target:** El atributo target especifica el contexto de navegación que será usado para mostrar el enlace. Por ejemplo, target="\_blank" mostrará el enlace en una nueva pestaña. Si quieres mostrar el enlace en la pestaña actual solo omite este atributo.

**Anatomía de un documento HTML**

Hasta aquí las bases de los elementos HTML de manera individual, pero no son muy útiles por sí mismos. Ahora veremos como los elementos individuales se combinan para formar una página HTML completa:

<!DOCTYPE html>  
<html>  
 <head>  
 <meta charset="utf-8">  
 <title>My test page</title>  
 </head>  
 <body>  
 <p>This is my page</p>  
 </body>  
</html>

Aquí tenemos:

1. **<!DOCTYPE html>:** El doctype. En sus inicios, cuando el HTML llevaba poco tiempo (alrededor de 1991/2), los doctypes servían como enlaces al conjunto de reglas que la página HTML debía seguir para considerarse buen HTML, lo que podía significar comprobación automática de errores y otras funcionalidades útiles. Un doctype de

aquella época podía parecerse a:

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "<http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd>">

En la actualidad se ignora y se considera un legado histórico que hay que incluir para que todo funcione correctamente. <!DOCTYPE html> es la secuencia de caracteres más corta se acepta como doctype válido y esto es lo único que realmente necesitas saber.

1. **<html></html>:** El elemento <html>. Este elemento engloba todo el contenido de la página y es conocido en ocasiones como el elemento raíz.
2. **<head></head>:** El elemento <head> (cabecera). Este elemento actúa como contenedor para todos los parámetros que quieras incluir en el documento HTML que NO SERÁ visible a los visitantes de la página. Incluye cosas como palabras clave y la descripción de la página que quieras mostrar en los resultados de búsqueda, así como la hoja de estilo para formatear nuestro contenido, declaraciones de codificación de caracteres y más. Aprenderás más acerca de esto en el siguiente artículo de la serie.
3. **<meta charset="utf-8">:** Este elemento establece que tu documento HTML usará la codificación uft-8, que incluye la gran mayoría de caracteres de todos los lenguajes humanos conocidos. En resumen: puede tratar cualquier contenido de texto que pongas en tu documento. No hay razón para no configurarlo y puede te ayudar a evitar problemas más adelante.
4. **<title></title>:** Este elemento establece el título de tu página, que aparece en la pestaña/ventana de tu navegador cuando la página se carga y se utiliza para describir la página cuando la agregas a tus marcadores o la marcas como favorita.
5. **<body></body>:** El elemento <body>. Contiene todo el contenido que quieres mostrar a los usuarios cuando visitan tu página, ya sea texto, imágenes, vídeos, juegos, pistas de audio reproducibles o cualquier otra cosa.

**Comentarios HTML**

En HTML, como en la mayoría de los lenguajes de programación existen mecanismos para escribir comentarios en el código. Los comentarios son ignorados por el navegador y son invisibles para el usuario. Su propósito es permitir comentar el código para aclarar su funcionamiento, explicar lo que hacen las diferentes partes del código , etc. Esto es muy útil como recordatorio del trabajo realizado si vuelve a ver código en el que no ha trabajado durante meses o si entrega su código a otra persona para que trabaje en él.

Para convertir una sección de contenido dentro de su archivo HTML en un comentario, debe envolverlo en los marcadores especiales <! - y ->, por ejemplo:

<p>I'm not inside a comment</p>

<!-- <p>I am!</p> →

el primer párrafo aparece, pero el segundo no.

Herramientas

La confirmación de los ejercicios y seguimiento a la guía debe ser en el editor de su preferencia **Sublime Text** (https://www.sublimetext.com/)

Ejercicios

Explorar los atributos de HTML y construir su CV, aquí la guia oficial de elementos HTML5 https://html.spec.whatwg.org/multipage/indices.html#elements-3