Definición de HTML y/o XHTML

sigla en inglés de ***HyperText Markup Language*** (lenguaje de marcas de hipertexto), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros. Es un estándar a cargo del *World Wide Web Consortium* (W3C) o Consorcio WWW, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación. Se considera el lenguaje web más importante siendo su invención crucial en la aparición, desarrollo y expansión de la World Wide Web (WWW). Es el estándar que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales han adoptado.

(*eXtensible HyperText Markup Language*), es básicamente HTML expresado como XML válido. Es más estricto a nivel técnico, pero esto permite que posteriormente sea más fácil al hacer cambios o buscar errores entre otros. En su versión 1.0, XHTML es solamente la versión XML de HTML, por lo que tiene, básicamente, las mismas funcionalidades, pero cumple las especificaciones, más estrictas, de XML. Su objetivo es avanzar en el proyecto del World Wide Web Consortium de lograr una web semántica, donde la información, y la forma de presentarla estén claramente separadas. La versión 1.1 es similar, pero parte a la especificación en módulos. En sucesivas versiones la W3C planea romper con los tags clásicos traídos de HTML.

Estándares de HTML y/o XHTML

Existen múltiples estándares HTML (HTML 4.01, XHTML 1.0, HTML 5, ...). Al principio del documento debe indicársele al navegador qué versión se va a utilizar, ya que cada una de ellas tiene unas reglas distintas para interpretar las etiquetas.

Si no se le indica al navegador el estándar que se está usando o si se le indica uno pero no se siguen correctamente las normas del mismo, el navegador intenta interpretar lo que se quería hacer. El problema de esto es que cada navegador trata de interpretarlo a su manera y puede llegar a conclusiones distintas. Por eso se deben usar los estándares: si sigues el estándar todos los navegadores saben lo que quieres hacer, no tienen que procurar adivinarlo.

HTML semántico.

Las etiquetas HTML no le dicen al navegador la apariencia que deben tener los elementos dentro de la página web, solamente señalan cuál es su función. La mayoría de los navegadores muestran los elementos de una forma similar; por ejemplo, un encabezado h1 se mostrará en letra grande y en negrita en casi todos los navegadores.

Hacer HTML semántico significa marcar cada elemento según su función dentro del documento (valor semántico). Por ejemplo, no marcar un texto como si fuese un encabezado sólo para que coja esa apariencia, ni dejar de marcar un encabezado como tal para darle una apariencia distinta. La apariencia de los elementos debe cambiarse con CSS, para mantener separada la estructura (HTML) de la apariencia (CSS).

El HTML semántico favorece la accesibilidad y la usabilidad. Por ejemplo, algunas herramientas (como JAWS) y navegadores (como Opera) pueden coger todos los encabezados de la página web y hacer un índice con ellos, permitiendo saltar de un encabezado a otro. Evidentemente, para que esto funcione, los títulos tienen que estar marcados como tales (h1, h2, ...).

Navegadores.

Un **navegador web** es un programa que permite ver la información que contiene una página web. El navegador interpreta el código, HTML generalmente, en el que está escrita la página web y lo presenta en pantalla permitiendo al usuario interactuar con su contenido y navegar.
Es de vital importancia contemplar los distintos navegadores con los que los usuarios van a utilizar nuestras páginas. En teoría, los estándares web publicados por el W3C deberían permitir que las páginas fueran visualizadas exactamente igual en todos los navegadores. La realidad, sin embargo, es distinta: **Cada navegador (especialmente, Internet Explorer) implementa diferencias que pueden hacer necesario el uso de técnicas "especiales" para que nuestros portales se muestren de la misma forma en todos los navegadores**.

Google Chrome

Google Chrome es un navegador web de código propietario desarrollado por Google y compilado con base en componentes de código abierto (proyecto Chromium).
Chrome es actualmente el navegador más utilizado de internet, y **el 59% de los usuarios de Euskadi.eus utilizaron este navegador durante el año 2019**. Está disponible gratuitamente para diversas plataformas (Windows, MacOS, Android, iOS, Linux...).
Google Chrome es totalmente complatible con los estándares HTML5 y CSS3.

Safari

Safari es un navegador web de código cerrado desarrollado por Apple Inc. Está disponible para Mac OS X, iOS (el sistema usado por el iPhone, el iPod Touch y el iPad) y Microsoft Windows.
El 13% de los usuarios de Euskadi.eus utilizaron este navegador durante el año 2019.
Safari es totalmente complatible con los estándares HTML5 y CSS3.

Microsoft Internet Explorer

Microsoft Internet Explorer, conocido comúnmente como IE, es un navegador web desarrollado por Microsoft para el sistema operativo Microsoft Windows desde 1995. Ha sido el navegador web más utilizado durante años, con un pico máximo de cuota de utilización del 95% entre el 2002 y 2003. Sin embargo, dicha cuota de mercado ha disminuido paulatinamente con los años. **El 11% de los usuarios de Euskadi.eus utilizaron este navegador durante el año 2019**. Microsoft anunció que a partir de Windows 10 dejará de publicar versiones de este navegador para sustituirlo por Edge.
Su versión más reciente es la 11. **La version instalada por defecto en los ordenadores del Gobierno Vasco es la 9 (equipos con Windows 7) y la 11 (equipos con Windows 10). Se trata del navegador "oficial" para las aplicaciones internas y de intranet del Gobierno Vasco. La versión 9 no es totalmente compatible con los estándares HTML5 y CSS3**

Mozilla Firefox

Mozilla Firefox es un navegador web libre y de código abierto descendiente de Mozilla Application Suite y desarrollado por la Fundación Mozilla. **El 10% de los usuarios de Euskadi.eus utilizaron este navegador durante el año 2019**.

Firefox puede ser utilizado en varios sistemas operativos (Windows, MacOS, Android, iOS, Linux...).

Firefox soporta completamente los estándares HTML5 y CSS3.

Microsoft Edge

Es el navegador integrado por defecto por Microsoft en su sistema operativo Windows 10, y está llamado a sustituir a Internet Explorer (producto que MS dejará de desarrollar y soportar a partir de 2019). **El 4% de los usuarios de Euskadi.eus utilizaron este navegador durante el año 2019.**

Todos los equipos del Gobierno Vasco con Windows 10 tienen este navegador integrado. Ademas de para Windows, existen versiones de Edge para Android e iOS.

Edge es compatible con los estándares HTML5 y CSS3.

Opera

Opera es un navegador web y suite de Internet creado por la empresa noruega Opera Software, capaz de realizar múltiples tareas como navegar por sitios web, gestionar correo electrónico, contactos, fuentes web, charlar vía IRC y funcionar como cliente BitTorrent. **Aproximadamente el 1% de los usaros de Euskadi.eus entraron usanfo este navegador**. Funciona en una gran variedad de sistemas operativos, incluyendo Microsoft Windows, MacOS, Android, iOS, GNU/Linux y FreeBSD.

Servidor WEB

En este artículo veremos que son los servidores, cómo funcionan y por qué son importantes.

Con "Servidor web" podemos referirnos a hardware o software, o a ambos trabajando juntos.

1. En cuanto a hardware, un servidor web es una computadora que almacena el software de servidor web, y los archivos que componen un sitio web (por ejemplo, documentos HTML, imágenes, hojas de estilos CSS y archivos JavaScript). Un servidor web -hardware- se conecta a internet y mantiene el intercambio de datos con otros dispositivos conectados a la web.
2. En cuanto a software, un servidor web tiene muchas partes que controlan cómo los usuarios de la web obtienen acceso a los archivos alojados en el servidor; es decir, mínimamente, un *servidor HTTP*. Un servidor HTTP es una pieza de software capaz de comprender URLs (direcciones web) y HTTP (el protocolo que tu navegador usa para obtener las páginas web). Un servidor HTTP puede ser accedido a través de los nombres de dominio de los sitios web que aloja, y entrega el contenido de esos sitios web alojados al dispositivo del usuario final.

Al nivel más básico, cuando un navegador necesita un archivo que está almacenado en un servidor web, el navegador requerirá el archivo al servidor mediante el protocolo HTTP. Cuando la petición alcanza al servidor web correcto (hardware), el *servidor HTTP* (software) acepta la solicitud, encuentra el documento requerido y lo envía de regreso al navegador, también a través de HTTP. (Si el servidor no encuentra el documento requerido, devuelve una respuesta 404 en su lugar.)

Tipo de Servidores:

Los servidores pueden ser Estático y Dinámico

Un **servidor web estático**, o pila, consiste en una computadora (hardware) con un servidor HTTP (software). Llamamos a "estático" a este servidor porque envía los archivos que aloja "tal como se encuentran" (sin modificarlos) a tu navegador.

Un **servidor web dinámico** consiste en un servidor web estático con software adicional, habitualmente una *aplicación servidor* y una *base de datos*. Llamamos "dinámico" a este servidor porque la *aplicación servidor* actualiza los archivos alojados, antes de enviar el contenido a tu navegador mediante el *servidor HTTP*.

Por ejemplo, para producir las páginas web que finalmente ves en tu navegador, la *aplicación servidor* podría llenar una plantilla HTML con contenido obtenido de una base de datos. Sitios como MDN o Wikipedia tienen miles de páginas web, que no son realmente archivos HTML. Típicamente, este tipo de sitios se componen de unas pocas plantillas HTML y una gigantesca base de datos, en vez de miles de documentos HTML estáticos. Esto hace más fácil el mantenimiento y entrega del contenido.

Editores Web

Un **editor de páginas web** es una aplicación diseñada con el fin de facilitar la creación y edición de documentos HTML o XHTML. Su complejidad puede variar desde la de un simple editor de texto plano, entornos WYSIWYG, hasta editores WYSIWYM.

Tipos de Editores

a) Editor de texto sin formato

También llamado editor de texto plano. Este tipo de editor suele ser muy sencillo. Dos ejemplos son Notepad o Bloc de Notas (incluido en Windows) y Kate (GNU/Linux). Con cualquiera de los dos bastaría para escribir las líneas de código necesarias para diseñar una página Web.

Existen editores de texto específicamente diseñados para la edición Web, que como Kate, incluyen dentro de su simplicidad coloreado de sintaxis y las etiquetas de marcado usuales necesarias en el lenguaje de hipertexto. Dichos editores incluyen una serie de botones para insertar rápidamente las etiquetas, o combinaciones de estas, más corrientes, salvar el documento con un clic y visionarlo posteriormente en una nueva ventana.

b) Editor de texto con ventana

Es una versión ligeramente más sofisticada que la anterior. Suelen constar de un par de ventanas. Una área de trabajo, donde se teclea el código HTML y el texto que se quiere incluir en la página, y en la otra se visualiza el resultado en tiempo real. En otras palabras se obtiene una pre visualización del documento generado. Lo que significa que no se tiene que guardar el documento, previamente, antes de volver a abrirlo con el navegador para comprobar el resultado.

c) Editores WYSIWYG

El término WYSIWYG es el acrónimo de What You See Is What You Get, que traducido al castellano quiere decir: "lo que tú ves es lo que obtienes", en los que de manera visual se pueden colocar distintos elementos sobre una vista previa de la página, encargándose el programa de generar el documento HTML. La manera de trabajar en este tipo de editores, es muy similar a la que se usa cuando se trabaja con un procesador de texto. Esto quiere decir que un usuario no tiene por qué teclear las etiquetas del lenguaje de marcado. En lugar de eso, el usuario escribe el texto, lo

formatea, e inserta las imágenes en los lugares deseados, trabajando igual a como lo haría con Writer, (el incluido en la suite ofimática OpenOffice.org y Word. Posteriormente el editor transforma la vista por pantalla en código HTML perfectamente configurado.

Cualquiera de estos editores son una buena alternativa a los editores de texto simple. Los mejores editores HTML señalan las líneas de código mediante distintos tipos de fuente a las usadas en el texto introducido directamente por teclado. Además, proporcionan la posibilidad de volver hacia atrás entre los distintos tipos de vista.

Comunicación entre el servidor y el navegador

Cómo funciona la web proporciona una vista simplificada de lo que sucede cuando ves una página web en un navegador web de tu computador o teléfono.

Esta teoría no es esencial para escribir código web a corto plazo, pero en poco tiempo empezarás a beneficiarte realmente al entender lo que está sucediendo en el fondo.

Los clientes y servidores

Las computadoras conectadas a la web se llaman **clientes** y **servidores**. Un diagrama simplificado de cómo interactúan se vería así:

- Los clientes son dispositivos de los usuarios conectados a Internet (por ejemplo, tu ordenador conectado a la red Wi-Fi o el teléfono conectado a la red de telefonía móvil) y el software que se encuentra disponible y permite acceder a Internet en dichos dispositivos (normalmente, un navegador web como Firefox o Chrome).
- Los servidores son computadores que almacenan páginas web, sitios o aplicaciones. Cuando un dispositivo cliente quiere acceder a una página web, una copia de la página web se descarga desde el servidor en el equipo cliente y se muestra en el navegador web del usuario.

Las otras partes de la caja de herramientas

El cliente y el servidor que describimos anteriormente, no cuentan toda la historia. Hay muchas otras partes involucradas y vamos a describirlas a continuación.

Por ahora, imaginemos que la web es un camino. En un extremo de la carretera, está el cliente, que es como tu casa. En el extremo opuesto del camino, está el servidor, que es una tienda en la que deseas comprar algo.

Además del cliente y el servidor, también tenemos que saludar a:

- **Tu conexión a Internet**: permite enviar y recibir datos en la web. Básicamente es el recorrido entre tu casa y la tienda.
- **TCP/IP**: **Protocolo de Control de Transmisión** y **Protocolo de Internet**, son los protocolos de comunicación que definen cómo deben viajar los datos a través de la web. Esto es, los medios de transporte que te permiten hacer un pedido, ir a la

tienda y comprar los productos. En nuestro ejemplo, podría ser un coche, una bicicleta o tus propios pies.

- **DNS:** los servidores del **Sistema de Nombres de Dominio** (DNS, por sus siglas en inglés), son como una libreta de direcciones de sitios web. Cuando escribes una dirección web en el navegador, el navegador busca los DNS antes de recuperar el sitio web. El navegador necesita averiguar en qué servidor vive el sitio web y así enviar los mensajes HTTP al lugar correcto (ver más abajo). Esto es como buscar la dirección de la tienda para que puedas llegar a ella.
- **HTTP:** el **Protocolo de Transferencia de Hipertexto** es un protocolo de aplicación que define un idioma para que los clientes y servidores se puedan comunicar. Esto es como el idioma que utilizas para ordenar tus compras.
- **Archivos componentes**: un sitio web se compone de muchos archivos diferentes, que son como las diferentes partes de los productos que comprarás en la tienda. Estos archivos se dividen en dos tipos principales:

    a) **Archivos de código**: los sitios web se construyen principalmente con HTML, CSS y JavaScript, aunque te encontrarás con otras tecnologías más adelante.
    b) **Recursos:** este es un nombre colectivo para el resto de materiales que conforman un sitio web, como imágenes, música, video, documentos de Word, archivos PDF, etc.

Origen de la Web

Hasta el verano de 1991, el uso de Internet era bastante restringido; todo se reducía a usuarios en Universidades y centros de Investigación repartidos por el mundo que accedían a los servicios que proporcionaba la Red, mediante programas cuya utilización exigía secuencias de comandos totalmente crípticos para el profano. Casi todas las máquinas utilizaban el SO. Unix, cuyo manejo no tiene precisamente fama de sencillo, y cuyo intérprete de comandos (un sistema de órdenes por las que el operador indica al sistema que quiere que haga), no es lo que podríamos calificar como muy "intuitivo". El resumen era que si uno quería, por ejemplo, conectar con un servidor FTP en un organismo o institución remota, consultar sus fondos, "bajarse" un fichero e imprimirlo, tenía que hacer todo esto a "pedal" (la Red solo era utilizada por especialistas).

Así las cosas, el británico Berners-Lee (al que se considera el padre de la Web), que a la sazón trabajaba en el CERN de Ginebra, empezó a escribir un programa que le permitiera almacenar información. De modo magistral, dio forma y aplicación a un par de conceptos que ya habían sido formulados anteriormente de forma más o menos vaga y genérica: El hipervínculo, que conducía directamente al concepto de hipertexto, de ahí al de páginas HTML (páginas Web) que a su vez, darían origen a un nuevo servicio de Internet (mejor diríamos una nueva forma de usar la Red) que acabaría arrasando, y a un nuevo paradigma de arquitectura de la información: Los "Hipermedia". Las páginas de hipertexto, con sus hipervínculos enlazando información en cualquier parte del mundo,

tejen una telaraña mundial, de ahí el nombre que recibió, Telaraña Mundial, "World Wide Web", abreviadamente "La Web"; WWW o W3.

Web 2.0

Es un concepto que se acuñó en 2003 y que se refiere al fenómeno social surgido a partir del desarrollo de diversas aplicaciones en Internet. El término establece una distinción entre la primera época de la Web (donde el usuario era básicamente un sujeto pasivo que recibía la información o la publicaba, sin que existieran demasiadas posibilidades para que se generara la interacción) y la revolución que supuso el auge de los blogs, las redes sociales y otras herramientas relacionadas. La Web 2.0 es una evolución de la Web que la convierte en una plataforma donde los contenidos pueden ser creados, compartidos, gestionados y modificados por los usuarios.

La Web 3.0

La noción de web se emplea para nombrar a una red informática y, en especial, a Internet. La idea de web 3.0 alude a una especie de extensión o de formato particular de la red tradicional.
Web 3.0 o web semántica, es una expresión que se utiliza para describir la evolución del uso y la interacción de las personas en internet a través de diferentes formas entre las que se incluyen la transformación de la red en una base de datos, un movimiento social con el objetivo de crear contenidos accesibles por múltiples aplicaciones non-browser (sin navegador), el empuje de las tecnologías de inteligencia artificial, la web semántica, la Web Geoespacial o la Web 3D. La expresión es utilizada por los mercados para promocionar las mejoras respecto a la Web 2.0. Esta expresión Web 3.0 apareció por primera vez en 2006 en un artículo de Jeffrey Zeldman, crítico de la Web 2.0 y asociado a tecnologías como AJAX. Actualmente existe un debate considerable en torno a lo que significa Web 3.0, y cuál es la definición más adecuada.

W3C y estándares

El World Wide Web Consortium, más conocido como W3C, es un consorcio internacional de organizaciones vinculadas a las tecnologías de información que busca promover la evolución de la Red a través del establecimiento de distintas pautas para su estandarización. El propio padre de la Web, Tim Berners-Lee, fundó el W3C con el objetivo de garantizar una Web universalmente accesible, más allá de las diferencias de idioma, navegado r, sistema operativo, plataforma, localización geográfica o aptitudes tecnológicas.
Para lograr esta meta, el W3C propone a diseñadores, programadores, editores y empresas de software dedicadas a web browsers el cumplimiento de ciertas directivas de desarrollo. El W3C ofrece pautas sobre la mayoría de los lenguajes y tecnologías de uso común en la construcción de páginas y aplicaciones web, como HTML, XHTML, CSS, XML y sus distintas versiones, entre muchas otras.

Etiqueta (lenguaje de marcado)

Una etiqueta (términos a veces reemplazados por el anglicismo tag) es una marca con clase que delimita una región en los lenguajes basados en XML. También puede referirse a un conjunto de juegos informáticos interactivos que se añade a un elemento de los datos para identificarlo (Oxford English Dictionary). Esto ocurre, por ejemplo, en los archivos MP3 que guardan información sobre una canción así como sobre el artista que la ha cantado o compuesto.

SGML

El lenguaje de marcado generalizado estándar o SGML (por sus siglas en inglés de Standard Generalized Markup Language) (SGML; ISO 8879: 1986) es un estándar para definir lenguajes de marcado generalizados para documentos. ISO 8879 define el Anexo A.1 de marcado generalizado:

XML

XML proviene de eXtensible Markup Language (“Lenguaje de Marcas Extensible”). Se trata de un metalenguaje (un lenguaje que se utiliza para decir algo acerca de otro) extensible de etiquetas que fue desarrollado por

el Word Wide Web Consortium (W3C), una sociedad mercantil internacional que elabora recomendaciones para la World Wide Web.

HTML

sigla en inglés de HyperText Markup Language (lenguaje de marcas de hipertexto), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros. Es un estándar a cargo del World Wide Web Consortium (W3C) o Consorcio WWW, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación. Se considera el lenguaje web más importante siendo su invención crucial en la aparición, desarrollo y expansión de la World Wide Web (WWW). Es el estándar que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales han adoptado.

XHTML

(eXtensible HyperText Markup Language), es básicamente HTML expresado como XML válido. Es más estricto a nivel técnico, pero esto permite que posteriormente sea más fácil al hacer cambios o buscar errores entre otros. En su versión 1.0, XHTML es solamente la versión XML de HTML, por lo que tiene, básicamente, las mismas funcionalidades, pero cumple las especificaciones, más estrictas, de XML. Su objetivo es avanzar en el proyecto del World Wide Web Consortium de lograr

una web semántica, donde la información, y la forma de presentarla estén claramente separadas. La versión 1.1 es similar, pero parte a la especificación en módulos. En sucesivas versiones la W3C planea romper con los tags clásicos traídos de HTML.

HTML5

(HyperText Markup Language, versión 5) es la quinta revisión importante del lenguaje básico de la World Wide Web, HTML. HTML5 especifica dos variantes de sintaxis para HTML: una «clásica», HTML (text/html), conocida como HTML5, y una variante XHTML conocida como sintaxis XHTML5 que deberá servirse con sintaxis XML (application/xhtml+xml).12 Esta es la primera vez que HTML y XHTML se han desarrollado en paralelo. La versión definitiva de la quinta revisión del estándar se publicó en octubre de 2014.

Al no ser reconocido en viejas versiones de navegadores por sus nuevas etiquetas, se recomienda al usuario común actualizar su navegador a la versión más nueva, para poder disfrutar de todo el potencial que provee HTML5.

Otros lenguaje relacionado al desarrollo web

CSS/CSS3

Hojas de estilo en cascada (o CSS, siglas en inglés de Cascading Stylesheets) es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado. Es muy usado para establecer el diseño visual de los documentos web, e interfaces de usuario escritas en HTML o XHTML; el lenguaje puede ser aplicado a

cualquier documento XML, incluyendo XHTML, SVG, XUL, RSS, etcétera. También permite aplicar estilos no visuales, como las hojas de estilo auditivas.

Junto con HTML y JavaScript, CSS es una tecnología usada por muchos sitios web para crear páginas visualmente atractivas, interfaces de usuario para aplicaciones web, y GUIs para muchas aplicaciones móviles (como Firefox OS).

JavaScript

JavaScript (abreviado comúnmente JS) es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador webpermitiendo mejoras en la interfaz de usuario y páginas web dinámicas4 aunque existe una forma de JavaScript del lado del servidor (Server-side JavaScript o SSJS). Su uso en aplicaciones externas a la web, por ejemplo en documentos PDF, aplicaciones de escritorio (mayoritariamente widgets) es también significativo.

Desde el 2012, todos los navegadores modernos soportan completamente ECMAScript 5.1, una versión de javascript. Los navegadores más antiguos soportan por lo menos ECMAScript 3. La sexta edición se liberó en julio del 2015.

JavaScript se diseñó con una sintaxis similar a C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo, Java y JavaScript tienen semánticas y propósitos diferentes.

AJAX

AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncronacon el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, mejorando la interactividad, velocidad y usabilidad en las aplicaciones.

Lenguaje del lado del Servidor

Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor web, justo antes de que se envíe la página a través de Internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. Los lenguajes de lado servidor más ampliamente utilizados para el desarrollo de páginas dinámicas son el ASP, JSP, PERL y PHP.

Lenguajes del lado del cliente

La programación con lenguajes en el cliente sirve para muchas cosas, ejemplos de ello son efectos diversos en las páginas, sonidos, videos, menús interactivos, control y respuesta a las acciones de un usuario en la página, control sobre los formularios, etc. Para hacer muchas de estas cosas podemos utilizar diversos lenguajes de programación como Java script VBScript, CSS y HTML o incluso podemos meter aquí programas como Flash entre otros.

Significa que la ejecución de los programas o scripts se realiza en el navegador del usuario. El navegador web se llama también cliente web y se denomina "cliente" porque hace las tareas de solicitud y consumo de servicios. El navegador o cliente web se conecta con un servidor, al que solicita páginas. El servidor web se las sirve para consumo del cliente, que las muestra a la persona que las ha solicitado.

Motor de renderizado

Un motor de renderizado (también conocido como motor de diseño, motor de representación, motor de browser o motor de navegación) es un componente de software básico de todos los principales navegadores web. La función principal de un motor de navegación es transformar los documentos HTML y otros recursos de una página web en una representación visual interactiva en el dispositivo del usuario (esta función se denomina renderización).

El motor de renderizado es software que toma contenido marcado (como HTML, XML, archivos de imágenes, etc.) e información de formateo (como CSS, XSL, etc.) y luego muestra el contenido ya formateado en la pantalla de aplicaciones. El motor "pinta" en el área de contenido de una ventana, la cual es mostrada en un monitor o una impresora. Los motores de renderizado se usan

típicamente en navegadores web, clientes de correo electrónico, u otras aplicaciones que deban mostrar y editar contenidos web.

Todos los navegadores web incluyen necesariamente algún tipo de motor de renderizado. Sin embargo, el término "motor de renderizado" solo alcanzó un uso popular cuando el proyecto Mozilla diseñó el motor de su navegador (Gecko) como un componente aparte del propio navegador. En otras palabras,

el motor de Mozilla era reutilizable por otros navegadores diferentes, y mucha gente se empezó a referir a Gecko como un "motor de renderizado" en sí, en lugar de como una parte del navegador.

El término motor de renderizado también puede referirse a motores de renderizado de texto como Pango o Uniscribe los cuales hacen presentables a los textos plurilingües, teniendo en cuenta los textos bidireccionales, combinaciones de "caracteres básicos" con acentos, y otras complicaciones del texto plurilingüe.

Herramientas de desarrollo

1.- Software WYSIWYG

WYSIWYG es el acrónimo de What You See Is What You Get (en español, "lo que ves es lo que obtienes"). Se aplica a los procesadores de texto y otros editores de texto con formato (como los editores de HTML) que permiten escribir un documento viendo directamente el resultado final, frecuentemente el resultado impreso. Se utiliza en contraposición a otros procesadores de texto, hoy en día poco frecuentes, en los que se escribía sobre una vista que no mostraba el formato del texto, hasta la impresión del documento. En el caso de editores de HTML este concepto se aplica a los que permiten escribir la página sobre una vista preliminar similar a la de un procesador de textos, ocupándose en este caso el programa de generar el código fuente en HTML.
El programa de la izquierda usa un editor WYSIWYG para producir un documento. El programa de la derecha contiene el código LaTeX, que cuando se compile producirá un documento que se verá muy similar al documento de la izquierda. Compilar código de formato no es un proceso WYSIWYG.
Ejemplos de editores HTML tipo WYSIWYG son: Dreamweaver, NVU/Kompozer, las versiones de Composer de Netscape y Mozilla, Amaya, Writer (de OpenOffice.org), Adobe Golive, Frontpage y Microsoft Word. También existen editores que se pueden integrar en formularios de páginas web como FCKeditor, TinyMCE, FreeRichTextEditor.

Otras Alternativas

Trabajar en el frontend de una web significa enfrentarse a varios desafíos. El primero de ellos es conseguir la apariencia visual adecuada, sin perder la vista a la facilidad de uso. Debido a los diferentes dispositivos y navegadores con los que los usuarios acceden a los contenidos web, es necesario que los elementos visuales y técnicos funcionen bien en la totalidad de las plataformas seleccionadas. La accesibilidad, así como un conjunto de buenas prácticas SEO, son tareas que no se pueden pasar por alto a la hora de programar aplicaciones para los usuarios.

Aquellos programadores que no quieren comenzar con el desarrollo de una interfaz web desde cero suelen utilizar frameworksUI o entornos de trabajo de interfaz de usuario. Muchos suelen

decantarse por Bootstrap, una solución desarrollada por Twitter cuyos componentes están optimizados para dispositivos móviles y para los principales navegadores, y con la cual, además, es fácil realizar cualquier idea. Sin embargo, su amplia distribución también se ha encargado de que exista una gran variedad de páginas web creadas con el código estándar de Bootstrap y que, por lo tanto, sean muy similares. Una pequeña crítica es también su complejidad, pues se basa en una gran cantidad de CSS y JavaScript y hace que el marcado de HTML sea mucho más exhaustivo, lo que puede reflejarse, entre otras cosas, en los tiempos de carga de la web.

Si bien es posible descargar el script y los archivos CSS reducidos, de tal forma que solo contengan los componentes Bootstrap necesarios, será necesario ocuparse intensivamente de ello. Aquí, la intención que lleva a muchos a usar un framework, esto es, la simplicidad y el ahorro de tiempo, se pierde un poco. Como consecuencia, a algunos desarrolladores no les parece que este frameworkUI sea siempre la mejor opción y se decantan, por lo tanto, por alguna de las alternativas a Bootstrap, algo que, debido al gran abanico de posibilidades, tampoco resulta una tarea fácil.

Un retrato de las mejores alternativas a Bootstrap

A continuación, presentamos cinco entornos de trabajo frontend que pueden ser utilizados como solución alternativa al framework de Twitter para crear una interfaz web completa y funcional. Además, exponemos sus ventajas y desventajas y sus diferencias con Bootstrap, así como estudiamos la amplitud de las colecciones de código y para qué labor resultan más pertinentes. Por último, analizamos cómo funciona el desarrollo web con cada alternativa.

ZURB Foundation

El framework modular Foundation nació como una guía de estilo producida por la agencia de diseño web ZURB para los proyectos de sus clientes. Posteriormente, ZURB lanzó sus numerosos componentes HTML, CSS y JavaScript como framework de código abierto. Su núcleo, **un GridLayout de doce partes**, facilita el diseño web responsivo, para que se adapte automáticamente a los tamaños de pantalla y las resoluciones de los diferentes dispositivos. Además de este sistema de cuadrículas, Foundation ofrece, entre otras, las siguientes plantillas:

- Sliders
- Botones
- Tipografía
- Media containers
- Listas y barras de menú
- Clases float y visibility integradas

De la misma forma que Bootstrap desde su cuarta versión, Foundation está basado en **Sass**, un lenguaje de estilo simplificado para crear y editar archivos CSS, pero no soporta Less (otro lenguaje de hojas de estilo con el que Bootstrap sí es compatible). Mientras que, a primera vista, las diferencias en cuanto a los plugins JavaScript y a los fragmentos CSS no son evidentes, Foundation ofrece **unnúmero significativamente menor de plantillas** así como un soporte reducido de otras plataformas.

El framework de ZURB supera significativamente a la solución de Twitter en lo que respecta a la individualidad del frontend de cada desarrollador, pues el código CSS subyacente

proporciona acceso a un **diseño compacto** que se puede adaptar fácilmente a las necesidades individuales. A diferencia de Bootstrap, algunas clases son implementadas directamente, ahorrando así tiempo y esfuerzo a la hora de añadirlas. Otra ventaja: ZURB ofrece diferentes cursos **y apoyo personal para tu proyecto de frontend** así como una versión especial del framework para diseñar newsletters.

Si deseas utilizar Foundation para el desarrollo de una interfaz web, puedes utilizar el juego completo de elementos o decantarte por un pack gratuito personalizado con los componentes necesarios en su página web oficial.

Pure.CSS

A mediados de 2013, Yahoo lanza Pure.CSS, una estructura básica para el desarrollo de interfaces web que es considerada como una alternativa a Bootstrap de gran calidad, pero que también se puede utilizar en conjunto con este framework. Pure se basa en la arquitectura modular y escalable para CSS SMACSS (Scalable and Modular Architecture for CSS), que se asegura de separar aquellos elementos repetitivos (tablas, botones o formularios) del diseño básico (tipo de letra, diseño, etc.) y les permite tener convenciones propias. El **framework de Yahoo** asigna a cada módulo (independientemente de si se trata del diseño regular o de reglas específicas) un nombre de clase estándar con el prefijo “pure” y los interpreta como submódulos. Así, por ejemplo, para insertar un formulario en el que los elementos se encuentran uno debajo del otro, se especificará, por defecto, la clase “pure-form” y la clase “pure-form-stacked” para denotar a la subclase.

Este framework para frontend, que puede descargarse en una versión responsiva y en una no responsiva, contiene seis módulos que, en su forma comprimida, tienen **un tamaño de 4 GB** (descomprimidos, de 16 KB):

- **Base** (base-min.css): base del framework, incluyendo su conjunto de normas
- **Grids** (grids-responsive-min.css): sistema grid flexible y responsivo
- **Forms** (forms-min.css/forms-nr-min.css): formularios
- **Buttons** (buttons-min.css): diferentes botones
- **Tables** (tables-min.css): tablas
- **Menus** (menus-min.css/menus-nr-min.css): menús

Todos estos módulos se basan, de la misma forma que los componentes de Bootstrap y de otros frameworks CSS, en las hojas de estilo de código abierto Normalize.css, que reemplaza el estilo estándar de los elementos HTML con **estilos optimizados para los diferentes navegadores**. En comparación con el framework de frontend de Twitter, la solución de Yahoo no contiene todas las aplicaciones JavaScript, aunque, como sucede a menudo, se pueden implementar manualmente en todo momento. Cabe resaltar que Pure.CSS no es tanto un frontend ya terminado que solo requiere ser ajustado a las necesidades personales, sino, más bien, **el punto de partida para un proyecto** y, por lo tanto, asociado a un grado significativamente mayor de libertad en lo que respecta al diseño.

Yahoo aloja a Pure.CSS en su propia Content Delivery Network, (Yahoo CDN) de tal forma que, para integrar tu proyecto, solo necesitas hacer una simple referencia en el encabezado (<head>). Lógicamente también es posible descargar Pure.CSS y alojarlo por tu cuenta. El enlace actual para la CDN y para la descarga de los archivos lo encuentras en su web oficial purecss.io.

Semantic UI

En 2013, el programador Jack Lukic publica un framework como solución para el desarrollo frontend bajo el nombre de Semantic UI. La gran ventaja de esta colección de código radica

en la manera como busca simplificar la escritura de código HTML a través de convenciones intuitivas y fáciles de usar. Para este propósito, Semantic UI dispone de **más de 3.000 clases CSS** que tienen como finalidad optimizar el proceso de desarrollo. Mientras que Bootstrap solo contiene un tema en el pack básico, Semantic UI cuenta, en su versión estándar, con **más de 20 plantillas prediseñadas**. Sin embargo, el diseño con este framework puede resultar un poco más complicado que con Bootstrap: además de los archivos CSS y JavaScript básicos, el pack estándar contiene fuentes, el gestor de paquetes PHP Composer, el administrador de archivos web Bower y el automatizador de tareas Gulp.

Los componentes individuales están distribuidos en las siguientes seis áreas:

- **Globals**: definiciones de estilo sobre la base de Normalize.css; tipografía y diseño básico
- **Elements**: componentes frontend como botones, iconos, contenedores y muchos más
- **Collections**: contenidos estructurados como la rejilla, los menús, las tablas o los formularios
- **Views**: elementos interactivos como campos para comentarios, feeds de noticias o banners publiciarios
- **Modules**: widgets tales como menús desplegables, ventajas emergentes o casillas de verificación
- **Behaviors**: interfaces de programación para JavaScript

Para principiantes y usuarios con conocimientos básicos, el sistema de nombres de Semantic UI puede ser inicialmente desconcertante y, en cada caso, estar **ligado a una cierta práctica**. Al final, no obstante, este esfuerzo vale la pena, debido a que leer el código HTML de su interfaz es mucho más intuitivo que en otros frameworks como Twitter Bootstrap, algo que resulta muy útil especialmente durante las revisiones posteriores. Semantic UI está disponible en CSS y Less; además, ahora también existe **una variante Sass**, un punto que este framework semántico comparte con Bootstrap. Una gran desventaja de esta alternativa a Bootstrap es el hecho de que muchos de sus componentes JavaScript son dependientes y no funcionan sin el lenguaje de script.

Puedes descargar Semantic UI desde su página oficial, enlazar los archivos en la red de entrega de contenidos JSDELIVR o utilizar los fragmentos de código disponibles en el repositorio del framework en GitHub.

UIkit

UIkit es la **solución de código abierto** para programación frontend desarrollada por la empresa de Hamburgo YOOtheme, con una gran experiencia en el desarrollo de aplicaciones web, así como de temas para WordPress, Joomla y su propio website builder YOOtheme Pro. La extensa colección de componentes HTML, CSS y JavaScript se encuentra disponible bajo la licencia libre MIT y, por lo tanto, se puede utilizar y modificar sin ningún problema. Todos los fragmentos CSS **están disponibles en una variante Less y en una Sass**. Los más de 30 módulos de esta alternativa a Bootstrap, como en el caso de sus principales competidores, están basados en **Normalize.css**, por lo que casi ningún navegador web tiene problemas para visualizar proyectos creados con UIkit.

Los componentes principales se dividen en las siguientes seis categorías:

- **Defaults**: base para la normalización de los elementos HTML, gracias a la cual se logra la capacidad multinavegador y las máximas de estilo básico

- **Layout**: herramientas para el diseño del frontend, por ejemplo, el sistema de cuadrícula, las cajas de contenido o clases CSS útiles para contenidos que se repiten
- **Navigations**: todos los elementos que facilitan que el usuario explore la página web, incluyendo, entre otras cosas, módulos para la paginación (numeración de páginas) o las clásicas barras de navegación
- **Elements**: estilos para bloques de contenido autónomos como tablas, listas y formularios
- **Common**: componentes que se utilizan normalmente dentro del contenido, por ejemplo, botones, iconos, insignias o animaciones
- **JavaScript**: módulos compuestos principalmente por JavaScript para implementar elementos interactivos

Para preparar contenidos para los diferentes tamaños de pantalla, UIkit dispone de varias clases responsivas. Con la ayuda de la herramienta online **Customizer** puedes ajustar elementos predefinidos de gran importancia tales como, por ejemplo, 1.200 píxeles para pantallas de gran tamaño o 479 píxeles para smartphones con pantallas pequeñas. Para evitar complicaciones con otros snippets del framework o de CSS, todas sus clases se definen con el prefijo "uk". Aquellos elementos JavaScript y CSS necesarios para **la creación de interfaces web más complejas** que no estén integrados en el núcleo del framework pueden serlo en cualquier momento, por ejemplo, cuando una web requiere un **área de administración** que incluye funciones de inicio de sesión, editor HTML y de carga de archivos.

A pesar de su impresionante funcionalidad, el tamaño de archivo de los componentes individuales, así como del framework completo, es sorprendentemente bajo. Esto se complementa con una **extensa documentación** que simplifica en gran medida el trabajo inicial con UIkit, característica por la que también sobresale Bootstrap, así como por la gran selección de temas y oferta de tutoriales. Adicionalmente, existen alrededor de 1.500 bifurcaciones (forks) en GitHub, donde es posible encontrar y descargar todos los módulos de este framework de frontend. Aunque el pack completo también está disponible en su página principal, este entorno de trabajo no permite la instalación de módulos de forma individual, con lo que aquellos que no se necesiten, tendrán que ser eliminados posteriormente.

Materialize

Materialize es un framework CSS que se basa en los principios básicos del material design, un estilo de diseño implementado por Google en 2015 y que ahora es utilizado en la mayoría de sus aplicaciones. Este concepto de diseño se basa en superficies gráficamente similares a tarjetas en un estilo minimalista (flat design), pero que a la vez **implementan muchas animaciones y sombras**. Los efectos de profundidad producidos con esta técnica permiten que los usuarios capturen elementos de información e interacción con más facilidad. El desarrollo de este entorno de trabajo con licencia MIT estuvo en manos de cuatro estudiantes de la universidad Carnegie Mellon en Pensilvania (Alvin Wang, Alan Chang, Alex Mark y Kevin Louie).

Esta alternativa a Bootstrap, que, como el framework de Twitter, tiene un sistema de rejilla de 12 columnas, contiene componentes CSS y JavaScript con más de 700 símbolos de material design en una Iconfont y en Roboto, la fuente predeterminada del concepto de diseño propio de Google. Además de los archivos CSS habituales en versión normal y reducida, también se puede, como con Bootstrap, aprovechar los archivos de origen SCSS escritos en Sass para facilitar el proceso de personalización de tu interfaz web. Independientemente de tu elección, tendrás a tu disposición 30 elementos diferentes:

- **CSS:** la función primaria de CSS es, como en Bootstrap y otros frameworks UI, una rejilla responsiva, que proporciona las bases para que una interfaz web funcione en todos los dispositivos. El grid de Materialized contiene **tres tamaños predefinidos** de visualización: menos de 600 píxeles para dispositivos móviles, hasta 992 píxeles para tabletas y alrededor de 992 píxeles para los ordenadores de escritorio. Otros snippets de CSS incluyen una paleta de colores basada en los colores primarios del material design, la ya mencionada fuente Roboto, así como diferentes clases que se denominan "helpers" y son un apoyo a la hora de alinear el contenido.

- **Components:** los componentes son las partes esenciales del framework de frontend necesarios para la realización de elementos de navegación y de áreas interactivas. Además de los componentes tradicionales como códigos para insertar paginación, formularios, barras de navegación o iconos, también encontrarás módulos de importancia fundamental **para la aplicación del concepto del material design**. Estos incluyen, por ejemplo, las "Cards" (las típicas colecciones de objetos de Google para la presentación de contenidos) o los "Chips" simbólicos que permiten representar etiquetas o contactos.

- **JavaScript:** en lo que respecta a las aplicaciones JavaScript, Materialize es, sin duda, una de las mejores alternativas a Bootstrap. Independientemente de si deseas mostrar tus imágenes en **vista carrusel**, añadir **cuadros de diálogo interactivos** o darle vida a la interfaz con el **efecto parallax**, siempre tendrás a tu disposición elementos JavaScript para lograr dicho fin. De esta forma, los usuarios del framework estarán equipados de la mejor manera para diseñar una interfaz web que, además, ofrece una gran usabilidad tanto en dispositivos móviles como en ordenadores de escritorio.

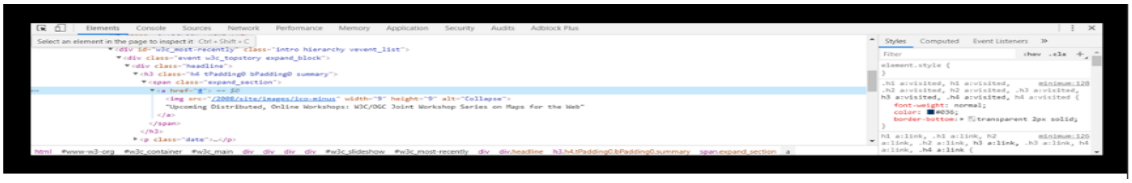
Para descargar los archivos del proyecto puedes visitar GitHub o su página web oficial materializecss.com. Allí también encontrarás las dos plantillas disponibles: "starter" y "parallax", así como el código de marcado para acceder a los archivos del proyecto a través de la **red de distribución de contenidos cdnjs**.

Inspección de código desde el navegador

La herramienta de inspección de código del navegador se encuentra integrada en cualquier navegador. Se deberá utilizar cuando con el resto de herramientas no se puede obtener la información deseada.

Permite:

- Ver el código HTML de algún elemento de la web.



¿Cómo usarla?

Localizar en la web el elemento que se quiere analizar, pulsar sobre él con el botón derecho y elegir la opción "Inspeccionar" (en chrome) / "Inspeccionar elemento" (en firefox).

Se abre a continuación una nueva zona en la pantalla (generalmente en la parte inferior pero se puede mover a la posición deseada), que muestra el código fuente del elemento seleccionado previamente. Así se podrá conocer si tiene determinados atributos, cómo se estructura dicho elemento respecto a sus elementos próximos, etc.

La herramienta tiene un aspecto ligeramente diferente en Chrome y Firefox:

- En Chrome, la pestaña que debe tenerse activa es la nombrada "Elements". No aparece directamente la opción de buscar, pero se puede acceder a ella pulsando Control + F. Se abrirá entonces un cuadro de búsqueda en la parte inferior que permitirá buscar dentro del código.
- En Firefox, la pestaña que debe tenerse activa es la nombrada "Inspector". Por defecto aparece debajo de "Inspector" un cuadro de búsqueda, que se usará para buscar dentro del código.

Servidores WEB

Con "Servidor web" podemos referirnos a hardware o software, o a ambos trabajando juntos. En cuanto a hardware, un servidor web es una computadora que almacena el software de servidor web, y los archivos que componen un sitio web (por ejemplo, documentos HTML, imágenes, hojas de estilos CSS y archivos JavaScript).

Tipo de Servidores:

Los servidores pueden ser Estático y Dinámico

Un **servidor web estático**, o pila, consiste en una computadora (hardware) con un servidor HTTP (software). Llamamos a "estático" a este servidor porque envía los archivos que aloja "tal como se encuentran" (sin modificarlos) a tu navegador.

Un **servidor web dinámico** consiste en un servidor web estático con software adicional, habitualmente una *aplicación servidor* y una *base de datos*. Llamamos "dinámico" a este servidor porque la *aplicación servidor* actualiza los archivos alojados, antes de enviar el contenido a tu navegador mediante el *servidor HTTP*.

Puntos básicos en la elaboración de una página Web.

1.- Define el objetivo de tu página:

El objetivo de tu sitio es la base de la estructura y por esto es muy importante que lo definas primero. Algunos objetivos para una página web pueden ser:
Tener una carta de presentación de tu empresa para poderla mostrar a clientes actuales.
Tener un sitio web para atraer visitantes y mediante un proceso transformarlos en clientes nuevos para tu negocio.
Una tienda en línea en la cual ofrezcas tus productos, servicios o contenidos a los usuarios de manera directa.
Un blog o portal de noticias el cual se convierte en un negocio de información más que de venta.
Una página para atraer visitantes a un evento.
Una combinación entre dos o tres de los incisos anteriores.
No importa cuál sea tu objetivo pero es importante tenerlo claro desde el principio ya que de esta forma podrás llegar a diseñar un sitio que sea efectivo
para su causa. Sin embargo siempre ten en cuenta que debe ser escalable ya que debe de poder crecer a la par de tu empresa.

2. Elige el estilo:

Es muy posible que sí estás en el punto de crear o rediseñar un sitio web es porque:
Tienes un negocio establecido que ya lleva un tiempo pero no tienes página web.
Eres un negocio establecido que lleva un tiempo y tienes una página web anticuada o desactualizada.
Eres un negocio nuevo que apenas comienza con todo lo que requiere para establecerse.
Cualquiera que sea el caso es importante que se tenga ya una base de diseño en cuanto a tu logotipo y aplicaciones, ya que esto servirá como la base para dictar parte del estilo que elegirás.

Hay distintos estilos en las páginas web y siempre es muy conveniente que revises varias páginas para tener una idea de distintos ejemplos que existen y cuáles son los que más te gustan.
Algunos de los estilos que puede tener una página son:

1. Una "One-Page".
2. "One-Page" con un carrusel de fotos de inicio.
3. Un sitio web con distintas secciones.
4. Una tienda en línea.
5. Un sitio de aterrizaje.

Una vez que tengas definido el estilo que más te gusta y el objetivo puedes pasar al siguiente paso.

3. Crea un mapa de sitio:

Crear un mapa de sitio es otro de los pasos básicos de una página web, esto no es más que determinar cuáles serán las secciones que deberá incluir y tu página y a su vez que subsecciones existirán dentro de estas. Lo más práctico para esto es tomar un pedazo de papel y una pluma y empezar a crear una lista. Una vez que se tenga la lista se podrá empezar a trabajar en que deberá incluir cada página y sub-página.

4. Elige las Palabras clave:

No importa si tu objetivo es únicamente tener una plataforma que puedan visitar tus clientes actuales o si buscas una plataforma para captar nuevos clientes, cualquiera que sea el caso es importante que cuando estas personas entren a un buscador y tecleen ciertas palabras relacionadas con tu empresa ¡aparezca tu página! Y aunque esto puede ser un reto, si ya estas tomando la decisión de crear un sitio web, es mejor que sea efectivo y este optimizado desde el principio.

Crear una lista de palabras clave debe estar centrado en como tus visitantes o clientes buscarían los productos/servicios de tu empresa, y con base en esto seleccionar las palabras más significativas para sobre eso desarrollar el contenido de la página.

5. Estructura la información y crea los contenidos:

Una vez que ya tienes claro cómo te buscan tus posibles clientes es importante que empieces a crear la información de cada sección de tu página, utiliza las palabras clave de manera natural, un lenguaje que usen tus visitantes y la descripción precisa de cada parte del negocio. Un tip que te puede servir bastante es que primero trabajes en todas las secciones de tu página y dejes la información de la página de inicio al final (esta deberá ser un resumen concreto de toda tu página).

6. Elige las imágenes y crea los gráficos necesarios:

Es hora de vestir tu página e ilustrar los puntos más importantes, selecciona una serie de imágenes adecuadas para cada sección de tu página. Estas imágenes pueden ser algunas que tengas de tu empresa y/o personal, de algunos de los trabajos que has realizado, de algunos de tus clientes, y si simplemente no cuentas con imágenes puedes utilizar algún stock de fotos para descargarlas de ahí.

También es importante que definas si requieres alguna tabla, gráfica o elemento visual para explicar mejor algún punto, si es el caso deberás crearlo o contratar a un diseñador que te genere estos elementos.

7. Agrega un blog a tu página:

Tal vez tu objetivo no vaya más allá de una simple página informativa o corporativa, sin embargo un blog dentro de tu página web ofrece bastantes beneficios ya que no solo te ayuda en el posicionamiento en buscadores como Google, sino que también se vuelve en una plataforma mediante la cual puedes ofrecer información valiosa a tus visitantes que les ayudarán a tomar una decisión de compra acerca de tus productos o servicios. También un blog te puede ayudar a responder las preguntas frecuentes de tus posibles clientes en diversos temas relacionados con tu industria.

8. Da difusión a tu nuevo sitio web:

Una vez que tengas listo el sitio debes hacer que la gente lo sepa para que lo empiecen a visitar, algunas formas de hacerlo son:
Incluirlo en la papelería de tu empresa (tarjetas de presentación, hojas membretadas, sobres, folletos, etc.).
Notificar a tus contactos actuales vía correo electrónico o mediante un emailing. Incluirlo en la firma electrónica de tu correo.
Hacer publicidad a través de las redes sociales.

Jerarquía de directorio de un proyecto web.

Para hacer un proyecto web debemos organizar los directorios que van a contener los documentos o archivos que vamos a utilizar. Debajo aparece el árbol de directorio que debemos crear.
 Debe crear una carpeta principal
 Debe crear varias carpetas:
1. Paginas
a) Documentos HTML
b) Documento CSS
c) Documento de JavaScript
2. Imágenes
3. Videos
4. Sonidos

Ordenado de esa manera nos será más fácil manipular los diferentes archivos que vamos a manejar en el proyecto.

Elementos de estructura para el diseño de una página Web:

Cuando hablamos de las partes de una página web, normalmente nos referimos a su estructura. O lo que es lo mismo, a todos los elementos o funcionalidades que la

componen como, por ejemplo, las secciones o menús, que hacen que se pueda navegar por ella y sea funcional.

Aunque menos habitual, también es posible referirnos a las partes de una página web para hablar de su contenido, es decir, las diferentes páginas que se albergan dentro de un sitio web, por ejemplo, la página de contacto, de producto o la propia home o página principal.

Lo que sí está claro es que existen cientos, por no decir miles, tipos de páginas web. Por tanto, te podrás imaginar que las partes de una página web no son las mismas en un ecommerce que en una página web corporativa o un blog de repostería, aunque sí es cierto que comparten una estructura y muchos elementos en común.

¿Quieres ver de cuáles son las partes de una página web y cuáles son los elementos más comunes? Pues entremos en materia.

, nos referimos a su estructura. Eso sí, muchas veces también se habla de partes de una web en función de su contenido. Así que, teniendo esto en cuenta, distingamos entre:

1. Partes de una página web en función de su estructura (estructura web) Cabecera o header
2. Cuerpo o body
3. Pie de página o footer
4. Partes de una página web en función de la distribución del contenido
5. Inicio o home
6. Contacto
7. Productos y servicios
8. Blog
9. Política de privacidad

Estructura de una página web

En la estructura de una página web tenemos tres partes diferenciadas: cabecera, cuerpo y pie de página. O, por sus términos en inglés: header, body y footer.

Cabecera o header

La cabecera es la parte superior de la página web. Aquí suele incluirse la información básica de la empresa o marca y es consistente en todo el sitio, es decir, se repite en cada página de la web por la que navegas.

Los elementos que se incluyen en la cabecera son:

- Logo de la empresa
- El menú de navegación

- Un cuadro de búsqueda
- Una pequeña descripción de la web

Estos elementos muchas veces difieren en función de la plantilla de WordPress que utilices, pero son los más comunes.

Si hablamos en lenguaje HTML, verás que todo el contenido de la cabecera aparece entre las siguientes etiquetas:

<header>Contenido de la cabecera</header>

Nota: Cuando creas una página web, es recomendable fijar la cabecera. ¿Qué quiere decir esto? Pues que a medida que un usuario hace scroll y navega a través de tu sitio, la cabecera se mantenga fija y tu marca y los accesos más importantes de tu página, como el menú, estén siempre visibles y accesibles.

Cuerpo o body

Esta es la parte de la web que alberga el contenido principal de tu página. Esta parte sí es diferente en cada página de tu sitio. Es decir, no será el mismo contenido el que incluyes en tu página de contacto que en la página principal.

En HTML puedes identificar rápidamente este contenido, ya que se encuentra dentro de las siguientes etiquetas:

<body>Contenido del cuerpo</body>

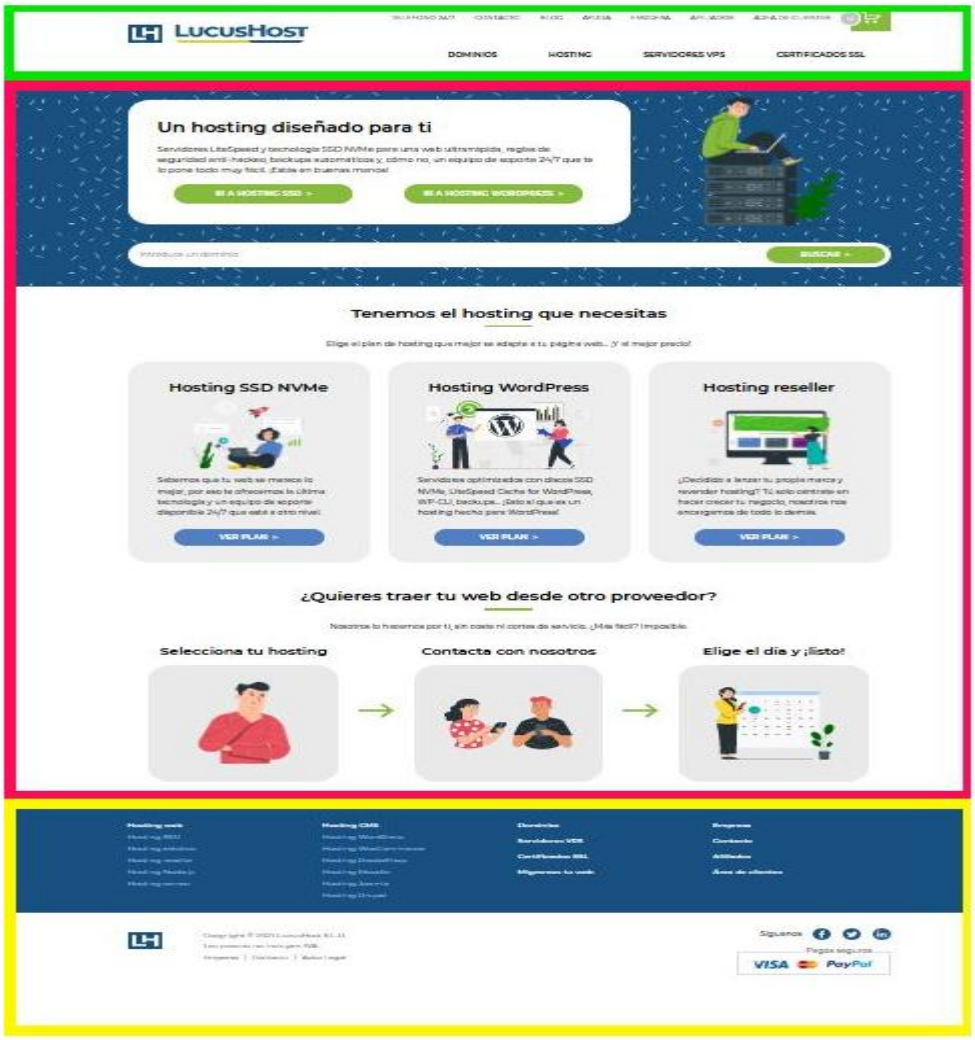
Pie de página o footer

El pie de página o footer es la parte inferior de una página web. Al igual que la cabecera, se repite y es consistente en cada página. En el footer podemos encontrar los siguientes elementos:

- Menú simplificado
- Información de contacto
- Botones de redes sociales
- Logo de la empresa
- Enlace a la política de privacidad

Tal y como vimos en las partes anteriores, en HTML el pie de página se encuentra entre estas etiquetas:

<footer>El contenido del pie de página</footer>



Su estructura sería la siguiente:

Aquí tenemos la cabecera o header. Como ves, en ella incluimos el logotipo de la marca y el menú de navegación. Esta parte se mantiene siempre estática, aunque estés navegando por otras páginas o hagas scroll.

Color magenta: sería el cuerpo o body. Esta información es diferente en cada página del sitio web.

Color amarillo: es el footer de la web. Siempre se encuentra al final de cada página de la web y el contenido es siempre el mismo. Nosotros incluimos el logo, información de contacto, métodos de pago disponibles y también los iconos de redes sociales.

Partes de una página web en función de su contenido

Acabamos de ver las partes de una página web en función de su estructura, ahora veamos cómo se divide una página web en función de la distribución de su contenido.

Inicio o home

Si hablamos de las partes de una web, la página principal, de inicio o la home es la más importante de todas. Aquí se incluye la información más relevante de tu empresa, ya que es en la que tienes que proyectar tu marca y mostrar a tus visitas a qué te dedicas, en qué te diferencias de tu competencia y por qué deberían elegirte a ti.

Además de los elementos propios de la cabecera y el footer (que ya sabes que son iguales en todas las páginas), en la página de inicio sueles encontrar:

CTA o llamada a la acción: imprescindibles para lograr que tus visitas realicen la acción que tú sugieres: comprar/suscribirse…

Resumen de tus productos y servicios: generalmente se suele hacer mención a tus productos mediante una buena imagen o un eslogan, con el objetivo de que tus visitas en solo 3 segundos sepan a qué te dedicas. Sin entrar en mucho detalle, para eso tendrás tu propia página.

Ofertas: si tienes una oferta importante y que quieres destacar, la página de inicio suele ser la parte de la web más adecuada para anunciarla.

Indicadores de éxito: número de clientes, número de empleados, países en los que operas, años de experiencia… Todo vale para mostrar que tú eres el mejor.

Testimonios o colaboraciones: Es muy habitual encontrar también en página principal algún testimonio, reseñas de Google o los logos de las empresas para las que has trabajado.

La página de inicio está alojada siempre en el dominio principal, no en un directorio. Para que me entiendas, la página de inicio o hombre de tu web será tudominio.com y no tudominio.com/servicios.

Contacto

Una página web no sirve únicamente para vender online.

Si tienes un negocio pequeño como, por ejemplo, una peluquería, tener una página web te ayudará a ganar visibilidad en tu zona y mostrar a tus clientes potenciales la forma de contactar contigo.

El teléfono, la dirección y los horarios son tres elementos básicos que deben estar sí o sí en todas las páginas de contacto y, como no, también en la cabecera y en el footer. Pero, además de ello es muy recomendable incluir los siguientes elementos:

Formulario de contacto: Darás a tus clientes la posibilidad de contactar contigo las 24 horas del día y esto transmite muy buena imagen. ¿Quieres crear el tuyo? En este post tienes una recopilación con los mejores plugins de formulario para WordPress. ¡Toma nota!

Dirección de correo profesional: Aunque cada vez es algo más habitual, tampoco es nada raro encontrarte con emails del tipo carpinteriafede@gmail.com. Así que, crea las direcciones de correo personalizadas con tu dominio y transmite que eres todo un profesional.

Productos y servicios

Cuando creas el contenido de una web, lo ideal es que la información más detallada de los productos que vendes o los servicios que ofreces la incluyas en una página específica.

En esta parte de la página web debes incluir todo lo relacionado con tus productos o servicios: ventajas de tus productos, principales características, a quién están dirigidos, el precio, gastos de envío…

Por supuesto, no te olvides de la importancia de una buena llamada a la acción en esta parte de tu web. Textos como «¡Lo quiero!», «Añadir a la cesta» o «Reserva ya» ayudarán a conducir a tus visitas hacia el objetivo final.

Blog

Si hay una parte fundamental en cualquier tipo de página web, esa es la sección del blog.

Crear un blog puede hacer que tu negocio despegue de una vez por todas. Es indispensable para tu estrategia de contenidos y una pieza clave si quieres mejorar tu posicionamiento orgánico en Google y ganar visitas a tu sitio.

Los elementos web más comunes que encontrarás en casi cualquier blog son:

Buscador: No todas las plantillas de una página web incluyen la función de buscador, pero las plantillas de blog casi todas lo tienen.

Formulario de suscripción: Incluir una caja de suscripción en un blog es una forma de generar leads de calidad. Aunque no sean clientes, ya tienes su cuenta de correo electrónico para enviarle el contenido que publicas y, poco a poco, puedes ir acercándolos al proceso de compra.

Categorías y etiquetas: Otro de los elementos que es recomendable incluir en esta parte de tu página web es una sección con las categorías y etiquetas de los contenidos de tu blog. Es una buena forma de estructurar el contenido y mejorar la usabilidad de tu sitio.

Política de privacidad

Seguramente que es una de las páginas menos vistas de una web, pero con el GDPR (y ya antes de su aprobación) uno de los requisitos básicos es proporcionar a tus usuarios toda la información relacionada con el procesamiento de los datos que se recopilan a través de tu sitio web.

Esta parte de tu web no tiene que ir en el menú principal, pero la tienes que incluir sí o sí. Generalmente se ubica en el pie de página o footer, así como en los avisos de cookies o cuando incluyes algún formulario de registro o suscripción.

Estas son solo algunas de las páginas más importantes o comunes que puedes localizar en una web, pero no las únicas. En función del contenido que necesites mostrar a tus usuarios, necesitarás crear o suprimir algunas. ¡Sobre esto no hay una regla escrita!

Concepto. Elementos de estructura para el diseño de una página Web:

La estructura web, es la forma en la que está organizada una web mediante su enlazado interno, y mediante la cual, reparte su autoridad a sus páginas internas.

La estructura interna de los enlaces de una web es muy importante, ya que, la página con más fuerza de nuestra web (normalmente la home), repartirá su fuerza y autoridad de enlaces externos al resto de páginas.

Linkjuice: traspasa fuerza a tus otras páginas

El Linkjuice es la fuerza que pasamos de una página a otra mediante enlaces. La fuerza de una página se divide en el número de enlaces salientes, y a cada una de ellas, se les deriva un valor de Linkjuice.

Veamos en profundidad cómo se reparte la fuerza de una página y se reparte entre las páginas enlazadas.

¿Cómo se reparte la autoridad en la estructura de una web?

En cada página, la autoridad se divide en función de los enlaces salientes y da como resultado esta fuerza de enlazado denominada Linkjuice. Los enlaces de una web producen cadenas de enlaces y cada página enlaza a otras, es por ello que cada vez es menor la fuerza que pasa de una página a otra, es decir:

Home -> Página 1 -> Página 3
-> Página 4
-> Página 2

La home pasará un 50% de su fuerza (conocida como Linkjuice) a la página 1 y a la página 2, mientras que la página 1, pasará un 50% de su Linkjuice, equivalente a un 25% del Linkjuice de la home. El gráfico quedaría algo así en porcentajes de Linkjuice (fuerza de enlazado):

100% -> 50% -> 25%
->25%
->50%

Realmente el cálculo de la fuerza de enlazado no es exáctamente así, esto solo es un ejemplo para que se entienda que por cada enlace, la fuerza que traspasa es mejor. Actualmente no se conoce cómo se calcula la fuerza de enlazado, aunque sí se conoce la fórmula empleada inicialmente por Google siendo esta muy compleja.

Es por este motivo, que siempre se recomienda que una web permita llegar a cualquier página mediante 3 clics, ya no solo por usabilidad y comodidad para el usuario, también para facilitar el crawleo de nuestra web.

Cómo influyen los enlaces nofollow en una web

Los enlaces nofollow de nuestra web, hacen que la fuerza de enlace se pierda y no vaya a ningún sitio, es por ello que siempre es preferible no tener enlaces en nofollow:

Datos a destacar de los enlaces a una página

Además de tener conocimientos sobre cómo se reparte la fuerza entre los enlaces de tu web y de conocer cómo afectan los enlaces nofollow a la tu web, es importante tener en cuenta que, cuando se tienen varios enlaces hacia una misma página dentro de una página concreta, solo da fuerza uno de los enlaces, el resto que están repetidos, no aportaran Linkjuice. Ejemplo gráfico:

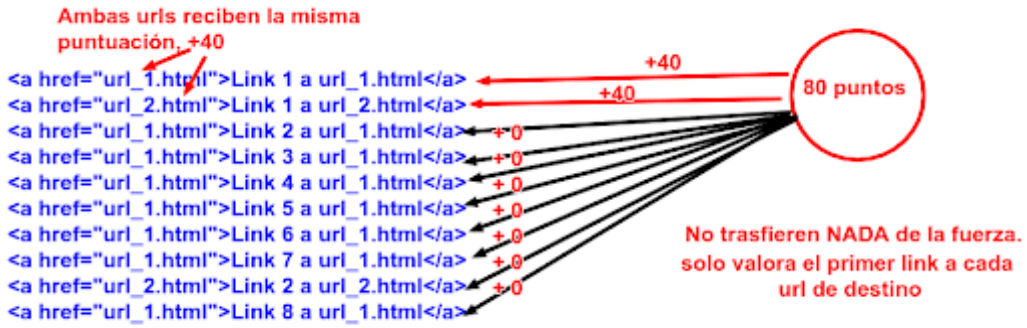


Imagen de mecagoenlos.com

¿Qué herramientas y elementos de código usamos para controlar la estructura de una web?

Como es evidente, no siempre nos va a interesar todas las urls de una página web, es por ello que existen diferentes utilidades y herramientas para ayudarnos a decidir, además de impedir, que estas páginas se indexen.
Las herramientas presentadas a continuación, no solo nos ayudarán a cómo ver la estructura de una página web, si no también nos ayudarán a tomar las decisión más correcta en cuanto qué hacer con cada url o cada conjunto de urls, algo muy importante en la arquitectura web seo.

Robots

Algo que no debe faltar es la configuración de los robots de Google. Estos robots se pueden controlar principalmente de dos formas:
Robot.txt o configuración de htaccess
Meta etiqueta robots
Controlar los robots ayudará a que el robot únicamente pase e indexe las páginas que nos interesan.
Es importante recordar que, si no nos interesa que Google detecte una url, siempre será mucho mejor eliminarla, siempre y cuando sea posible.
NOTA: La página de carrito o el checkout de una tienda online, no nos interesa posicionarla, pero tampoco eliminarla.
Una url que abre una ventana y es una imagen más grande, esto nos interesa posicionarlo y podría llegar a intentar eliminarlo.
Herramientas de crawleo

Para conocer bien qué páginas nos interesa posicionar y cuáles no, previamente hay que hacer o un estudio de palabras clave, definir las páginas que se van a necesitar para el proyecto y cómo se van a enlazar entre estas, o en caso de ya estar creada la web, revisar esta estructura web mediante herramientas de crawleo, es decir, bots:

1. Screaming Frog
2. Sitebulb
3. Oncrawl
4. Seotools for excel
5. Netpeak Spider

Herramientas de seguimiento y analítica

Existen también herramientas de analítica que pueden ayudarnos a detectar algunas páginas que pueden estar indexándose o por las cuáles los robots de Google están pasando y no están posicionando, como:

Google Analytics: Filtrando con tráfico orgánico podemos ver las urls que tienen visitas de buscadores. También pueden usarse otras herramientas de analítica para esto.

Search console: La versión nueva proporciona mucha más información de esto, ya que nos muestra páginas que están en noindex y el robot a rastreado, páginas que están sin posicionar, etc.

Herramienta de estructura de enlaces

Las siguientes herramientas nos facilitan ver de forma gráfica cómo están construidos los enlaces y cómo se enlazan las páginas entre ellas.

1. Cognitive SEO
2. Fandangoseo
3. Screaming Frog
4. Sitebulb

Herramientas para ver páginas más enlazadas internamente y su fuerza

Ahrefs: información de las páginas más enlazadas y más fuerza de enlace

Excel: importando datos y aplicando fórmulas puedes hacer tus propios cálculos automáticamente e ir probando a modificar enlaces para ver cómo se repartiría esta autoridad, aunque es bastante complejo.

Open site explorer: nos da información sobre la fuerza de cada página

Netpeak Spider y Safecont: son dos herramientas que valora la fuerza de cada una de las páginas en función de los enlaces internos. Screaming Frog también puede hacer esto desde la versión 10

Herramientas para ver si ranquean urls en Google

Esto puede sernos útil para detectar si existen tipologías de urls que están posicionando en Google o están indexadas pero apenas nos aportan tráfico.

Por tipologías de url, nos referimos a urls que pertenecen a un conjunto de urls dentro de una web, como artículos de una categoría, productos de una categoría, etc.

En este caso, es conveniente ver si pueden aprovecharse y posicionarse, o nos interesa más eliminar todas estas urls, para mejorar la frecuencia de rastreo sobre otras páginas que pueden mejorar la posición media de sus palabras clave, y aportarnos más tráfico en consecuencia.

Para esto, es necesario revisar las palabras clave por las cuales posicionan o deberían posicionar y calcular aproximadamente, cuánto tráfico podría traernos en base al promedio de búsquedas. Existen varias herramientas que realizan esto, entre ellas:

1. Search Console
2. Google analytics
3. Semrush
4. Ahrefs
5. Sistrix
6. Cognitiveseo

Por ejemplo, si tenemos más de 20 páginas que no posicionan y tienen una misma tipología, este conjunto no interesará posicionarlo seguramente o tocará revisar que está sucediendo y si realmente nos interesa posicionar estas páginas.

Herramientas de cruce de datos

Las herramientas de cruce de datos nos facilita mucho la toma de decisiones, por lo que puede ser útil si contamos con varias herramientas y queremos analizar los enlaces entrantes y la valoración de estos enlaces, así como otros datos.

URL profiler

Esta herramienta nos permite conectarnos vía API y a muchas otras herramientas y exportar toda la información ya cruzada en un excel.

Excel

Es la herramienta por excelencia que casi todo el mundo usa para cruzar datos. Por 3 motivos: es fácil de manejar, cuenta con mucho soporte en la red y no es necesario programar en una gran mayoría de casos.

También existen complementos que ayudan a realizar las tareas de importar la información. Con un par de fórmulas, puedes unificar la información extraída de varias fuentes y compararla.

Herramientas de análisis de logs

Las herramientas de análisis de logs nos indican la navegación de los diferentes robots y usuarios sobre nuestra web, además de obtener datos sobre posibles errores 400, 500 o redirecciones.

1. Excel
2. Screaming frog log analyzer
3. Oncrawl

Existen más herramientas, pero la más asequible en relación calidad/precio sería Screaming Frog Log Analyzer.

La posibilidad de ver las urls más rastreadas te permitirá tener información, hacerte preguntas y contrastar la información: ¿Cuál es la url más rastreada por los robots de búsqueda?, ¿Realmente

queremos que sea la más rastreada?, ¿Está posicionando esa url?, ¿Cuánto tráfico nos aporta?, ¿Los usuarios que llegan a esa página realizan el objetivo que queremos?….

Respondiendo a estas preguntas, en base a la información extraída de los logs, podemos replantearnos la estructura de nuestra web y los enlaces internos.

Tipos de arquitectura web

Existen varios tipos de arquitectura web con los que la mayoría de expertos trabajan. Suelen ser bastante fáciles, aunque según el sector, estas estructuras pueden variar y llegar a ser bastante más complejas.

Estas estructuras dependen de la estrategia de cada negocio y no todas serán siempre igual, pero ayudan a partir desde una base por la cual empezar conocer algunas de estas estructuras básicas.

Estructura silo

La estructura de silo es una arquitectura ramificada donde los enlaces van de arriba a abajo y viceversa. Las páginas de un mismo nivel solo se pueden enlazar entre ellos si se encuentran dentro de un mismo grupo.

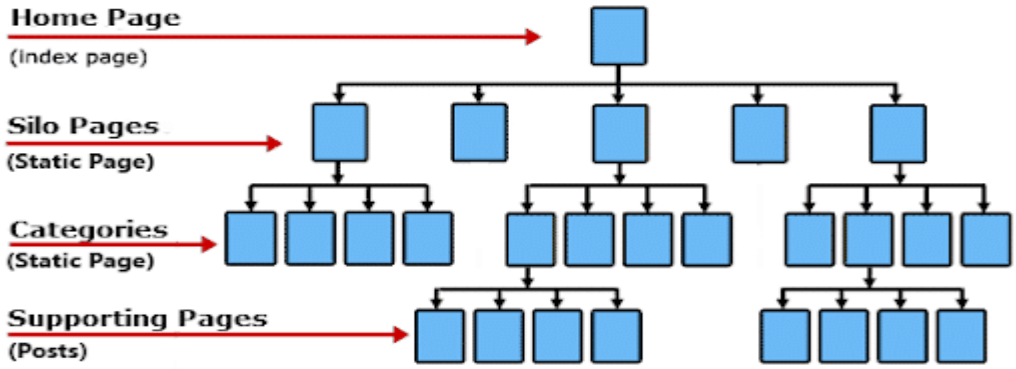


Imagen de

www.linkbuildingcorp.com

Estructura mixta

Este tipo de estructura es bastante similar a la anterior, pero no tiene la restricción anterior, cualquier página puede enlazar a las otras.

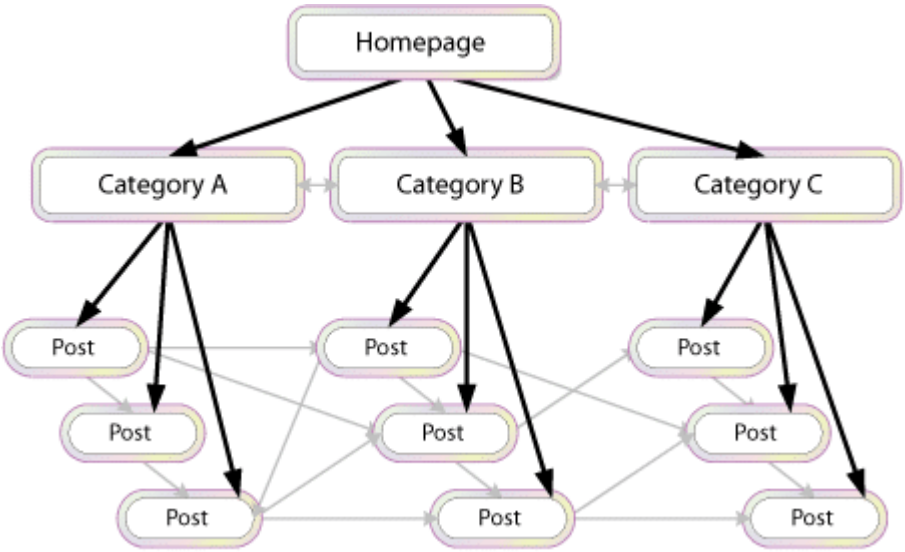


Imagen de Neilpatel.com

Estructura de clusters

La estructura de enlaces mediante clusters suele ser un mega contenido con sentido amplio para posicionar por keywords más complejas, el cual enlaza a otros contenidos más específicos para posicionar a otras keywords long tail, y estos a su vez, enlazan a este megacontenido.

Lo que pretenden es reforzar la temática y el significado interno de cada grupo de enlaces mediante la clasificación de contenidos en clusters de contenidos

¿Cómo utilizar una arquitectura web en una estrategia de SEO?

Una vez ya hemos visto todas las herramientas y hemos visto algunos ejemplos de estructuras de enlaces, vamos a empezar a crear una arquitectura web.

Arquitectura web desde 0

En una arquitectura web hay que tener en cuenta si la página va ir orientada a:

- Servicios
- Blog
- Venta de enlaces
- Tienda online
- Afiliados
- Otros tipos de webs

Cada una de estas webs tendrá una estructura distinta y, según la cantidad de páginas existentes, trataremos de enlazar la mayor cantidad de estas entre ellas mismas (arquitectura mixta), o trataremos de crear una estructura más ramificada (arquitectura silu) en caso de que la web sea muy grande.

Siempre se puede partir inicialmente de una arquitectura silu, para después pasar a una mixta, reforzando con enlaces las secciones más flojas a nivel de enlazado interno.

Para entender esto mejor, pondré un ejemplo. Las páginas de servicios, destinadas a capturar leads o de afiliados, suelen ser webs con pocas páginas, por lo que se intenta que todas estén entrelazadas entre ellas.

Las páginas de tiendas online o revistas, suelen acumular demasiado contenido con el paso de tiempo, por lo que es mejor una estructura ramificada o no transmitiremos bien la fuerza a los contenidos más importantes.

Además, en el caso de las tiendas online, lo que se pretende es dar más fuerza a las categorías, ya que, normalmente son las que más se buscan, y los productos son mucho más fáciles de posicionar y no requieren mucho esfuerzo para alcanzar las primeras posiciones en Google.

Arquitectura web ya creada

En una arquitectura web que ya está planteada y no podemos estructurarla nosotros mismos, debemos moldearla y realizar un estudio de todas las urls de la web.

Para ello, es importante extraer las urls por las que pasa el crawler de Google mediante los logs. Analizando los logs podemos saber cuántas veces pasa Google por las urls de nuestra web, además de poder comparar este dato con el tráfico que nos trae desde Google gracias a herramientas como Search Console y Google Analytics.

Es importante no olvidar, que también deberemos revisar el enlazado externo de la web y conocer cuáles son las urls más enlazadas, tanto a nivel interno como externo.

El cruce de datos podemos hacerlo gracias a excel y su función avanzada de tablas dinámicas o utilizar fórmulas para ello.

Una vez hemos analizado cómo está la arquitectura web actual, las urls que indexan, las que no, así como las que posicionan y las que no, toca reestructurar todo el enlazado interno.

Al reestructurar la web, seguramente existirán urls que eliminemos ya que no serán necesarias, o decidamos desindexar.

Si decidimos eliminar, no hay que olvidar que hay que hacer redirecciones 301 por cada página eliminada, a una página similar o a la categoría superior donde se encuentra esta. Sí que existirán casos (si son muchas urls sin enlaces o que no exista nada similar), que nos interesará desindexar las páginas directamente.

Si por el contrario, decidimos desindexar, pasamos a noindex y conforme pase el tiempo, luego bloqueamos la url mediante el fichero robots.txt. Existen dos alternativas más:

Desindexar desde Search Console las urls y bloquear desde robots

Bloquear desde robots.txt directamente la url, pero Google tardará más tiempo en desindexar la página.

Reglas para validar una página web

Una de las principales diferencias entre los ordenadores y los seres humanos es que los humanos estamos preparados para afrontar situaciones con información incompleta o errónea. Los ordenadores, sin embargo, necesitan exactitud y precisión para poder seguir adelante. Así, en el caso de los lenguajes los humanos podemos entendernos aunque no hablemos con corrección y sabemos reconocer la ironía, el cinismo, los juegos de palabras, las paradojas o las contradicciones. Sin embargo, los lenguajes de programación de ordenadores exigen siempre que los programas se ajusten exactamente a las reglas del lenguaje, sin ambigüedades.

El HTML no es un lenguaje de programación, pero tiene su propio vocabulario (las marcas, atributos y valores de atributos que podemos utilizar) y su propia sintaxis (las reglas de utilización de sus elementos). Las páginas web son documentos creados por humanos e interpretados por los ordenadores (por los navegadores), por lo que podemos entender el HTML como un lenguaje con el que los humanos nos comunicamos con los ordenadores, aunque sólo sea para decirles cómo tienen que mostrar el contenido de las páginas web a otros humanos.

Quizás uno de los motivos del éxito de la web (es decir, del HTML) fue que los navegadores siempre han sido capaces de procesar documentos con errores y de conseguir en muchos casos mostrarlos como deseaba su autor. Por supuesto, esa capacidad se debe al trabajo y esfuerzo de los programadores que crean los navegadores, que han incluido en los navegadores numerosas reglas para reconocer los errores más comunes e intentar corregirlos sobre la marcha.

En el año 2000, el W3C intentó con el XHTML obligar a que las páginas web dejaran de tener errores de sintaxis. Ese intento fue un fracaso, y en el HTML 5 se ha tomado el camino contrario, detallando cómo actuar ante violaciones de la propia sintaxis y consiguiendo un comportamiento similar en todos los navegadores. Por cierto, aunque en el HTML 5 se mantiene una variante con sintaxis estricta, el XHTML 5, su uso real es residual.

De todas formas, en un curso de creación de páginas web es lógico insistir en la necesidad de escribir páginas sin errores de sintaxis. Es verdad que podemos cometer errores y es bueno que los navegadores aun así consigan mostrar las páginas correctamente, pero si nos acostumbramos a escribir páginas plagadas de errores, en el momento en que los navegadores ya no puedan entender nuestras páginas será más difícil encontrar el origen del error porque habrá demasiados. Merece la pena acostumbrarse desde el principio a no cometer errores y crear páginas más fáciles de entender, modificar y reutilizar en el futuro.

Para ayudarnos a detectar errores debemos utilizar validadores. En esta lección se comenta la instalación y uso de validadores, tanto en el editor como en el navegador.

Reglas para escribir HTML

Obligatorio:

- delimitar el ámbito de las etiquetas (salvo excepciones)
- etiqueta interna se cierra antes que externa
- etiquetas a nivel de bloque pueden contener etiquetas a nivel de línea, pero no al revés
- etiquetas no sensitivas a mayúsculas
- en vez de espacios agregar raya al piso

Pero no está de más cumplir además con XHTML:

- Cerrar todos los tags, ie: ... , <li> ...
- Incluir marca de cierre para tags vacíos, ie: <hr />,
- Poner entre comillas todos los atributos, ie:
- Todos los tags en minúsculas
- Terminar con </html>
- El siguiente ejemplo es el contenido del archivo `eje2.htm` en el directorio corriente:

```
<HTML>
      <HEAD>
            <TITLE>P&aacute;gina de ejemplo</TITLE>
      </HEAD>
      <BODY>
            <H2> Encabezado </H2>
            <P>Primer p&aacute;rrafo debajo del encabezado.</P>
            <H3> Un encabezado <BR> en dos renglones </H3>
            <P>P&aacute;rrafo con partes <B>en negrita</B> </P>
   </BODY>
</HTML>
```

Etiqueta

HTML es un markup language, lo que significa que está escrito con códigos que puede leer una persona sin que sea necesario compilarlo primero. En otras palabras, el texto en una página web está «marcado» con estos códigos para dar instrucciones al navegador web sobre cómo mostrar el texto. Estas etiquetas de marcado son las propias etiquetas HTML.

Cuando escribes código en HTML, estás escribiendo etiquetas HTML. Todas las etiquetas HTML están hechas con un número de partes específicas, incluyendo:

- El carácter “menor que” <
- Una palabra o carácter que determina qué etiqueta se está escribiendo
- Cualquier número de atributos HTML que se quiera usar, escritos de la forma nombre =”valor”
- El carácter “mayor que” >

El hipertexto (HTML) es un lenguaje informático que forma la mayoría de las páginas web y plataformas online. HTML no se considera un lenguaje de programación, ya que no puede crear una funcionalidad dinámica. Sin embargo, los usuarios web pueden crear y estructurar secciones, párrafos y enlaces usando elementos, etiquetas y atributos.

En la actualidad existen 142 etiquetas HTML disponibles que permiten la creación de varios elementos. A pesar de que algunos ordenadores nuevos ya no admiten algunas de estas, es importante tener constancia de la existencia de ellas.

Sin embargo, el HTML ha ido evolucionando los últimos años. Para que nos hagamos una idea, la primera versión contaba solamente con 18 etiquetas. Desde entonces, cada nueva versión ha traído nuevas etiquetas y atributos. La actualización más importante fue la introducción en 2014 del HTML5. La principal diferencia entre ambas es que la nueva admitía nuevos tipos de controles de formularios. También incluía diversas etiquetas semánticas que describían mejor el contenido, como <article>, <headers> y <footer>.

La mayoría de páginas web tienen varias páginas HTML diferentes. Por ejemplo, una página de inicio, otra de producto, otra de contacto, etc. Cada una de estas tiene HTML separados. Los documentos HTML son archivos que acaban con .html o .htm. Un navegador lee el archivo y muestra su contenido para los internautas puedan verlo.

Como hemos comentado anteriormente, todas las páginas HTML contienen una serie de elementos HTML que a la vez disponen de diferentes etiquetas y atributos. Es decir, los elementos HTML son los componentes básicos de una página web. Una etiqueta contiene mucha información. Le indica al navegador dónde empieza y dónde termina cada elemento, mientras que un atributo describe las características.

Los elementos HTML suelen dividirse en tres partes. Esta combinación de las tres crea un elemento HTML.

Etiqueta de apertura: Se utiliza para indicar dónde empieza un elemento. Está envuelta en corchete de apertura y cierre. Por ejemplo puedes usar la etiqueta de inicio para crear un párrafo.

Contenido: El contenido es el resultado que ve la audiencia.
Etiqueta de cierre: Es lo mismo que la etiqueta de apertura pero con una barra inclinada delante del nombre del elemento. Es decir, para finalizar un párrafo.
Otra parte fundamental de un elemento HTML son los atributos. Estos tienen dos secciones:
Nombre: El nombre identifica la información adicional que un usuario quiere agregar.
Valor del atributo: Da más detalles que el anterior.

Etiqueta Básica

Hay una serie de etiquetas que son las más usadas para crear cualquier documento HTML, a continuación las explicamos:

- <body> para el contenido
- <head> para información sobre el documento
- división dentro del contenido
- <a> para enlaces
- para poner el texto en negrita
-
 para saltos de línea
- <H1>…<H6> para títulos dentro del contenido
- <img> para añadir imágenes al documento

- para listas ordenadas, <ul> para listas desordenadas, <li> para elementos dentro de la lista
- <p> para parágrafos
- para estilos de una parte del texto
- <body> </body> Indica la parte del cuerpo del contenido de un documento HTML. Es una etiqueta esencial para cualquier documento ya que indica donde empieza el contenido visible del documento.
- <head></head> La parte superior del documento HTML, es donde podremos indicar los metadatos: título del documento, hojas de estilos, javascript, CSS…
-   Un elemento que es usado mayoritariamente para agrupar otros elementos y actuar como plantilla de otros controles. La etiqueta nos ayuda a estructurar el documento en secciones.
- <a> Es una etiqueta que nos ayuda a poder crear un enlace a una página web. El atributo principal de la etiqueta HTML es href, donde pondremos el enlace al que queremos conectar. Otro atributo muy usado es target, el cual nos sirve para indicar si el enlace se abrirá en una nueva ventana o en la misma.
- <IMG> </IMG> Usamos la etiqueta IMG para mostrar imágenes dentro del contenido. Necesita el atributo src para funcionar, ya que será donde indicaremos desde donde tiene que mostrar la imagen.
- <OL> <OL> | <UL> <UL> Las etiquetas OL y LI nos sirven para crear listas, OL para listas ordenadas y UL para listas sin orden. Dentro de las listas, los elementos se identifican con la etiqueta LI.
- <SPAN> </SPAN> Con la etiqueta podemos personalizar el estilo de solamente una parte del texto.

Ventajas y desventajas del HTML

Como todo lenguaje informático, el HTML tiene sus ventajas y sus desventajas. Entre las ventajas, podríamos destacar que es apto para principiantes, que tiene una curva de aprendizaje poco profunda y que es accesible. Además, es de código abierto y completamente gratuito y se ejecuta de forma nativa en todos los navegadores web.

Por otro lado, entre las desventajas, se encuentra que para una funcionalidad dinámica es posible que haya que utilizar JavaScript o un lenguaje back-end como PHP. Además, los usuarios deben crear páginas web individuales para HTML, incluso si los elementos son los mismos y es posible que los navegadores más antiguos no muestren las etiquetas más nuevas.

Atributo en HTML

Los atributos se agregan a una etiqueta para proporcionar al navegador más información sobre cómo debe aparecer o comportarse la etiqueta. Los atributos constan de un nombre y un valor separados por un signo igual (=), con el valor entre comillas dobles. Aquí hay
un ejemplo style="color:black" ,.

Las etiquetas HTML pueden contener uno o más atributos. Los atributos se agregan a una etiqueta para proporcionar al navegador más información sobre cómo debe aparecer o comportarse la etiqueta. Los atributos constan de un nombre y un valor separados por un signo igual (=), con el valor entre comillas dobles. Aquí hay un ejemplo style="color:black",.

Los atributos HTML se pueden agregar a los elementos HTML para proporcionar más información sobre ese elemento

Ejemplo

<abbr title="Hypertext Markup Language">HTML</abbr>

Este ejemplo utiliza la etiqueta <abbr>, que se utiliza para indicar una abreviatura o un acrónimo. Pero se ha agregado algo extra a la etiqueta: un atributo. Este atributo particular (el atributo es title) proporciona un título para el elemento.

El nombre del atributo es title. En este ejemplo, le hemos dado un valor de Hypertext Markup Language.

El atributo title se puede usar (opcionalmente) en cualquier elemento HTML para proporcionar información adicional sobre el contenido del elemento. Cuando se usa con la etiqueta <abbr>, permite usar una extensión de la abreviatura o acrónimo (es decir, podemos decir lo que significa el acrónimo).

Cuando se usa el atributo title, la mayoría de los navegadores mostrarán su valor como “información de ayuda” cuando el usuario se desplace sobre el elemento.

Múltiples atributos

Puedes agregar más de un atributo a un elemento dado.

Este es un ejemplo de cómo agregar dos atributos al elemento <a> (que se usa para crear un hipervínculo a otra página web).

HTML desde Cero

El atributo href especifica la ubicación/dirección de la página web a la que estamos vinculando.

También utilizamos el atributo title para proporcionar un texto de consulta, en caso de que el usuario no lo supiera…

Más atributos

Hay muchos atributos diferentes disponibles para los elementos HTML. Aquí hay algunos atributos que encontrarás a menudo en cualquier sitio web moderno:

class: Se utiliza con Hojas de estilo en cascada (CSS) para aplicar los estilos de una clase determinada.

style: Se utiliza con CSS para aplicar estilos directamente al elemento (es decir, estilos en línea).de ayuda" para sus elementos. Tú suministras el texto.

Algunos atributos pueden usarse en cada elemento HTML (lista completa aquí), algunos están disponibles en muchos (pero no en todos) los elementos, mientras que otros atributos solo están disponibles en un elemento específico.

He reunido una lista alfabética de etiquetas HTML5. Cuando haces clic en cada etiqueta, puedes ver todos los atributos que se pueden usar con esa etiqueta, así como una explicación sobre cada atributo.

Pero hay muchos atributos y no es necesario que los memorices todavía. Lo bueno de los atributos es que, en la mayoría de los casos, son opcionales.

Muchos elementos HTML asignan un valor predeterminado a sus atributos, lo que significa que, si no incluyes ese atributo, se asignará un valor de todos modos. Dicho esto, algunas etiquetas HTML requieren un atributo (como el ejemplo de hipervínculo anterior).

Verás más atributos a lo largo de este tutorial.

Hay 3 tipos de atributos que puedes agregar a tus etiquetas HTML: Atributos de contenido específicos de elemento, Atributos globales y Atributos de controlador de eventos.

Elemento HTML.

Cuando cargas una página web en tu navegador, todos los elementos HTML se organizan de forma predeterminada. Así, los elementos <p> o párrafos se agrupan en bloques que **van de lado a lado en la página web**, mientras que, los elementos <a>, usados en los *links*, se organizan en **una sola línea**.

Casi todos los elementos HTML siguen alguno de estos patrones: pertenecen a los **elementos de bloque** o a los **elementos en línea**.

Elementos de bloque

Un **elemento de bloque HTML** es aquel que ocupa **todo el ancho de la página** o del elemento que lo contiene. Ya has visto algunos de ellos, como por ejemplo:

- <p>
- <h1>hasta <h6>
-

Aunque sólo pongas una palabra dentro del elemento <p>, ésta ocupará todo el ancho que tiene disponible. En el siguiente ejemplo, puedes ver que el elemento <p> ha sido coloreado; aunque sólo una parte tiene realmente texto, el tono azul **va de un extremo de la imagen al otro**.

Aunque sean pocas palabras, se extienden por toda la página.

Como los elementos de bloque ocupan todo el ancho de la página, suelen agruparse uno sobre otro:

Aquí hay otro elemento de párrafo que hace parte de los bloques.

Este párrafo también se organiza a lo ancho de la página. Es lo mismo.

Fíjate, todos se agrupan de esa manera.

Elementos en línea

Un **elemento en línea HTML** solo ocupa **el ancho de su contenido**. Mira algunos ejemplos con los que has practicado hasta ahora:

- <a>
- <b>
- <i>
- <img>

Para notar la diferencia, tomemos el grupo de párrafos de arriba y reemplacemos los elementos <p> por <a>. Lo que va a pasar es que, en vez de ocupar el ancho de la página, cada uno se limitará a ocupar el espacio suficiente para su contenido, en este caso, para cada frase. Aquí, coloreamos el fondo para que puedas ver la diferencia claramente:

¿Ves que no se extienden más allá del texto que contienen? A su vez, tienen tanto espacio que caben en una sola línea y no se moverán de ella hasta que el próximo elemento ya no quepa.

Gracias a esta característica los elementos en línea son perfectos para usarlos dentro de elementos de bloque.

Puedes usar elementos de bloque dentro de otros elementos de bloque, pero evita utilizar elementos de bloque dentro de elementos en línea. Es posible que el navegador los muestre correctamente, pero, como no es una forma correcta de usarlos, puede terminar mostrando un error o, simplemente, tu contenido se verá desordenado.

¿Cómo vas? ¿Has aprendido mucho sobre HTML?

No te preocupes por memorizar cuáles son los elementos de bloque y cuáles los de línea. Lo más importante es que los tengas **presentes y los diferencies** para saber por dónde empezar cuando algo en tu página web no se muestre como esperabas.

Elementos estructurales

Existen otros dos elementos que te permiten dar orden a tu página e incluso, organizar los elementos de bloque y en línea. Estos son: y .

Div
El es un **elemento de bloque** que se utiliza exclusivamente para **agrupar otros elementos**. Por ejemplo, puedes reunir unos cuantos párrafos con un elemento , y quedaría así:

```
<div>
<p>Aquí hay un elemento de párrafo, que es de bloque.</p>
<p>Lo mismo pasa con este otro elemento de párrafo.</p>
<p>Fíjate en la forma como están agrupados.</p>
</div>
```

Visualmente el elemento no aparece en la página. Si cargas el HTML de arriba en tu navegador, no se verá nada. Los tres párrafos agrupados se verán igual que antes, solo que al contener los elementos evitan que se mezclen con otros.

Cuando empieces a aprender **CSS** y **JavaScript** te darás cuenta de los grandes beneficios del . Además, al agrupar elementos con , este hace que sea más fácil buscar ciertas partes de tu página web cuando quieras cambiar su aspecto o hacer algo con ellas.

Span

El elemento funciona de forma **muy similar** al elemento , pero **para elementos en línea**.
Imagina que quieres agrupar algunas de las palabras de un párrafo <p>, pero no todas. Como el sirve para los elementos de bloque, en este caso, no aplicaría. Pero , sí:

```
<p>Aquí hay un párrafo, pero <span>estas palabras</span> están agrupadas.</p>
```

Al igual que el elemento , el elemento no se ve en la página si la cargas en el navegador.

Estructura básica de Archivo Web

Para crear una página web se necesita un documento HTML utilizando tres elementos o tags principales que cualquier sitio Web usa: html, head y body.

Un documento web (o página web) es, como hemos apuntado en la introducción, un conjunto de tags HTML que se escriben en un editor de texto plano (sin formato) y se ejecutan en un navegador web.

CÓMO CREAR UNA PÁGINA WEB

Para crear una página web se necesita un documento HTML utilizando tres elementos o tags principales que cualquier sitio Web usa: html, head y body.

Para crear un verdadero documento HTML comenzará con tres elementos contenedores:

- <html>
- <head>
- <body>

Estos tres se combinan para describir la estructura básica de la página:

- **<html>**: Este elemento envuelve todo el contenido de la página (excepto la DTD)
- **<head>**: Este elemento designa la parte de encabezado del documento. Puede incluir información opcional sobre la página Web, como puede ser el título (el navegador lo muestra en la barra de título), palabras clave de búsqueda opcionales y una hoja de estilo opcional.
- **<body>**: Este elemento alberga el contenido de su página Web, es decir, aquello que queremos que aparezca en el área de navegación del navegador..

Sólo hay una manera correcta de combinar estos tres elementos. He aquí su colocación exacta, con el *doctype* al comienzo de la página:

```
<!DOCTYPE html>
<html>
        <head>
                ...
        </head>
        <body>
                ...
        </body>
</html>
```

Toda página Web utiliza esta estructura básica. Los puntos suspensivos (...) muestran dónde insertaría la información adicional.

Una vez colocado el esqueleto XHTML, debe añadir dos conectores más a la mezcla.
Toda página Web requiere un elemento **<title>** en la sección del encabezado. A continuación, deberá crear un contenedor para el texto en la sección del cuerpo de texto (<body>).

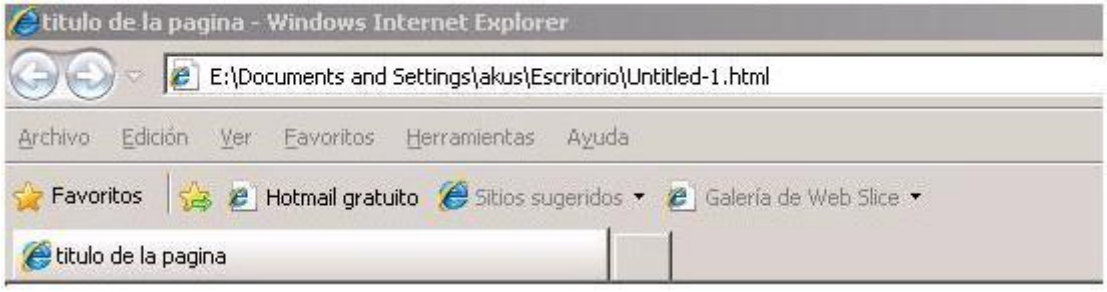
Un elemento contenedor de texto multiuso es **<p>**, que representa un párrafo. Veamos con más detalle los elementos que hay que agregar:

- **<title>**: Establece el título de la página Web, el cual tiene varias funciones. Primero, los navegadores lo muestran en la parte superior de la ventana. Segundo, cuando un visitante crea un marcador para la página, el navegador emplea el título para etiquetarlo en el menú Marcador (o favorito). tercero, cuando la página aparece en una búsqueda Web, el motor de búsqueda suele enseñar este título como primera línea en los resultados, seguido de un fragmento del contenido de la página.
- **<p>**: Indica un párrafo. Los navegadores no los sangran pero añaden un pequeño espacio entre varios consecutivos para mantenerlos separados.

He aquí la página con estos dos nuevos ingredientes:

```
<!DOCTYPE html>
<html>
        <head>
                <title>
                        Bienvenido a mi sitio Web
                </title>
        </head>
        <body>
                <p></p>
        </body>
</html>
```

Si abre este documento en un navegador Web, verá que la página está vacía, pero ahora aparece el título.



Cuando un navegador muestra una página Web, el título se presenta en la parte superior de la ventana, con el texto al final. Si el suyo utiliza la navegación por pestañas, el título también aparece en ellas.

Tal y como está ahora, este documento HTML es una buena plantilla para futuras páginas. La estructura básica está en su lugar; simplemente necesita cambiar el título y añadir algo de texto.

Lo primero que tenemos que saber es que en toda página web existen dos partes claramente diferenciadas: la cabecera, o **head**, y el cuerpo, o **body**.

Vamos hacer es crear una carpeta, dentro de "Mis documentos", para almacenar los ficheros de prueba que vayamos a usar.

A esta carpeta la llamaremos **pruebas-html** en el resto de los ejercicios. Con la carpeta abierta, haga doble clic sobre el ícono que representa al bloc de notas.

Se le abrirá un documento en blanco.

Escriba el siguiente texto:

```
<!DOCTYPE  html>
<html>
        <head>
                <title>Título de la página</title>
        </head>
        <body>
        </body>
</html>
```

Cuando lo tenga escrito, guárdelo en la carpeta con el nombre **plantilla.html**.

Es vital que la extensión sea **.html**, ya que sólo por la extensión el sistema operativo reconoce este archivo como un documento web, y no como un simple y archivo de texto.

El nombre del archivo deberá escribirse como está, en minúsculas y sin espacios ni caracteres especiales.

Los únicos signos de puntuación admitidos son el punto (sólo uno), que lo usaremos para separar el nombre de la extensión y el guión bajo.

El nombre podrá contener letras y números, pero deberá empezar por una letra. Así mismo, nos abstendremos de meter en el nombre de un archivo letras acentuadas, eñes, cedillas, o cualquier otro carácter de algún alfabeto local.

Concepto de Imagen en HTML

Para poner una imagen simple en una página web, utilizamos el elemento <img>. Se trata de un elemento vacío (lo que significa que no contiene texto o etiqueta de cierre) que requiere de por lo menos de un atributo para ser utilizado: src (a veces denominado por su nombre completo, *source*). El atributo src contiene una ruta que apunta a la imagen que quieres poner en la página, que puede ser una URL relativa o absoluta, de la misma forma que el elemento <a> contiene los valores del atributo href.

Por ejemplo, si tu imagen se llama dinosaur.jpg, y está en el mismo directorio que tu página HTML, deberás incrustar la imagen de la siguiente manera:

```
<img src="dinosaur.jpg">
```

Si la imagen estaba en el subdirectorio images, que estaba en el mismo directorio que la página HTML (lo que Google recomienda con fines de indización y posicionamiento en buscadores SEO), entonces deberías incrustar la imagen así:

```
<img src="images/dinosaur.jpg">
```

y así sucesivamente.

Puedes incrustar la imagen usando la URL absoluta, por ejemplo:

```
<img src="https://www.example.com/images/dinosaur.jpg">
```

Pero esto no tiene sentido, solo hace que el navegador trabaje más buscando la dirección IP desde el servidor DNS cada vez, etc. Casi siempre mantendrás las imágenes para tu sitio web en el mismo servidor de tu HTML.

Advertencia: La mayoría de imágenes tienen derechos de autor. **No** muestres una imagen en tu página web a menos que:

1) seas dueño de la imagen,
2) tengas permiso escrito explícito del dueño de la imagen o
3) tengas suficientes pruebas de que la imagen es de dominio público.

El incumplimiento de las normas de los derechos de autor es un acto ilegal y poco ético. Por lo tanto, no apuntes **nunca** tu atributo src a una imagen que esté alojada en un sitio web si no tienes el permiso para hacerlo. Esto se llama *hotlinking*. Asimismo es ilegal robar el ancho de banda de alguien. Además, ralentiza tu página y te deja sin control sobre la imagen si la eliminan o reemplazan por otra que incluso podría resultar embarazosa.

Nuestro código anterior debería darnos el resultado siguiente:

Nota: Los elementos como <img> y <video> a veces se denominan **elementos reemplazados**. Esto se debe a que el tamaño y el contenido del elemento se especifican en un recurso externo (como un archivo de imagen o video), no en el contenido del elemento en sí.

Nota: Puedes encontrar el ejemplo terminado de esta sección en Github (consulta también el código fuente).

El próximo atributo que veremos es alt. Su valor debe ser una descripción textual de la imagen para usarla en situaciones en que la imagen no puede ser vista/mostrada o tarde demasiado en mostrarse por una conexión lenta a internet. Por ejemplo, nuestro código anterior podría modificarse así:

```
<img src="images/dinosaur.jpg"
    alt="La cabeza y el torso de un esqueleto de dinosaurio;
        tiene una cabeza grande con dientes largos y afilados">
```

La forma más fácil de probar el texto alt es escribir mal el nombre de archivo. Si, por ejemplo, escribimos el nombre archivo de nuestra imagen como dinosooooor.jpg, el navegador no podrá mostrar la imagen, en su lugar mostrará el texto alternativo:

¿Qué hay que escribir exactamente en el atributo alt? Esto depende en primer lugar de *por qué* la imagen está en ese lugar. En otras palabras, qué se pierde si la imagen no aparece:

- **Decoración.** Para las imágenes decorativas deberían utilizarse imágenes de fondo CSS. Pero si es inevitable usar HTML, la mejor forma de hacerlo es con alt="". Si la imagen no es parte del contenido, el lector de pantalla no debería malgastar el tiempo en leerla.
- **Contenido.** Si tu imagen proporciona información significativa, proporciona la misma información en un texto alternativo (alt) breve. O mejor aún, en el texto principal que todos pueden ver. No escribas texto alternativo redundante. ¿Acaso no resultaría molesto para un usuario con visión ordinaria si todos los párrafos se escribieran dos veces en el contenido principal? Si la imagen se describe en el cuerpo principal del texto de modo adecuado, puedes simplemente usar alt="".
- **Enlace.** Al poner una imagen dentro de una etiqueta <a> para convertirla en un enlace, aun debes proporcionar texto de enlace accesible. En tal caso podrías escribirlo dentro del mismo elemento <a>, o dentro del atributo alt de la imagen. Lo que mejor funcione en tu caso.
- **Texto.** No deberías poner tu texto en imágenes. Si tu título de encabezado principal necesita, por ejemplo, un sombreado paralelo, usa CSS para ello en vez de poner el texto en una imagen. Pero, *si realmente no puedes evitarlo*, deberías proporcionar el texto en el atributo alt.

En el fondo, la clave es ofrecer una experiencia usable, incluso cuando las imágenes no puedan verse. Esto asegura que ningún usuario pierda ninguna parte del contenido. Desactiva las imágenes en tu navegador y observa cómo se ven las cosas. Te darás cuenta de lo útil que resulta el texto alternativo cuando no es posible ver la imagen.

Anchura y altura

Puedes usar los atributos ancho (width) y alto (height) para especificar la anchura y altura de tu imagen. Puedes encontrar el ancho y el alto de tu imagen de diversas maneras. Por ejemplo, en Mac puedes usar Cmd + I para mostrar información del archivo de imagen. Volviendo a nuestro ejemplo, podríamos hacer esto:

```
<img src="images/dinosaur.jpg"
    alt="La cabeza y el torso de un esqueleto de dinosaurio;
        tiene una cabeza grande con dientes largos y afilados"
    width="400"
```

```
    height="341">
```

Esto no proporciona una gran diferencia en la pantalla en circunstancias normales. Pero si la imagen no se muestra, por ejemplo, porque el usuario acaba de acceder a la página y esta aún no se ha cargado, observarás que el navegador reserva un espacio para la imagen:

Hacerlo así es bueno porque la página se carga con mayor rapidez y fluidez.

Sin embargo, no deberías alterar el tamaño de tus imágenes utilizando atributos HTML. Las imágenes podrían verse granulosas y borrosas si estableces un tamaño demasiado grande; o bien demasiado pequeñas, y se desperdiciaría ancho de banda descargando una imagen que no se ajusta a las necesidades del usuario. La imagen también podría quedar distorsionada, si no mantienes la proporción de aspecto correcta. Deberías utilizar un editor de imágenes, para dar a tu imagen el tamaño adecuado, antes de colocarla en tu página web.

Nota: Si tienes que alterar el tamaño de una imagen es mejor usar CSS.

Al igual que con los enlaces, también puedes añadir atributos title a las imágenes para proporcionar más información de ayuda si es necesario. En nuestro ejemplo, podríamos hacer esto:

```
<img src="images/dinosaur.jpg"
    alt="La cabeza y el torso de un esqueleto de dinosaurio;
        tiene una cabeza grande con dientes largos y afilados"
    width="400"
    height="341"
    title="Exposición de un T-Rex en el museo de la Universidad de Manchester.">
```

Esto nos da una etiqueta de ayuda (tooltip) como las de los enlaces:

Sin embargo, no se recomienda incluir esta propiedad en las imágenes. title presenta algunos problemas de accesibilidad, principalmente porque los lectores de pantalla (*screen readers*) tienen un comportamiento imprevisible y la mayoría de navegadores no la mostrarán a menos que pases el ratón por encima de la imagen (y por tanto es inútil para quien usa teclado). Si estás interesado en esta cuestión, puedes leer el artículo The Trials and Tribulations of the Title Attribute de Scott O'Hara.

Tipos de imágenes o formatos que podemos utilizar en una página.

Para comenzar, vamos a entender que, así como ya se expuso en nuestro artículo exclusivo sobre diseño gráfico, existen dos tipos de imágenes, digitalmente hablando:

Bitmap y **vector**.

¡No confundas **formatos con tipos!** Esos dos tipos de imágenes serán abordado en este artículo sobre **formatos de imágenes.** Ellos poseen algunas diferencias y afectan directamente la forma en como son utilizados.

Bitmap
También conocido como “mapa de bits” (traducción literal) o imagen rasterizada (del inglés *raster*) es uno de los tipos de imágenes más comunes.

Los bitmaps son, literalmente, un **mapa de bits.** Eso significa que la imagen es formada por diversos puntos minúsculos (pixeles).

A cada uno de los pixeles se les asigna un color y, a través de coordenadas X y Y, esos pixeles de colores se posicionan en una malla y, así, forman una imagen.

Las fotos son bitmaps.

Son imágenes que se basan en polígonos formados por puntos. Estos puntos son interpretados por el computador teniendo en cuenta sus distancias.

Siendo así, los **vectores son infinitamente escalables.** Puedes aumentar un vector cuanto quieras, pues no hay pérdida de resolución en el proceso.

Los vectores presentan una cantidad de detalles menor que un bitmap por sus limitaciones, sin embargo, existen artistas que desarrollan técnicas usando vectores que se aproximan al gráfico de fotos reales, como en el ejemplo:

Ahora que sabemos la diferencia entre estos dos tipos de archivos, vamos a aprender sobre **los diferentes formatos de imagen.**

BMP
El significado de esta sigla es, literalmente, *bitmap.*

En la década de los 90 podían encontrarse muchas imágenes con la extensión “*.BMP”* en la internet. El problema es que las tasas de compresión de los archivos de BMP son muy bajas, lo que hace que los archivos queden muy pesados (para que la imagen no pierdan resolución).

TIFF
También conocido como TIF, es el formato de imagen que se utiliza mucho para archivos de impresión. Es muy versátil y mantiene la resolución, posibilita el uso de capas, entre otros, pero es también un formato muy pesado.

JPEG
Igualmente conocido como JPG, es el más famoso de los formatos utilizados digitalmente. Su tasa de compresión ajustable es lo que anima su uso en los medios digitales.
A través de esa posibilidad, puedes equilibrar calidad/tamaño de la imagen.
Aun así, incluso en lo mínimo de compresión, el tamaño del archivo no queda tan grande.
Es altamente recomendado para los medios digitales, sin embargo, no poseen el canal alpha (transparencia).

GIF(*Graphics Interchange Format)*

(formato para intercambio de gráficos) y fue el primer formato de imágenes con un alta tasa de compresión, gracias a que redujo sensiblemente el tamaño de las imágenes y posibilitó la descarga rápida.
Se popularizó por contar con la tecnología *interlaced*, que posibilita que la imagen sea cargada gradualmente. Siendo así, el usuario puede interrumpir la carga teniendo solo una parte de la imagen cargada.

Otro factor que facilitó mucho el usó de lo GIFs al principio de la internet, fue la posibilidad de tener imágenes con el fondo transparente (canal alfa) y también por poder ser un medio no estático.

Los GIFs poseen una limitación de colores: tan solo 256 colores. Eso hace que las imágenes que sean muy complejas, pierdan muchos detalles.

Hoy en día, los GIFs son prácticamente un sinónimo de animación. Este formato posibilita que diversas imágenes sean exhibidas en secuencia, generando un clip sin audio.

Los GIFs animados presentan una altísima tasa de enganche, sea estos graciosos, memes, o sean infográficos animados como este.

GIF estático X GIF animado

Como ya fue dicho, el GIF es un formato de imagen bien versátil que, cuando fue creado, presentaba posibilidades innovadoras para la época, ¡volviéndose bien popular en toda la internet!

Con la evolución de los medios de transmisión de datos, la internet quedó más rápida y las imágenes en el formato de GIF estáticas quedaron muy anticuadas.

Queremos imágenes nítidas y detalladas, pero la limitación de los 256 colores de los GIFs hizo que las imágenes en este formato cayeran en desuso.

Por más que posibiliten la transparencia (canal alpha), esa transparencia tienen muchas limitaciones y presentan *aliasing* (cuando los pixeles quedan muy expuestos y pareciera que existe un error en la imagen). Por eso, cuando necesitamos de imágenes estáticas con transparencia, usamos PNG.

Las animaciones presentan aproximadamente 15 cuadros por segundo (es decir, 15 imágenes exhibidas cada segundo). Esa velocidad hace que las limitaciones de los GIFs no sean tan perceptibles (o simplemente no incomoden tanto).

Además, por más que la calidad no sea increíble y los movimientos no sean tan fluidos, la carga es muy rápida. O sea, una cosa compensa otra, y el GIF animado continuó siendo popular.

PNG (*Portable Network Graphics).*
En 1995 el algoritmo de compactación del GIF (LZW) fue planteado. De este modo, Adobe invirtió en la creación de un nuevo formato que pudiera sustituir al GIF y, por qué no, volverlo aún mejor.

EL PNG, aunque no soporte animaciones, presenta diversas ventajas en relación al GIF.

- Posee una variación de colores infinitamente mayor a la de los GIFs (que es tan solo de 256 colores).
- También posee el canal alpha adicional, posibilitando inclusive las variaciones de opacidad lo que evita que las imágenes presenten *aliasing* y expande las posibilidades de aplicación de la imagen.
- Utiliza un algoritmo de compactación muy eficiente, generando imágenes de altísima calidad y un tamaño razonable para los patrones actuales de la internet.
  Este es, actualmente, **el formato más indicado para utilizar en tus estrategias digitales,** ¡por tener un tamaño en bits adecuado y mantener la calidad de las imágenes altísima!

PDF(***Portable Document Format)*** **(archivo de documento portátil) y fue creado por Adobe en 1993.**

Existía la necesidad de un tipo de archivo universal, que funcionara independientemente del software que había sido usado para su ejecución, del sistema operacional, de su resolución y tamaño.

Los PDFs son muy versátiles, debido a que pueden almacenar bitmaps, vectores, textos, pueden tener diversas páginas, entre otras innumerables funciones.

EPS (*Encapsulated PostScript)*.
También fue desarrollado por Adobe, pero fue posteriormente sustituido por el PDF.

SVG (*Scalable Vector Graphics).*
Un formato libre (sin vínculos con cualquier empresa). El SVG es un formato vectorial que puede ser reconocido por la mayoría de los navegadores web modernos. Siendo así, puedes usar ese formato en tu sitio web, blog, etc.

Otra función interesante del SVG es que puede ser animado a través de la programación en *HTML 5.0*. Este formato se popularizó de forma rápida por su ínfimo tamaño y su escalabilidad infinita.

Atributos de la etiqueta img

El elemento de imagen/**Image** configura gráficos en una página web. Estos gráficos pueden ser fotografías, pancartas, logotipos de empresas, botones de navegación, etc. Estás limitado solo por tu creatividad e imaginación.
El elemento de imagen es un elemento vacío y no está codificado como un par de etiquetas de apertura y cierre. El siguiente ejemplo de código configura una imagen llamada logo.gif, que se encuentra en la misma carpeta que la página web:

```
<img src="logo.gif" height="200" width="500" alt="HTML desde Cero">
```

- El atributo src especifica el nombre de archivo de la imagen.
- El atributo alt proporciona un reemplazo de texto, típicamente una descripción de texto, de la imagen.

- El navegador reserva la cantidad correcta de espacio para tu imagen si utiliza los atributos de altura y anchura con valores iguales o aproximadamente al tamaño de la imagen.

Proporcione valores precisos para la altura y el ancho de la imagen para conservar la relación de aspecto de la imagen, que es la relación proporcional entre el ancho y la altura de una imagen. **El navegador puede sesgar o distorsionar la imagen si proporciona valores inexactos** para la altura y/o el ancho de la imagen.

Atributos de la etiqueta figure

La etiqueta <figure> permite agregar contenido variado asociadas a una descripción o título, el cual es asociado semánticamente a este, de tal forma que podemos agregar una imagen, ilustración, diagrama, o incluso código. Como regla general, el contenido que agreguemos debe de estar relacionado al texto principal de la página.

Un claro ejemplo donde se puede emplear <figure> es para agregar imágenes a un artículo, donde las imágenes tienen por lo general una descripción justo por debajo. Un claro ejemplo sería cualquier artículo de Wikipedia, ya que este inserta a lo largo de todas sus secciones imágenes que tiene que ver con el texto en cuestión.

Actualmente <figure> es más utilizado para representar imágenes, porque es lo más común en una página además del texto, pero es importante resaltar que se puede utilizar para representar cualquier contenido. La clave de <figure> es que nos permite asociar cualquier cosa con una descripción, de tal forma que son asociados semánticamente.

La etiqueta <figure> se utiliza en conjunto con la <figcaption> para representar la descripción del contenido. La etiqueta <figure> puede tener cualquier contenido seguido de <figcaption> o al revés, <figcaption> seguido de cualquier contenido.

Si quieres aprender más sobre las mejoras y la web semántica, **puedes ver estre otro articulo** donde lo explico con más detalle.

Etiqueta Figure en acción:

Para comprender mejor como es que figure se puede utilizar, analizaremos un pequeño ejemplo de una página con una imagen, esta imagen tiene una descripción asociada. La idea es comparar una página que utiliza la etiqueta <img> y otra de cómo quedaría con la etiqueta <figure> .

```
<!DOCTYPE HTML>
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="styles.css">
    </head>

    <body>
        <section class="container">
            <hgroup class="split">
```

```
                <h1>Ejemplo utilizando Figure</h1>

                <h2>Este ejemplo utiliza <strong>figure</strong> para mostrar la
imagen</h2>

            </hgroup>

            <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer
ac placerat sapien, eu molestie mi. Cras feugiat feugiat hendrerit. Mauris inter
dum condimentum augue at cursus. Fusce eget gravida odio. Morbi diam leo, aliqua
m et tortor sed, tempus sollicitudin elit. Sed luctus sagittis risus, at luctus
nunc rutrum tincidunt. Integer faucibus est sit amet auctor imperdiet. Morbi pos
uere quam in nisl pellentesque aliquam. Etiam a nulla non orci ultricies suscipi
t eu vitae purus.</p>

            <figure>

                <img src="http://assets.climatecentral.org/images/made/3_16_16_u
pton_paris_720_405_s_c1_c_c.jpg" >

                <figcaption>Fig.1 - París de noche (Enero de 2016)</figcaption>

            </figure>

            <p>Quisque eget venenatis ex. Aenean at risus bibendum, placerat leo
ac, ullamcorper quam. Curabitur tempus tellus diam, ut suscipit orci faucibus ve
l. Etiam id leo diam. Vivamus eget ex at sapien suscipit porttitor eu in dui. Su
spendisse sem ipsum, euismod sodales auctor id, vulputate nec ex. Aenean et maxi
mus purus. Duis eu est a enim cursus accumsan nec in tellus. In gravida eros odi
o, ut consectetur neque mollis vitae. Cras at mattis diam. Vestibulum eleifend s
ed tellus vel feugiat.</p>

        </section>

    </body>

</html>
```

Atributos de etiqueta figurecaption

La etiqueta HTML <figcaption> es un elemento de contenido que permite definir un texto coherente y complementario en forma de leyenda o título de un bloque de contenido, representado a su vez con la etiqueta <figure>.

Mientras el elemento HTML <figure> es usado para mostrar contenido autónomo (del resto de un texto web) en forma de imagen, ilustración, gráfico, tabla de datos, vídeo, audio o fragmentos de código, entre otros, la etiqueta <figcaption> es opcional y proporciona una breve identificación o descripción asociada a ese elemento macro.

Por su semántica (viene de figure «imagen/ilustración» y caption «título/leyenda», en inglés), indica el título de una imagen o ilustración. Puede ser ubicado antes (arriba) o después (debajo) del elemento que se enmarca con la etiqueta <figure>. Sin embargo, no es independiente, <figcaption> está dentro de <figure>, funcionan como una unidad.

Ejemplo de uso

El siguiente es un ejemplo de uso simple, muestra la visualización como código correcto de una imagen con un pie:

```
<figure>
  <img src=https://i.pinimg.com/originals/e4/ef/98/e4ef988696aab947c6b939eafc941fea.jpg
alt="europa política mapa">
  <figcaption> Mapa político de Europa actual.</figcaption>
</figure>
```

Lenguaje del código: HTML, XML (xml)
Se puede observar que entre las etiquetas de bloque <figure> y </figure> se inserta el enlace de una imagen, representado dentro de la etiqueta de hipertexto <img>, seguida del texto que la identifica como título, precedido de la etiqueta <figcaption> y delimitado con </figcaption>.

Atributos obligatorios

Los campos de texto constituyen la principal forma de entrada de datos en un formulario, permitiendo al usuario introducir información no normalizada mediante caracteres alfanuméricos.

Su sintaxis es la siguiente:

```
<input type="text" ...>
```

Los atributos obligatorios que deberán añadirse a todos los campos de texto de un formulario son los siguientes:

- id: Identificación del elemento, que permitirá cumplir ciertas pautas de accesibilidad como son el uso no intrusivo del javascript.
- name: Nombre del campo, necesario para que el elemento sea identificado en el envío (submit) del formulario.
- size: Tamaño del campo en la pantalla ajustado al número de caracteres que contendrá habitualmente el mismo.
- maxlength: Longitud máxima de caracteres que podrán introducirse en el campo, evitando errores en el proceso de persistencia de la información por intentar almacenar un texto con una longitud mayor que la que soporte el campo de la base de datos.
- tabindex: El índice de tabulación para cumplir con las pautas de usabilidad.

Cambios sufridos en HTML5:

En HTML5 existen nuevos atributos para este tipo de campo. Entre ellos, cuando proceda, serán obligatorios los siguientes:

- Required: indica que es obligatorio introducir un valor en el campo para enviar ese formulario, algo que en XHTML y HTML4 se comprueba con Javascript o en el servidor.
- Min y Max: el valor que se introduzca en el campo debe estar comprendido en un rango dado por los valores de estos atributos.
- Autofocus: se utiliza en el primer elemento de un formulario para indicar que debe obtener el foco al cargar la página.

Atributo	Valor	Descripción
src	Dirección URL	Video a reproducir. Obligatoria si actua como etiqueta contenedora.
poster	Dirección URL	Muestra una imagen a modo de presentación.
preload	auto | metadata | none	Indica como realizar la precarga del video.

mediagroup	nombre	Establece un nombre para un grupo de contenidos multimedia.
autoplay		Comienza a reproducir el video automáticamente.
loop		Vuelve a iniciar el video cuando finaliza su reproducción (bucle).
muted		Establece el video sin sonido (silenciado).
controls		Muestra los controles de reproducción. Por defecto no se muestran.
width	tamaño	Indica el tamaño de ancho del video.
height	tamaño	Indica el tamaño de alto del video.

Atributo de Etiqueta de Video

En HTML5 se introduce la interesante posibilidad de mostrar videos directamente desde nuestro navegador. De hecho, si arrastramos un video a la ventana del navegador, veremos que comienza a reproducirse en él. Para poder insertar videos en nuestras páginas HTML tenemos que utilizar la etiqueta <video>, que junto a la etiqueta <source> podremos utilizar estas capacidades multimedia de HTML5.

Video (modo básico)

La etiqueta <video> tiene varios atributos a nuestra disposición:
Un primer ejemplo muy básico para colocar un video en nuestra página web:

Eemplo

```
<video src="video.mp4" width="640" height="480"></video>
```

Sin embargo, esto mostrará el primer fotograma del video, con un tamaño de 640x480, pero se verá como una imagen, ya que no muestra los controles del video y tampoco tiene la autoreproducción activada. Podríamos solucionarlo indicando los atributos controls o autoplay.

Otro Ejemplo

```
<!-- Ejemplo 1 -->
<video src="video.webm" poster="presentacion.jpg" controls></video>
<!-- Ejemplo 2 -->
<video src="video.mp4" autoplay muted loop></video>
```

En este caso cargamos un video, pero que no se mostrará porque se ha indicado que se utilice una imagen de presentación que se mostrará hasta que el usuario pulse en el botón de reproducir de los controles. En el segundo ejemplo, tenemos un video que se reproduce automáticamente al cargar la página, en silencio y en bucle, iniciándose una y otra vez.

Desde 2017, Chrome, Firefox y otros navegadores establecieron un cambio de políticas con el atributo de reproducción automática autoplay, por lo que no funcionará salvo que el usuario haya interactuado antes en la página.

Formatos de video

Antes de continuar con el modo avanzado de etiquetas de video, debemos conocer una serie de conceptos básicos y los diferentes formatos de video que existen actualmente. En primer lugar, debemos saber que un archivo de video tiene dos partes principales: el formato contenedor, que es el formato del video en sí, mientras que en su interior puede tener múltiples componentes codificados con diferentes codecs.

De hecho, un video básico suele tener, como mínimo, un componente de video y otro de audio, pero puede tener muchos más (subtítulos, imágenes, etc...). Estos detalles son muy importantes, ya que dependiendo del formato y/o codec de un video, puede que sea factible utilizarlo para web o no, así que hay que conocer un poco sobre estos conceptos.
A continuación tenemos un listado de los formatos/codecs más conocidos y utilizados:
Otros formatos como MOV, FLV, 3GP, MPG, RMVB o ASF/WMV no se recomiendan para su

Formato	Codec utilizado	Características	¿Recomendado?
MP4	x264, DivX H264	Alta calidad. Codec x264 libre.	Sí, buen soporte
WebM	VP8, VP9	Alternativa libre a MP4 de Google.	Sí, soporte medio
AV1	Basado en VP10, Daala y Thor	Compite con HEVC/H.265	No, soporte bajo
HEVC	x265, DivX HEVC	Futura evolución de MP4.	No, poco soporte
OGV	Theora	Alternativa libre a MP4.	Con precaución
MKV	Matroska	Buena compresión. Potente.	No, alto consumo CPU
AVI	XviD, DivX 3/5	Menor compresión que MP4.	No, anticuado

utilización en web ya que son anticuados, propietarios o poco eficientes. Si te interesa saber más sobre estos temas, aconsejo la lectura del artículo Formatos de video: Todo lo que deberías saber.

Video (modo avanzado)

Sin embargo, aún con toda esta información, no hemos visto ni la mitad de posibilidades multimedia que tenemos con HTML5. La etiqueta <video> también puede actuar como etiqueta contenedora e incluir varias etiquetas HTML para dotar de mayor compatibilidad, o capacidades adicionales.

Videos alternativos

Si utilizamos la etiqueta <video> como etiqueta contenedora, podemos incluir etiquetas <source> en su interior para proporcionar formatos alternativos y tener mayor compatibilidad con otros navegadores y navegadores antiguos que no soporten HTML5:

```
<video width="640" height="480">
  <source src="video.mp4" type="video/mp4" />
  <source src="video.webm" type="video/webm" />
  <source src="video.ogv" type="video/ogg" />
  <img src="imagen.png" alt="Video no soportado" />
  Su navegador no soporta contenido multimedia.
</video>
```

En este ejemplo, los navegadores no mostrarán todos los contenidos a la vez, sino que seguirán el siguiente procedimiento:

Intenta mostrar el primer formato (MP4). Si el navegador no soporta este formato, salta al siguiente.

Intenta mostrar el segundo formato (WEBM). Si el navegador no soporta este formato, salta al siguiente.

Intenta mostrar el tercer formato (OGV). Si el navegador no soporta este formato, salta al siguiente.

Si se trata de un navegador que no soporta HTML5, intentará mostrar la imagen.

Si se trata de un navegador de terminal de texto (o sin capacidades gráficas), mostrará el texto "Su navegador no soporta contenido multimedia".

De esta forma tenemos soporte completo para todo tipo de dispositivos.

Utilizando los fragmentos multimedia se pueden conseguir algunas acciones interesantes, como por ejemplo especificar el momento concreto del video (o audio) en el que se quiere empezar a reproducir o terminar de reproducir. Veamos unos ejemplos:

```
<!-- Ejemplo 1 -->
<video autoplay controls src="video.mp4#t=15"></video>

<!-- Ejemplo 2 -->
<video autoplay controls src="video.mp4#t=25,45"></video>
```

Formato de Audio y Video

Tipos de formatos

Existen diferentes tipos de formato según la compresión del audio. Es importante saber distinguir entre formato de archivo y *códec*. El *codec* codifica y decodifica los datos del audio mientras estos datos son archivados con un formato de audio específico. La mayoría de los formatos de archivo de audio públicamente documentados pueden ser creados con uno de dos o más codificadores o *codecs*. Aunque la mayoría de formatos de archivo de audio solo soportan un tipo de datos (creado con un *codec* de audio), un contenedor de formato de multimedia como MKV o AVI puede soportar múltiples tipos de datos de audio y vídeo.

Hay tres grupos principales de formatos de archivo de audio:

- Formatos de audio sin comprimir, como WAV, AIFF o AU
- Formatos sin pérdida (formato de audio comprimido sin pérdida) como FLAC, MPEG-4 SLS, MPEG-4 ALS, MPEG-4 DST, WavPack, Shorten, TTA, ATRAC, Apple Lossless WMA Lossless y ape Monkey's Audio
- Formatos con pérdida (algoritmo de compresión con pérdida) como MP3, Vorbis, Musepack, AAC, WMA y Opus

Formatos de audio sin comprimir

Hay un formato principal sin comprimir, PCM, que normalmente esta archivado como .wav en windows y .aiff en MAC. WAV y AIFF son formatos flexibles creados para almacenar varias combinaciones de frecuencia de muestreo o tasa de bits, esto los hacen adecuados para

archivar grabaciones originales. Existe otro tipo de archivo llamado CDA (audio CD Track) que es un archivo pequeño que sirve como acceso directo a parte de los datos de un CD. El formato AIFF está basado en el formato IFF, mientras que el formato WAV está basado en el formato RIFF, que realmente son muy diferentes. BWF (Broadcast Format) es el formato de audio sin comprimir creado por la Unión Europea de Radiodifusión como sucesor a WAV y permite el almacenamiento de meta-datos en el archivo. Este formato es principalmente usado por muchos programas profesionales de edición de audio en las industrias de televisión y cine. Archivos BWF contienen una referencia de timestamp estandarizado que permite sincronizar fácilmente con un elemento de foto separado. Stand-alone, Grabadoras multi-pista de dispositivos de audio, Zaxcom, HHB USA, Fostex y Aaton utilizan BWF como su formato preferido.

Formatos de audio comprimido sin pérdida (*Lossless*)

El formato sin pérdida requiere más tiempo de procesamiento que los formatos sin comprimir pero más eficiente en cuanto el espacio que ocupa. Formatos de audio sin comprimir codifican tanto audio como silencio con el mismo número de bits por unidad de tiempo. Codificar un minuto de silencio en un formato sin comprimir produce un archivo del mismo tamaño que codificar un archivo sin comprimir de un minuto de música de orquesta. Sin embargo en estos archivos la música ocupa un archivo ligeramente más pequeño y el silencio no ocupa casi nada. Estos formatos de compresión proporcionan un ratio de compresión de más o menos 2:1. El desarrollo de estos formatos intenta reducir el tiempo de procesamiento manteniendo un buen ratio de compresión.

Formatos de audio comprimido con pérdida

En este sistema de codificación se comprimen los datos descartando partes de ello. El proceso intenta minimizar la cantidad de datos que mantiene el archivo reduciendo su peso y por lo tanto su calidad. Realmente solo pierde los canales no audibles al oído humano, de tal modo que conservan gran parte de su calidad.

Formatos abiertos

GSM*]*

Diseñado para el uso de telefonía en Europa. gsm es un formato muy práctico para voces de calidad teléfono. Es un buen compromiso entre calidad y tamaño. Los archivos de wav pueden ser codificados con GSM. Recomendado este formato para voz. Fíjese que los archivos wav pueden también ser codificados con el códec gsm.

Dct

Es un formato de códec variable diseñado para dictado. Tiene información de encabezado de dictado y puede cifrarse (a menudo necesario por las leyes de confidencialidad médica)

VOX

Este formato es comúnmente utilizado para el *codec* ADPCM Dialógico (Adaptive differential pulse code modulation). Similar a otros formatos ADCPM comprime a 4 bits. Los archivos vox son similares a archivos wav, salvo que no contienen información sobre el archivo, de modo que la frecuencia de muestreo y el número de canales debe ser especificado para reproducir un archivo vox.

SMAF

Es un formato de audio creado por Yamaha cor. Utilizado en dispositivos móviles. La extensión de los archivos para este formato es .mmf.

Formatos abiertos libres

Aiff

Audio Interchange File Format (AIFF) es un estándar de formato de audio usado para almacenar datos de sonido en computadoras personales. El formato fue codesarrollado por Apple Inc.

AU

El formato de archivo estándar utilizado por Sun, Unix y Java. El audio de archivos au puede ser PCM o comprimido con a-law o G.729.

Flac

flac (free lossless audio codec) es un *códec* de compresión sin pérdida. Puede imaginar la compresión sin pérdida como un archivo zip pero para audio. Si comprime un archivo PCM en flac y luego lo restaura otra vez, entonces va a tener una copia perfecta del original. (Todos los otros códecs tratados aquí son con pérdida, lo cual significa que se pierde una pequeña parte de la calidad). El costo de esta falta de pérdida es que la relación de compresión no es buena. Pero recomendamos flac para almacenamiento de archivos PCM donde la calidad es importante (p.ej. usarlo para la difusión o música).

Alac

Alac es el acrónimo de *Apple Lossless Audio Codec*. Es el códec favorito para las transmisiones de Apple Music. Las diferencias en cuanto a eficiencia o calidad de sonido respecto al formato Flac no son muy notables.

Ogg

Es un formato de archivo contenedor abierto y libre compatible con una variedad de códecs, el más popular de ellos es el códec de audio Vorbis. Los archivos Vorbis son generalmente comparados con los archivos MP3 en términos de calidad. Pero el simple hecho que los mp3 sean ampliamente admitidos, dificulta la recomendación de archivos ogg.

Vorbis

Códec de audio digital general con pérdidas, libre desarrollado por la Fundación Xiph.Org, que utiliza el formato de archivo de audio o contenedor.

Opus

Códec de audio digital con pérdidas, muy versátil, abierto y libre de patentes (Nueva licencia BSD), que utiliza el formato de archivo de audio o contenedor Ogg.

Mpc

Musepack o MPC (anteriormente conocido como MPEGplus o MP+) es un formato código abierto, específicamente optimizado para la compresión transparente de audio estéreo a una velocidad de 160-180 bits/s.

RAW

Un archivo raw puede contener audio de cualquier codec aunque suele ser utilizado con datos de audio PCM. Suele ser utilizado solo en pruebas técnicas.

TTA

TTA (the true audio) un *codec* de audio sin pérdida en tiempo real. Sin compresión. PCA

Formatos propietarios

mp3

El formato MPEG Layer-3 es el más popular para descargar y almacenar música. Mediante la eliminación de partes del archivo de audio que son esencialmente inaudibles, los archivos mp3

se comprimen aproximadamente a una décima parte del tamaño de un archivo PCM equivalente manteniendo una buena calidad de audio.

AAC

AAC (Advanced Audio Coding) este formato está basado en MPEG2 y MPEG4. Los archivos acc suelen ser contenedores ADTs o ADIF.

mp4

mp4 o m4a, MPEG-4 audio más a menudo AAC pero a veces MP2/MP3, MPEG-4 SLS, CELP, HVXC y otros tipos de objetos de audio pueden ser definidos en MPEG-4 audio. Este tipo de archivos no necesita reproductor, el mismo archivo incorpora uno para ejecutarse. Los archivos ocupan un 30% menos que los MP3 (la compresión es de 16:1). Como dato interesante, las canciones en MP4 solo son distribuidas con previa autorización del artista. Los formatos que componen un MP4 estándar son: MP3, AAC y Apple Lossless (sonido), MPEG-4, MPEG-4 y MPEG (video), JPG y PNG (imagen) y XMT y BT (subtítulos).

Wma

(windows media audio) este formato fue creado por Microsoft y está diseñado con habilidades de gestión de derechos digitales para protegerlo de copia.

Wav

formato de archivo de audio estándar usado mayormente en equipos con Windows. Comúnmente usado para almacenar archivos sin comprimir (PCM), de sonido con calidad de CD, lo cual significa que pueden ser grandes en tamaño - alrededor de 10MB por minuto de música. Es poco conocido que los archivos wav pueden ser también codificados con una variedad de códecs para reducir el tamaño del archivo (por ejemplo los códecs GSM o mp3). Una lista de códecs comunes de archivos wave pueden ser encontrados aquí. Muestra de archivo .wav.

Atrac

atrac (.wav), El estilo antiguo de formato Sony ATRAC. Siempre contiene una extensión de formato.wav. Para abrir estos formatos hay que instalar unos drivers ATRAC3.

Ra & rm

Un formato RealAudio diseñado para el *streaming* de audio por Internet.

Ram*]*

Un archivo de texto que contiene un enlace a una página web donde el archivo *RealAudio* está almacenado. Archivos ram no contienen audio.

Dss*[*

dss (digital speech standard) Es un formato de propiedad de la corporación Olympus. Es relativamente viejo y su *codec* es mediocre.

Dvf

Un formato de Sony para archivos de voz comprimidos, normalmente utilizado en grabadoras de dictado.

IVS

Un formato desarrollado por 3D solar UK ltd., con gestión de derecho digital utilizado para descargar música de su tienda digital Tronme y en su reproductor interactivo de música y vídeo.

m4p

Una versión de ACC en mp4 desarrollada por Apple con gestión de derecho digital para la utilización de descargas de la tienda de Itunes.

Iklax

Es un formato de audio digital multi pista que permite varias acciones en datos musicales como arreglos de volumen y mezclas.audio

MIDI

Se trata de un protocolo de comunicación serial estándar que permite a los computadores y otros dispositivos musicales electrónicos comunicarse y compartir información para la generación de sonido

Etiqueta iframe. Atributo

Si quieres añadir un banner dentro de una página web, la mejor manera de hacerlo es a través de un iframe. Se trata de un marco en el que defines el tamaño y haces una llamada a otra página que deseas mostrar, de modo que esta última se visualiza a través de esa pequeña ventana que has abierto.

Añade un iframe en tu web

En el caso de que, por ejemplo, hayas convertido una animación en Flash a Html5, tal y como mostramos en el artículo Convierte archivos Flash a HTML5, habrás comprobado que el código que se genera es demasiado largo y complejo cómo para añadirlo directamente sobre la página donde lo quieres incluir. De hacerlo así, el peso de la página aumentaría mucho, además de resultar, seguramente, mucho más difícil de indexar por los motores de búsqueda.

La mejor solución en este caso, es añadir un iframe, es decir un marco o una pequeña “ventana” dentro de la página que va a contener, a su vez, otra página html, en este caso, la animación en Html5. El código para añadir un iframe dentro de una página web es muy sencillo. Basta con copiar el siguiente código en el punto donde lo quieras insertar:

```
<iframe src="nombre.html" width="xxx" height="xxx" >

Texto alternativo para los usuarios que no ven iFrames.

</iframe>
```

Con esto, quedaría resuelto tu problema. Se trata de un ejemplo muy sencillo en el que hemos añadido el nombre de la página que quieres visualizar y le hemos asignado un ancho y un alto a través de los atributos width y height, respectivamente.

Si no tienes muy claro qué dimensiones debe tener el iframe para que se muestre correctamente la animación generada en Html5, puedes consultarlo en el código fuente de este documento. Aunque es bastante completo, debes saltar el código del script y fijarte casi al final, donde se encuentra la etiqueta <body> desde donde se hace la llamada al mismo.

Por tanto, sólo debes añadir el nombre de la página web e indicar el tamaño del marco, que ya sabes. En este ejemplo:

<iframe src="automatismo.html" width="564" height="227" >

Texto alternativo para los usuarios que no ven iFrames.

</iframe>

Como puedes comprobar, hemos añadido un pequeño texto entre medias de las etiquetas que abren y cierran el iframe. Se trata de un texto alternativo que puedes editar a tu antojo, y que aparecerá en aquellos casos en la que la versión de navegador desde el que se visualiza la página no sea compatible con este elemento. También puedes sustituir el texto por una imagen estática.

Una vez que termines, puedes hacer una comprobación visualizando la página en el navegador. En este ejemplo, puedes visualizar que la página con la animación realizada en Html5 se ha añadido sin problemas, dentro de la principal.

Atributos de un iframe

Como has podido comprobar, el código necesario para generar un iframe es muy sencillo. No obstante, dispones de algunos atributos con los que puedes conseguir hacerlo más atractivo.

Algunos de ellos son:

Frameborder= “1/0”: añade un borde al marco

Marginwidth “% píxeles”: añade un margen en los laterales en el porcentaje indicado

Marginheight “% píxeles”: añade un margen en el extremo superior e inferior del marco, en el porcentaje indicado

Scrolling=”yes/no/auto”: genera una barra de scroll en el caso de que sea necesario para mostrar el contenido de la página dentro del iframe.

Align="left/center/right": ajusta el contenido del iframe a un lado u otro de la ventana.

Puedes añadir estos atributos junto con la etiqueta que te hemos mostrado. Por ejemplo, si quieres añadir un pequeño borde alrededor del marco, puedes incluir la propiedad: frameborder= “1”.

Por último, comprueba en el navegador que, efectivamente, se ha añadido el borde que acabas de indicar.

```
<iframe src="automatismo.html" width="564" height="227" style="position
:absolute align="left" frameborder="1" framespacing="0" scrolling="no"
  border="0" >

Texto alternativo para los usuarios que no ven iFrames.

</iframe>
```

Concepto de tabla

Las tablas HTML están pensadas para utilizarse con datos tabulados. Por desgracia, mucha gente utiliza las tablas HTML para hacer compaginaciones de páginas web. Por ejemplo, una fila para contener la cabecera, una fila para contener las columnas de contenido, una fila para contener el pie de página, etc. Puede encontrar más detalles y un ejemplo en Diseños de página en nuestro Módulo de aprendizaje de accesibilidad. Se solía hacer este uso de las tablas porque la compatibilidad CSS entre navegadores solía ser terrible. Los diseños de tablas son mucho menos comunes hoy en día, pero aún se pueden ver en algunos rincones de la web.

Ya hemos hablado bastante sobre la teoría de las tablas, así que veamos un ejemplo práctico y construyamos una tabla simple.

1. En primer lugar, haz una copia local de blank-template.html y minimal-table.css en un directorio nuevo de tu ordenador.
2. El contenido de cada tabla está delimitado entre estas dos etiquetas: **<table></table>**. Añádelas al cuerpo de tu código HTML.
3. El contenedor más pequeño dentro de una tabla es una celda, que se crea con un elemento **<td>** ('td' significa 'table data', *datos de tabla*). Añade lo siguiente dentro de tus etiquetas de tabla:

```
<td>Hola, soy tu primera celda.</td>
```

Si quieres una fila de cuatro celdas, tienes que copiar estas etiquetas tres veces. Actualiza el contenido de la tabla para que se vea así:

```
<td>Hola, soy tu primera celda.</td>

<td>Soy tu segunda celda.</td>

<td>Soy tu tercera celda.</td>

<td>Soy tu cuarta celda.</td>
```

Como verás, las celdas no se colocan una debajo de la otra, sino que se alinean automáticamente entre sí en la misma fila. Cada elemento <td> crea una sola celda, y juntas forman la primera fila. Cada celda que agregamos hace crecer la fila.

Para detener el crecimiento de esta fila y comenzar a colocar las celdas posteriores en una segunda fila, necesitamos usar el elemento **<tr>** ('tr' significa 'table raw', *fila de tabla*). Vamos a verlo en detalle.

Coloca las cuatro celdas que has creado dentro de las etiquetas <tr>, de esta forma:

```
<tr>
  <td>Hola, soy tu primera celda.</td>
  <td>Soy tu segunda celda.</td>
  <td>Soy tu tercera celda.</td>
  <td>Soy tu cuarta celda.</td>
</tr>
```

Añadir encabezados con elementos <th>

Ahora nos vamos a centrar en los encabezados de tabla: celdas especiales que van al comienzo de una fila o columna y definen el tipo de datos que contiene esa fila o columna (por ejemplo, observa las celdas «Propietario» y «Edad» en el primer ejemplo que se muestra en este artículo). Para ilustrar por qué son útiles, echa un vistazo al ejemplo de tabla siguiente. En primer lugar, el código fuente:

```
<table>
  <tr>
    <td>&amp;nbsp;</td>
    <td>Knocky</td>
    <td>Flor</td>
    <td>Ella</td>
    <td>Juan</td>
  </tr>
  <tr>
    <td>Raza</td>
    <td>Jack Russell</td>
    <td>Caniche</td>
    <td>Perro callejero</td>
    <td>Cocker Spaniel</td>
  </tr>
  <tr>
    <td>Edad</td>
    <td>16</td>
    <td>9</td>
    <td>10</td>
    <td>5</td>
  </tr>
  <tr>
    <td>Propietario</td>
    <td>Suegra</td>
    <td>Yo</td>
    <td>Yo</td>
    <td>Cuñada</td>
  </tr>
  <tr>
    <td>Hábitos alimentarios</td>
    <td>Come las sobras de todos</td>

<td>Mordisquea la comida</td>
    <td>Come en abundancia</td>

<td>Come hasta que revienta</td>
  </tr>
</table>
```

El problema aquí es que, si bien puedes distinguir lo que sucede, no es tan fácil cruzar datos de referencia. Sería mucho mejor si los encabezados de las columnas y las filas se destacasen de alguna manera.

1. Primero, haz una copia local de nuestros archivos dogs-table.html y minimal-table.css en un directorio nuevo de tu ordenador. El HTML contiene el mismo ejemplo sobre perros que viste arriba.
2. Para reconocer los encabezados de la tabla como encabezados, tanto visual como semánticamente, puedes usar el elemento **<th>** ('th' significa 'table header', *encabezado de tabla*). Funciona exactamente igual que un <td>, excepto que denota un encabezado, no una celda normal. Entra en el código HTML, y cambiar todos los elementos <td> que delimitan los encabezados de tabla por elementos <th>.
3. Guarda tu HTML y cárgalo en un navegador. Los encabezados deberían verse como tal.

Etiqueta summary. Funcionamiento y atributos

El elemento summary representa un resumen, título o etiqueta para los contenidos de un elemento details . La descripción provista por este elemento será mostrada encima de los contenidos del elemento details para el cual ha sido declarado. El soporte provisto por los navegadores para details y summary es incompleto.

En el siguiente ejemplo, usaremos un elemento summary con el fin de proveer un título para los contenidos de un elemento details. El texto en este título describe apropiada y concisamente al resto de los contenidos del elemento details.

p>Cosmos: un viaje personal es una serie documental de divulgación científica escrita por Carl Sagan, Ann Druyan y Steven Soter (con Sagan como guionista principal y presentador), cuyos objetivos fundamentales fueron: difundir la historia de la astronomía y de la ciencia, así como sobre el origen de la vida; concienciar sobre el lugar que ocupa nuestra especie y nuestro planeta en el universo, y presentar las modernas visiones de la cosmología y las últimas noticias de la exploración espacial, y en particular, las misiones Voyager. <details> <summary>Carl Sagan</summary> Carl Edward Sagan fue un astrónomo, astrofísico, cosmólogo, escritor y divulgador científico estadounidense. Sagan publicó numerosos artículos científicos y comunicaciones1 y fue autor, coautor o editor de más de una veintena de libros. Defensor del pensamiento escéptico científico y del método científico, fue también pionero de la exobiología, promotor de la búsqueda de inteligencia extraterrestre a través del Proyecto SETI e impulsó el envío de mensajes a bordo de sondas espaciales, destinados a informar a posibles civilizaciones extraterrestres acerca de la cultura humana. </details>

Que es contenedor wrapper

La palabra inglesa *wrapper* designa un embalaje o envoltorio. En el contexto del *software*, este término hace referencia a programas o códigos que rodean otros componentes de programa.

Los *wrappers* pueden utilizarse por diversos motivos: a menudo se usan para mejorar la **compatibilidad** o **interoperabilidad** entre diferentes estructuras de software o para poder representar algo a nivel visual, p.ej., los *wrappers* HTML o CSS. Pueden ser *wrappers* diferentes componentes de software, productos de *software* independientes, arquitecturas de *software*, clases de programación orientada a objetos o *frameworks*.

Si dentro de un programa hay que usar funciones o bloques de programas en otro lenguaje de programación, estos elementos pueden "envolverse". El programa principal **solo se comunica con el *wrapper*** y este se encarga de transmitir los comandos al programa envuelto y de devolver los resultados al programa principal. El *wrapper* es el único componente del proceso que está en contacto directo con ambos elementos de software.

En el mundo de la programación y el desarrollo de *software*, los *wrappers* cuentan con una infinidad de usos. En los siguientes ejemplos, te mostramos qué es un *wrapper*, cómo funciona exactamente y qué tareas asume.

Wrappers en la programación orientada a objetos

En la programación orientada a objetos, se usan diferentes **patrones estructurales** que funcionan siempre de la misma manera, sin importar el lenguaje de programación que se utilice. Los patrones de diseño, adaptador y decorador pertenecen a la categoría de patrones estructurales y se consideran *wrappers*.

Un **adaptador** envuelve a dos interfaces incompatibles entre diferentes clases. Al traducir una interfaz a la otra, las clases ya pueden volver a comunicarse entre sí. Esto es muy importante cuando queremos seguir usando clases o bibliotecas enteras de clases en proyectos nuevos. Estas bibliotecas usan interfaces estandarizadas y unívocas que no deben modificarse ya que deben ser compatibles con un gran número de programas. El *wrapper*, el adaptador en este caso, es el eslabón intermedio determinante para la comunicación.

Un **decorador** permite ampliar una clase con funciones adicionales sin modificarla. Para el objeto de programa consultante, el decorador presenta la misma interfaz que la clase original. Así, el objeto que lo consulta no requiere modificación alguna. En su función de *wrapper*, el decorador transmite las consultas a la clase. Las funciones nuevas, no incluidas en la clase, son procesadas directamente por el decorador. A continuación, este devuelve los resultados de manera que el objeto consultante los perciba como resultados de la clase decorada.

Wrappers para diseñar documentos HTML

Es muy común usar *wrappers* en el (cambio del) diseño de páginas web HTML y CSS. Sin *wrappers*, incluso los cambios más pequeños, por ejemplo, cambiar los espacios de los márgenes de una ventana del navegador, implicarían tener que modificar varias hojas de estilo y revisar al final que todas siguen siendo compatibles.

Es mucho más sencillo meter todo el contenido de la página en un **contenedor DIV**, tal y como muestra el siguiente ejemplo:

```
<html>
  <head>
  ...
  </head>
  <body>
    <div class="wrapper">
```

```
    …
    </div>
    </body>
</html>
```

En el contenedor *wrapper* se encuentra el contenido real de la página.

En el archivo CSS correspondiente se define el *wrapper* como hoja de estilo:

```
body {
        margin: 0;
        padding: 0
}
.wrapper {
        width: 500px;
        margin: 25px auto;
}
```

En este ejemplo, se usa el parámetro *width:* para atribuir un ancho de 500 píxeles al contenedor. Los márgenes superior e inferior se fijan mediante el parámetro *margin:* a un valor de 25 píxeles. Los márgenes izquierdo y derecho derivan automáticamente del ancho de la ventana del navegador y el ancho del contenedor.

Basta con modificar el *wrapper* para modificar los márgenes laterales de la página sin necesidad de intervenir más en el código HTML o CSS.

Que es Header

Cuando un usuario entra a una página web, lo primero que verá será el header o cabecera web. Aunque se trata de un término de diseño web y programación, todos los profesionales del marketing y emprendedores deberían familiarizarse con él, ya que un buen header influirá en el posicionamiento en buscadores.
En este post te detallaremos qué es un header y qué función tiene, así como también te mostraremos algunos ejemplos para que te inspires.

- ¿Quieres saber cómo mejorar tus landing pages para captar leads o incrementar tus ventas? Clica aquí y descárgate **nuestro curso gratuito.**

¿Qué es un header y para qué sirve?
El **header o cabecera de una página web** es el término que hace referencia a la **parte superior de un sitio web**. Al ser lo primero que el usuario visualiza, debes tener en cuenta dos cosas. Primero, que los usuarios puedan navegar fácilmente por él. Y, segundo, ofrecer información relevante sobre tu identidad para conectar lo antes posible con tus visitantes.

A grandes rasgos, estos son los elementos que habitualmente forman la cabecera de una web. Algunos de estos son opcionales, pero otros son obligatorios:

- **Elementos de tu identidad corporativa**. Estos son el logo, el eslogan y el nombre de la empresa.

- **Un menú**. Gracias a este, el usuario podrá navegar por la web para encontrar lo que busca. Aquí deberán estar todas las secciones de la web.

- **Iconos de las redes sociales**. Aquellas donde tengas presencia. Algunas marcas ponen estos botones en el footer.

- **Información de contacto**. Puede ser un icono de teléfono o correo electrónico.
- **Un buscador**. A través de este el usuario puede introducir una palabra clave o relacionada a tu negocio para encontrar rápidamente algo en tu web.

Funciones de un header

Al igual que pasa en el footer, el header es un elemento que se repite en todas las páginas de un sitio web. Pero, ¿para qué sirve? La cabecera tiene, por un lado, la finalidad de transmitir al usuario quién hay detrás de esa marca. Y, por otro lado, ofrecer al usuario enlaces de otras páginas de la web que también son relevantes para facilitar su navegación por ella.
Por tanto, la cabecera de un sitio web tiene entre sus funciones mejorar la usabilidad de la web y ofrecer una buena experiencia al usuario; sin olvidar su función estética, que debe ir en línea con su identidad corporativa.

Ejemplos de una cabecera web

Existen muchos tipos de web y por ello podemos encontrar **diferentes tipos de cabeceras**. Por ejemplo, no es lo mismo la cabecera de un blog que la de un ecommerce o tienda online.

En un **blog** lo habitual es encontrarse, además del logo, el nombre y/o eslogan, las categorías, la página de contacto, los iconos de las redes sociales y la información de contacto.

En cambio, el **header de un ecommerce** es bien distinto. En este caso, además del logo, hay otros elementos que son imprescindibles como la home, la categoría de productos, mi cuenta, el carrito, área privada, entre otros.

Aquí tienes un ejemplo visual, muy común en páginas webs para que veas cómo se usa el elemento <header>. En él se incluye el logotipo, el nombre de la empresa y un formulario de búsqueda.

```
<header>
 <img src="../../../images/cyberclick-logo.png"
 <h1>El nombre de la compañía</h1>
 <form action="search.php">
   <input type="text">
   <input type="submit" value="Buscar">
 </form>
```

Que es contenido (content).

El elemento HTML <content> es usado dentro de un Shadow DOM como un insertion point . No está pensado para ser usado en HTML ordinario . Es usado con Web Components.

Nota: Aunque está presente en un draft inicial de las especificaciones e implementado en varios exploradores , este elemento ha sido removido en versiones posteriores a la especificación .

select

Una lista de selectores separada por comas . Estos tienen la misma sintaxis que los selectores de CSS . Seleccionan el contenido a insertar en lugar del elemento <content> .

Ejemplo

Aquí hay un ejemplo simple del uso del elemento <content> . Es un archivo HTML con todo lo necesario en el .

Nota: Para que este código funcione , el explorador en el que se muestre debe de soportar Web Components .

```
<html>
  <head></head>
  <body>
      <!-- The original content accessed by <content> -->
     <div>
         <h4>My Content Heading</h4>
         <p>My content text</p>
      </div>
      <script>
            // Get the <div> above.
            var myContent = document.querySelector('div');
           // Create a shadow DOM on the <div>
           var shadowroot = myContent.createShadowRoot();
          // Insert into the shadow DOM a new heading and
          // part of the original content: the <p> tag.
          shadowroot.innerHTML =
         '<h2>Inserted Heading</h2> <content select="p"></content>';
  </script>
  </body>
</html>
```

Que es menú (menú).

El elemento menú representa una barra de menú que consiste en una lista de comandos representada por una lista no ordenada de ítems (elementos li). Este elemento es simplemente una alternativa semántica al elemento ul que se utiliza para representar una lista no ordenada de comandos.

En el siguiente ejemplo, insertaremos un menú de barra de herramientas que pretende ser la barra de menú de un editor de texto. Los comandos listados en este menú no ejecutarán ninguna acción, pero el ejemplo será de todos modos suficiente para mostrar la funcionalidad del elemento menú. Como ha sido dicho anteriormente, cada ítem en el menú es representado por un ítem de lista (elemento li) conteniendo a un botón (elemento button).

Las funciones declaradas en el siguiente ejemplo no están diseñadas para funcionar. Este ejemplo apunta solamente a probar la funcionalidad provista por los navegadores para los menús.

```
<li><button onclick="nuevo()">Nuevo...</button></li>
<li><button onclick="abrir()">Abrir...</button></li>
<li><button onclick="guardar()">Guardar</button></li>
<li><button onclick="guardarcomo()">Guardar
como...</button></li> </menu>
```

Que es pie (footer).

El *Elemento HTML Footer* (<footer>) representa un pie de página para el contenido de sección más cercano o el elemento raíz de sección (p.e, su ancestro mas cercano <article>, <aside>, <nav>, <section>,<blockquote>, <body>, <details>, <fieldset>, <figure>, <td>). Un pie de página típicamente contiene información acerca del autor de la sección, datos de derechos de autor o enlaces a documentos relacionados.

<meta http-equiv="content-type" content="text/html; charset=utf-8"/>

Nota: Notas de uso:

Encierra la información acerca del autor en un elemento **<add**El *Elemento HTML Footer* (<footer>) representa un pie de página para el contenido de sección más cercano o el elemento raíz de sección (p.e, su ancestro mas cercano <article>, <aside>, <nav>, <section>,<blockquote>, <body>, <details>, <fieldset>, <figure>, <td>). Un pie de página típicamente contiene información acerca de el autor de la sección, datos de derechos de autor o enlaces a documentos relacionados.

<meta http-equiv="content-type" content="text/html; charset=utf-8"/>

Nota: Notas de uso:

- Encierra la información acerca del autor en un elemento <address> que puede ser incluido dentro del elemento <footer>.
- El elemento <footer> no es contenido de sección y en consecuencia no introduce una nueva sección en el esquema.

- ress> que puede ser incluido dentro del elemento <footer>.
- El elemento <footer> no es contenido de sección y en consecuencia no introduce una nueva sección en el esquema.

Que es lateral (sidebar).

El **Sidebar o barra latera**l es el espacio que está al lado del contenido de una página web (especialmente en un blog), puede estar colocado en diferentes zonas, aunque lo más habitual es encontrarlo o a la izquierda o a la derecha del contenido (normalmente a la derecha).

Un ejemplo fácil, si miras este post en un ordenador, es la barra que ves a la derecha de estas líneas y que comienza conmigo haciendo como si trabajara en un ordenador (seguramente estaba perdiendo tiempo en Instagram o algo.

```
<footer>
    <p>Mario Roberto Dominguez Diaz 2021</p>
</footer>
```

Utilidad del contenedor

Un contenedor div sirve principalmente para contener a otros elementos HTML. Aunque nos permite posicionar grupos y áreas, div no tiene su propio significado semántico en HTML. Su tarea principal es, por tanto, crear y delimitar áreas que luego puedan ser formateadas mediante CSS.

Esto también implica que div **ha dejado de jugar un papel importante** en HTML. Aunque el contenedor div se utilizaba con frecuencia en versiones anteriores, su uso se vio reducido drásticamente con la llegada de HTML5 y HTML 5.1. Solo se puede utilizar el contenedor div para la estructura si no se puede hacer uso de ningún otro elemento como, por ejemplo, *article, aside, header, main, nav* o *section*. Por ello, se recomienda buscar soluciones semánticamente más adecuadas antes de utilizar div en HTML. Como elemento de nivel de bloque, el contenedor div se inserta en un salto de línea anterior y otro posterior.

HTML div se utiliza principalmente para **dar formato con CSS a los elementos HTML** que aparezcan de manera simultánea, con el fin de agrupar diferentes bloques HTML y posicionarlos de manera diferenciada, al igual que para animar contenidos HTML y CSS con la ayuda de JavaScript. Con el siguiente ejemplo, puedes ver cómo se construye un contenedor div sin atributos:

```
<div>
<h2>Lista de ejemplos</h2>
<ul>
<li>Primer elemento de la lista</li>
<li>Segundo elemento de la lista</li>
<li>Tercer elemento de la lista</li>
<li>Cuarto elemento de la lista</li>
</ul>
</div>
```

En un principio, dentro de un contenedor div se puede colocar casi cualquier contenido fluido. En HTML, div no tiene ningún efecto sobre la presentación del contenido, sino que únicamente lo delimita.

Para que utilizamos estas etiquetas

<Div>

Una etiqueta div es una etiqueta que define las divisiones lógicas existentes en el contenido de una página web. Puede utilizar etiquetas div para centrar bloques de contenido, crear efectos de columna y crear diferentes áreas de color, entre otras posibilidades.

Si no está familiarizado con el uso de etiquetas div y hojas de estilos en cascada (CSS) para crear páginas web, puede crear un diseño CSS basado en uno de los diseños predefinidos que se suministran con Dreamweaver. Si no se siente cómodo utilizando código CSS pero sí con las tablas, también puede probar a utilizar tablas.

Ejemplo

```
<div>
<h2>Lista de ejemplos</h2>
<ul>
<li>Primer elemento de la lista</li>
<li>Segundo elemento de la lista</li>
<li>Tercer elemento de la lista</li>
<li>Cuarto elemento de la lista</li>
</ul>
</div>
```

<article>.

El Elemento de HTML <article> representa una composición auto-contenida en un documento, una página, una aplicación o en un sitio, que se quiere que sea distribuíble y/o reutilizable de manera independiente, por ejemplo, en la redifusión

Ejemplo

```
<article class="forecast">
   <h1>Weather forecast for Seattle</h1>
   <article class="day-forecast">
      <h2>03 March 2018</h2>
      <p>Rain.</p>
   </article>
   <article class="day-forecast">
      <h2>04 March 2018</h2>
      <p>Periods of rain.</p>
   </article>
   <article class="day-forecast">
      <h2>05 March 2018</h2>
      <p>Heavy rain.</p>
   </article>
</article>
```

<aside>.

El elemento HTML <aside> representa una sección de una página que consiste en contenido que está indirectamente relacionado con el contenido principal del documento.

```
<aside>
   <H3>Anuncio</H3>
```

```
        <P align="justify">Ventas para programas de computadoras que trabajan en ambiente
        web, programas desarrollados en php con base de  datos MYSQL. Visite nuestro local
        ubicado en la ciudad de puerto plata o puede llamarnos al telefono 809-586-0000.</p>
    </asideE>
```

<details>.

El elemento HTML Details <details> es usado como un widget de revelación a través del cual el usuario puede obtener información adicional . Contenido dinámico , contenido de seccionamiento, contenido interactivo, contenido palpable

Ejemplo

```
<details>
  <summary>Some details</summary>
  <p>More info about the details.</p>
</details>
```

<footer>.

El Elemento HTML Footer (<footer>) representa un pie de página para el contenido de sección más cercano o el elemento raíz de sección en nuestra pagina web.

Ejemplo

```
<footer>
    <p>Todo derecho Reservado, Puerto Plata Rep. Dom. MMXXII</p>
</footer>
```

<header>.

El elemento de HTML Header (<header>) representa un grupo de ayudas introductorias o de navegación. Puede contener algunos elementos de encabezado, así como también un logo, un formulario de búsqueda, un nombre de autor y otros componentes.

Ejemplo

```
<header>
    <div>
      <!--<img src="c:\RobertoWeb\trebor.jpg"> -- >
      <video width="320" height="120" controls><source src="c:\robertoweb\logo.mp4"
      type="video/mp4"></video>
    </div>
      <nav>
        <ul>
      <li><a href="c:\RobertoWeb\inicio.html">Inicio</a></li>
      <li><a href="http://www.w3bai.com/es/js/js_htmldom_events.html">Curso</a></li>
      <li><a href="https://trebor88.blogspot.com/">Blog Trebor</a></li>
      <li><a href="https:// www.tuparada.com/nombres/">Tuparada</a></li>
      </ul>
```

```
        </nav>
    </header>
```

<main>.

El elemento HTML <main> representa el contenido principal del <body> de un documento o aplicación. El área principal del contenido consiste en el contenido que está directamente relacionado, o se expande sobre el tema central de un documento o la funcionalidad central de una aplicación.

Ejemplo

```
<!-- other content -->
<main>
  <h1>Apples</h1>
  <p>The apple is the pomaceous fruit of the apple tree.</p>

  <article>
    <h2>Red Delicious</h2>
    <p>These bright red apples are the most common found in many
    supermarkets.</p>
    <p>... </p>
    <p>... </p>
  </article>

  <article>
    <h2>Granny Smith</h2>
    <p>These juicy, green apples make a great filling for
    apple pies.</p>
    <p>... </p>
    <p>... </p>
  </article>

</main>
<!-- other content -->
```

<mark>.

El Elemento HTML Mark <mark> representa un texto marcado o resaltado como referencia o anotación, debido a su relevancia o importancia en un contexto particular.

Ejemplo

```
<blockquote>
     It is a period of civil war. Rebel spaceships, striking from a
    hidden base, have won their first victory against the evil
    Galactic Empire. During the battle, <mark>Rebel spies managed
     to steal secret plans</mark> to the Empire's ultimate weapon,
     the DEATH STAR, an armored space station with enough power to
     destroy an entire planet.
</blockquote>
```

<nav>.

El elemento HTML <nav> representa una sección de una página cuyo propósito es proporcionar enlaces de navegación, ya sea dentro del documento actual o a otros documentos. Ejemplos comunes de secciones de navegación son menús, tablas de contenido e índices

Ejemplo

```
<nav class="menu">
 <ul>
  <li><a href="#">Inicio</a></li>
  <li><a href="#">Sobre nosotros</a></li>
  <li><a href="#">Contacto</a></li>
 </ul>
</nav>
```

<section>

El elemento de HTML section (<section>) representa una sección genérica de un documento. Sirve para determinar qué contenido corresponde a qué parte de un esquema. Piensa en el esquema como en el índice de contenido de un libro; un tema común y subsecciones relacionadas.

Ejemplo

Antes de HTML 5

```
<div>
  <h1>Encabezado</h1>
  <p>Un montón de contenido impresionante.</p>
</div>
```

Ahora Con HTML5

```
<section>
  <h1>Encabezado</h1>
  <p>Un montón de contenido impresionante.</p>
</section>
```

<time>.

El elemento HTML <time> representa un periodo específico en el tiempo. Puede incluir el atributo datetime para convertir las fechas en un formato interno legible por un ordenador, permitiendo mejores resultados en los motores de búsqueda o características personalizadas como recordatorios.

Ejemplo

```
<p>El concierto empieza a las <time>20:00</time>.</p>
```

Otro

```
<p>El concierto fué el <time
 datetime="2001-05-15T19:00">15 de Mayo</time>.</p>
```

<meta> viewport

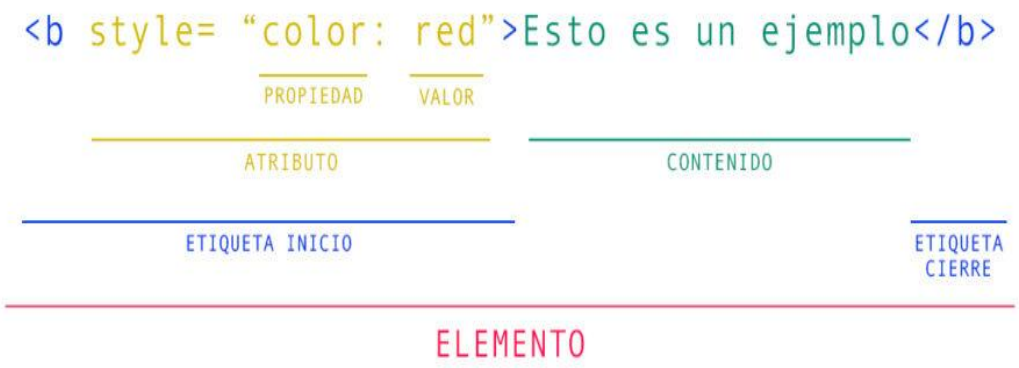
Como definición rápida, diremos que viewport podría traducirse como vista o ventana y nos sirve para definir qué área de pantalla está disponible al renderizar un documento, nivel de escalado y el zoom que debe mostrar inicialmente. Todo ello, con parámetros que le damos a la propia etiqueta META

```
<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">

<meta name="viewport" content="user-scalable=no, width=device-width, initial-scale=1">

<meta name="viewport" content="width=device-width,initial-scale=1.0">
```

Utilidad de las etiquetas

Antes de profundizar en este concepto, debemos aclarar que el código HTML hace referencia al lenguaje que se emplea para el desarrollo de páginas web. Dicho código se compone por diferentes etiquetas que el navegador interpreta y a través de las cuales se puede crear o modificar la apariencia de un documento.

Las **etiquetas HTML** son fragmentos de código que nos permiten crear elementos HTML. Los elementos son la estructura básica de HTML. Dichos elementos tienen dos propiedades básicas: atributos y contenido.



Un **elemento**, para que sea válido, generalmente tiene una etiqueta de inicio (<nombre-del-elemento>) y una etiqueta de cierre (</nombre-del-elemento>).

En este punto cabe destacar que los elementos **no** son etiquetas. Los elementos están representados por etiquetas, pero, generalmente por error, se les considera lo mismo. Este contenido es lo que se verá afectado por la funcionalidad o el significado del elemento.
Ejemplo: ****Esto es un ejemplo****
Como podemos ver en el ejemplo, existe la **etiqueta de apertura** (“”) y la **etiqueta de cierre** (“”). El texto que hay en medio de ambas etiquetas, “Esto es un ejemplo”, se conoce como **contenido** del elemento.
A parte de las etiquetas y el contenido, un elemento puede tener **atributos**. Los atributos son la forma en que los creadores o desarrolladores web definen las **propiedades** de un elemento.
Los atributos deben insertarse en forma de lista y separados por espacios, siempre dentro de la etiqueta para que sea válido. El atributo debe añadirse después del nombre del elemento o de la etiqueta html y precedidos por un espacio.

Cada uno de estos elementos está compuesto por un nombre, el signo igual (=) y el **valor** o función (a veces entre comillas).
Ejemplo: **<b style=”color: red”>** Esto es un ejemplo ****
Por último, los **eventos** son diseñados para permitir a los desarrolladores web crear diferentes scripts en sus páginas como forma de respuesta a las interacciones de los usuarios.
La función de un evento es asociar una acción, realizada por el usuario o por el propio sistema, con una función o script. Por ejemplo cuando el usuario pone el cursor del ratón sobre un elemento y éste cambia de color.