

Manual de HTML



Ruben Alvarez
Miguel Angel Alvarez
Christian Santalucía



desarrolloweb.com/manuales/manual-firebase.html

Introducción: Manual de HTML

El Manual de HTML te enseña a trabajar con el lenguaje de marcación que sirve para construir las páginas web. HTML es el primer paso que debería completar cualquier persona que quiera dedicarse al desarrollo de web en general. Es además un conocimiento recomendado para cualquier persona que trabaje en el medio Internet.

Este es un manual con bastante detalle, que empieza en el conocimiento más básico y recorre cada uno de los elementos que se pueden usar para construir todo tipo de contenido en una web, incluso formularios, tablas, etc.

Hemos actualizado este manual de HTML en 2016, eliminando las partes que ya se han quedado en desuso, así como para incorporar nuevas informaciones importantes a día de hoy, buenas prácticas aconsejables, etc.

Encuentras este manual online en:

<http://desarrolloweb.com/manuales/manual-html.html>

Autores del manual

Las siguientes personas han participado como autores escribiendo artículos de este manual.

Rubén Alvarez

Rubén es doctor en química y programador aficionado con experiencia en PHP.



Miguel Angel Alvarez

Miguel es fundador de DesarrolloWeb.com y la plataforma de formación online EscuelaIT. Comenzó en el mundo del desarrollo web en el año 1997, transformando su hobby en su trabajo.



Christian Santalucía

Christian fue redactor en DesarrolloWeb.com durante los años 2002 a 2003. Es desarrollador frontend y aficionado a experimentar con nuevas tecnologías.

Introducción a HTML

Introducción al manual de HTML y al lenguaje de modelado de páginas web. Veremos qué es HTML y las primeras nociones que nos ayudarán a realizar las primeras pruebas de creación de una página web sencilla.

Prólogo al manual de HTML

Cuál es el enfoque y contenido de este Manual de HTML, a quién va dirigido, así como lecturas y materiales aconsejados para sacarle todo el provecho.

Bienvenidos al Manual de HTML de DesarrolloWeb. A través de varios capítulos vamos a descubrir el principal lenguaje utilizado para la creación de páginas web: **Hyper Text Markup Language**, más conocido mediante sus siglas **HTML**.

Puede que en un principio, el hecho de hablar de un lenguaje informático pare los pies a más de uno. No os asustéis, el HTML no deja de ser más que una forma un tanto peculiar de especificar el contenido de las páginas, indicando el texto y otros elementos como imágenes, tablas, listas, etc. Al final es de suma importancia el lenguaje porque es el medio con el cual se suministra el contenido a los navegadores y por tanto, si queremos comenzar a aprender a crear páginas web, forzosamente debemos comenzar por aquí.

Este manual queremos que sea lo más práctico posible. Iremos conociendo el lenguaje a través de numerosos ejemplos que te sugerimos realizar por ti mismo para asimilar mejor los conocimientos. Pero antes de entrar en materia, permítenos recomendarte la lectura de de nuestro manual [Publicar en Internet](#), en el cual se habla de una manera muy general sobre el proceso de diseñar y publicar una página web. El mencionado manual también os dejara bien claro lo que aporta HTML dentro del contexto de la creación de una página web, trata sobre editores, programas para subir archivos al servidor, etc.

El público al que va enfocado este manual es a todos aquellos que, con conocimientos mínimos de informática, desean hacer mundialmente público un mensaje, una idea o una información usando para ello el medio más práctico, económico y actual: Internet.

Qué necesitas para aprender HTML

Lo que necesitáis como base para llevar a buen término el aprendizaje de HTML es lo siguiente:

- Saber escribir con un teclado
- Saber manejar un ratón
- Tener ganas de aprender

Puede parecer una broma el listado anterior de requisitos, pero realmente queremos remarcar que cualquier persona que sepa manejar un ordenador tiene los conocimientos básicos para aprender HTML.

Obviamente, en lo que respecta al trabajo con un ordenador, debemos saber también a abrir programas,

editar un archivo con texto plano, guardar nuestros archivos dentro de alguna carpeta y ejecutarlos con un doble clic. Estamos seguros que si has llegado a este manual sabrás realizar todo este tipo de tareas básicas.

Que aprenderás en este manual

Si le pones un poco de ganas y sigues este manual hasta el final, tendrás las siguientes habilidades o conocimientos:

- Identificar qué se debe hacer con HTML y qué no.
- Capacidad para crear y publicar vuestro propio sitio web con un mínimo de calidad.
- Conocimientos de todo tipo sobre las tecnologías y herramientas empleadas en el ámbito de la Red.

Quizás aquí comience una bonita historia y sirva como primer paso para toda una serie de experiencias y aprendizajes, no solo de HTML, sino también de muchos otros lenguajes y tecnologías que están relacionadas con el desarrollo de sitios web. Estamos seguros que para muchos se convertirá en una afición que puede derivar en pasión y terminar, en algunos casos, siendo un vicio o un oficio. Pensar que todos los profesionales del desarrollo en Internet han pasado por aquí y comenzado como vosotros con este, u otro, manual de HTML.

Referencias a otros contenidos que pueden resultar de interés

DesarrolloWeb.com está plagado de manuales útiles para realizar sitios web desde los más simples hasta los más complejos, sin embargo, en el punto en el que estás queremos recomendar algunas referencias que te pueden resultar de utilidad.

Para quien no sepa nada sobre crear una página web, y le gusta que le expliquen las cosas desde cero y de manera visual, recomendamos ver el [vídeo donde mostramos el proceso de creación de la primera página básica](#). Además, para complementar las explicaciones de este manual, también recomendamos el [videotutorial de HTML](#).

Finalmente, antes de comenzar con el temario, queremos daros una referencia importante a la [sección HTML a fondo](#), donde publicamos todos los contenidos que tienen que ver con HTML y donde encontrarás este y otros manuales relacionados con el lenguaje.

Revisión de 2016

Estamos revisando el manual en 2016. El texto original se escribió en 2001 y aunque el HTML en sí no ha sufrido muchas variaciones, es importante revisar el enfoque del texto. Queremos que las personas que comiencen a leer en el día de hoy tengan una información fiel a las costumbres y buenas prácticas a la hora de usar este lenguaje. En esta revisión estamos seguros que agregaremos mayor valor al manual, ampliando las informaciones, pero también eliminando algunas cosas que han quedado en desuso. Esperamos que este esfuerzo sea de provecho todavía para muchas personas a lo largo del mundo.

Pasemos pues sin más preámbulos a ver de qué se trata el lenguaje HTML...

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 29/09/2016
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Introducción a HTML

Las primeras cosas que debes saber sobre HTML: historia, objetivos y demás conocimientos donde sentar las bases del manual.

HTML es el lenguaje con el que se escribe el contenido de las páginas web. Las páginas web pueden ser vistas por el usuario mediante un tipo de aplicación llamada cliente web o más comúnmente "navegador". Podemos decir por lo tanto que el HTML es el lenguaje usado por para especificar el contenido que los navegadores deben representar a la hora de mostrar una página web.

Este lenguaje nos permite aglutinar textos, imágenes, enlaces... y combinarlos a nuestro gusto. La ventaja del HTML a la hora de representar el contenido en un navegador, con respecto a otros formatos físicos como libros o revistas, es justamente la posibilidad de colocar referencias a otras páginas, por medio de los enlaces hipertexto.

Un poco de historia

El lenguaje HTML se creó en 1991. Tiene una historia realmente corta pero para su poca vida ha sufrido importantes cambios. Su padre es Tim Berners-Lee que lo diseñó con objetivos divulgativos. Inicialmente no se pensó que la web llegaría a ser un área de ocio con carácter multimedia, de modo que, nació sin dar respuesta a todos los posibles usos que se le iba a dar y a todos los colectivos de gente que lo utilizarían en un futuro. Sin embargo, pese a esta deficiente planificación, si que se han ido incorporando modificaciones con el tiempo, agregando nuevas características para cubrir las nuevas necesidades.

Como hemos dicho, los programas que leen y presentan las páginas web a los usuarios se llaman navegadores. Éstos son los responsables de interpretar el HTML y "pintar" una página tal como ellos entiendan que deben hacer. Sin embargo, esas reglas de representación no son subjetivas de cada fabricante del navegador, sino que existe una organización llamada W3C que se encarga de definir el estándar que todos deben seguir a la hora de escribir e interpretar el HTML. Estos estándares del HTML se conocen como "Especificaciones", las cuales han ido apareciendo en el tiempo. El HTML5 es el último estándar en la actualidad.

Históricamente los navegadores, además de la propia comunidad de usuarios, han sido los mayores impulsores de los cambios ocurridos en el lenguaje. Una vez detectada la necesidad es el W3C el que crea el estándar y marca una dirección que todos deben seguir.

Los navegadores y sus problemas

El conflicto generado por los navegadores es debido a su diversidad. Existen multitud de navegadores o clientes web presentes en el mercado los cuales muchas veces no son capaces de interpretar un mismo código de una manera unificada. Esto obliga al desarrollador a, una vez creada su página, comprobar que esta puede ser leída satisfactoriamente por todos los navegadores, o al menos, los más utilizados. Cuando surgen problemas de interpretación, queda de parte del desarrollador resolver el problema tirando de técnicas o conocimientos que él disponga.

Afortunadamente, en la actualidad las diferencias de interpretación de los navegadores con respecto a un

mismo código HTML son mínimos, pero en el pasado los desarrolladores tenían que emplear mucho tiempo en remar contracorriente para solucionarlos. Sin embargo, quedan todavía muchos usuarios que navegan con sistemas anticuados, ya sea por falta de interés para actualizarse, conocimientos, o por disponer ordenadores muy antiguos.

Pero no todo ha sido malo por parte de los navegadores. Ellos también son los responsables de introducir nuevas etiquetas en el uso común del día a día, que se han ido incorporando al estándar HTML en sucesivas versiones. Aunque antes de estandarizarse esas etiquetas era común que cada navegador crease su etiqueta propietaria para resolver la misma necesidad, lo que obligaba a los desarrolladores a repetir código o incluso a hacer versiones de páginas diferentes para navegadores. Con todo esto no queremos asustar a nadie y volvemos a repetir que las diferencias en la actualidad son mínimas, pero sí deseamos que quede clara la necesidad de la estandarización creada por el W3C, responsable de marcar una pauta que actualmente cumplen todos los navegadores modernos de manera bastante fiel.

Los lenguajes de la web

HTML no está solo como único lenguaje para crear la web, aunque en un principio sí que era así. Su evolución tan anárquica ha supuesto toda una serie de inconvenientes y deficiencias que han debido ser superados con la introducción de otras tecnologías accesorias capaces de organizar, optimizar y automatizar el funcionamiento de las webs. Ejemplos que pueden sonaros son las [CSS](#), o [JavaScript](#). Veremos más adelante en qué consisten algunas de ellas.

Lo que es importante para el desarrollador es conocer el enfoque de cada lenguaje, para saber cuál es la manera correcta de utilizarlo y cómo se complementan los unos a los otros. No es necesario que se sea experto en todos ellos, pero sí saber qué cosas se deberían hacer con cada cual, para no cometer errores que deriven en una mala interpretación por parte de los navegadores. Así mismo tenemos que pensar que no todas las personas van a acceder a una web a través de un ordenador, sino también de un teléfono o de navegadores especializados para personas con discapacidades, por ejemplo para ciegos. Es por ello que es importante escribir correctamente los lenguajes, respetando los estándares y así cada navegador podrá hacer su mejor papel para representar la pagina lo más correctamente posible.

En este manual queremos incidir mucho en este detalle, la correcta utilización del HTML: escribir el contenido, para que nuestro trabajo sea lo más adecuado y de elevada calidad.

Los editores de HTML

Además del navegador necesario para ver los resultados de nuestro trabajo, necesitamos evidentemente otra herramienta capaz de crear la página en sí. Un archivo HTML (una página) no es más que un texto plano al que le colocamos extensión ".html". Es por ello que para programar en HTML necesitamos un editor de texto.

Nota: En 2001 cuando escribimos este manual por primera vez recomendábamos probar a comenzar con el [Bloc de notas](#) que viene con Windows. El motivo es que es un programa tan simple que nos permite centrarnos simplemente en el HTML, eliminando todo tipo de ayudas. Hoy preferimos recomendar otras alternativas, aunque hemos dejado esta nota por motivos nostálgicos.

Es recomendable usar un editor de textos sencillo, de texto plano. Queremos remarcar que nunca se debe usar el tipo de editor de textos que se usan para escribir documentos, cartas, trabajos para el colegio, como Wordpad o Microsoft Word, pues colocan su propio código especial al guardar los documentos y HTML es **únicamente texto plano**, con lo que podremos tener problemas. Otro tipo de editor que tampoco recomendamos es DreamWeaver, un editor que te escribe el HTML a base de tocar botones, negritas, listas, crear tabla, etc. Para aprender es extremadamente recomendable que se use un programa que te permita escribir el código en crudo, así no tendrás problemas en el futuro.

El tipo de editores que recomendamos son aquellos específicos para la edición de código, los cuales están pensados para facilitar los procesos de la programación y de la escritura de código plano como el del lenguaje HTML. Existen infinidad de editores de código interesantes, que nos aportan más o menos facilidades y que nos permiten aumentar nuestra productividad. No obstante, es aconsejable en un principio utilizar una herramienta lo más sencilla posible para poder prestar la máxima atención a nuestro código y familiarizarnos lo antes posible con él. Siempre tendremos tiempo más adelante de pasarnos a editores más versátiles con la consiguiente ganancia de tiempo.

No es posible decir a nadie el editor que debe de usar, porque cada uno tendrá sus preferencias. No obstante, en 2016 y para las personas que están comenzando nosotros recomendamos:

- [Atom](#)
- [Brackets](#)

Igual alguna persona que comience puede pensar que tienen muchas opciones, pero realmente podemos comenzar con lo básico, crear nuevos archivos, editar el código, guardarlos en nuestro disco duro... y punto. Recomendamos estos editores porque están disponibles para todas las plataformas, Windows, Mac y Linux y porque son gratuitos para cualquier uso.

A continuación tienes algunas referencias para ampliar la información, aunque con estas recomendaciones tienes para empezar. Para una referencia más claro todo el tema de editores y los tipos que existen, visita los artículos:

- [Editores de HTML.](#)
- Si buscas en DesarrolloWeb, en el buscador por "editores", encontrarás muchas otras alternativas.

HTML es para escribir el contenido

Volveremos sobre este punto, pues es de vital importancia para entender y usar bien HTML. Graba e tu memoria que HTML es para especificar el contenido de las páginas web y no el aspecto que van a tener.

Cuando nos referimos al contenido queremos indicar párrafos, imágenes, listas, tablas y todo aquello que forma parte de "el qué". Nunca debemos usar HTML para definir cómo se debe de ver un contenido, si el texto debe tener color rojo, con una fuente mayor, o si se debe alinear a la derecha. Para especificar el aspecto que debe tener una web se usa un lenguaje complementario, llamado [CSS](#).

Por tanto, HTML sirve para decir qué contenido debe tener una página y CSS sirve para decir cómo se debe representar tal contenido, con qué estilo. Es fácil saltarse esta regla, porque en HTML existen diversas etiquetas (y atributos, de los que no hemos hablado todavía) que realmente están pensados para definir la presentación. Es una herencia de versiones pasadas del HTML y aunque en este manual se nos pueda saltar alguna vez alguna excepción, no debemos caer en la trampa de usar el HTML para definir cómo debe de

representarse un elemento en la página.

Conclusión

Como has visto, una página es un archivo donde está contenido el código HTML en forma de texto. Estos archivos tienen extensión .html o .htm (es indiferente cuál utilizar). De modo que cuando programemos en HTML lo haremos con un editor de textos y guardaremos nuestros trabajos con extensión .html, por ejemplo mipágina.html

Consejo: Utiliza siempre la misma extensión en tus archivos HTML. Eso evitará que te confundas al escribir los nombres de tus archivos unas veces con .htm y otras con .html. Cabe remarcar que hoy todo el mundo usa la extensión ".html" y no ".htm"

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 29/09/2016
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Sintaxis del HTML

Descripción de la sintaxis con la que se trabaja en el lenguaje HTML, así como la estructura que tendrá el documento básico HTML.

El HTML es un "lenguaje de marcado". Basa su sintaxis en un elemento base al que llamamos marca, tag o simplemente etiqueta. A través de las etiquetas vamos definiendo los elementos del documento, como enlaces, párrafos, imágenes, etc. Así pues, un documento HTML estará constituido por texto y un conjunto de etiquetas para definir la función que juega cada contenido dentro de la página. Todo eso servirá al navegador para saber cómo se tendrá que presentar el texto y otros elementos en la página.

Existen etiquetas para crear negritas, párrafos, imágenes, tablas, listas, enlaces, etc. Así pues, aprender HTML es básicamente aprenderse una serie de etiquetas, sus funciones, sus usos y saber un poco sobre cómo debe de construirse un documento básico. Es una tarea muy sencilla de afrontar, al alcance de cualquier personas, puesto que el lenguaje es muy entendible por los seres humanos.

Anatomía de una etiqueta HTML

La etiqueta presenta frecuentemente dos partes, su apertura y cierre, y se encierran ambas partes entre símbolos "menor que" y "mayor que". Lo veremos a continuación.

Apertura

El inicio de una etiqueta se produce de la siguiente manera:

```
<etiqueta>
```

Cierre

El final de una etiqueta se produce de manera similar a su apertura, aunque agregando una barra:

```
</etiqueta>
```

Nota: Por razones de formato del texto, en DesarrolloWeb.com, cuando no estamos escribiendo un código y queremos referirnos a una etiqueta la escribimos con mayúsculas. Los códigos se diferencian fácilmente porque tienen su estilo particular. Donde escribimos las etiquetas sin los mayores y menores que y con mayúsculas es en los párrafos del texto.

Todo lo incluido en el interior de esa etiqueta sufrirá las modificaciones que caracterizan a esta etiqueta. Así por ejemplo:

La etiqueta **B** define un texto en negrita. Si en nuestro documento HTML escribimos una frase con el siguiente código:

```
<b>Esto esta en negrita</b>
```

Veremos que las palabras "Esto esta en negrita" aparecen en negrita. Es así de simple.

Otro ejemplo rápido. La etiqueta **P** define un párrafo. Si en nuestro documento HTML escribimos:

```
<p>Hola, estamos en el párrafo 1</p>  
<p>Ahora hemos cambiado de párrafo</p>
```

Como resultado obtendríamos dos párrafos con esos textos. En HTML los párrafos están separados por un doble salto de línea. Se verían más o menos de esta manera:

Hola, estamos en el párrafo 1

Ahora hemos cambiado de párrafo

Partes de un documento HTML

Además de todo esto, **un documento HTML ha de estar delimitado por la etiqueta HTML**. Dentro de este documento, podemos asimismo distinguir dos partes principales:

La cabecera, delimitada por la etiqueta HEAD, donde colocaremos etiquetas de índole informativo, como por ejemplo el título de nuestra página. El contenido de la cabecera no suele aparecer en el cuerpo de la página, pero sirve a los navegadores y otros sistemas para encontrar información útil para entender y procesar el documento.

El cuerpo, flanqueado por la etiqueta BODY, que será donde colocaremos nuestro texto e imágenes

delimitados a su vez por otras etiquetas como las que hemos visto.

El resultado de un documento básico tiene la siguiente estructura:

```
<html>
<head>
  <title>Mi documento básico</title>
</head>
<body>

  <p>Este es el cuerpo de mi primera página HTML</p>
  <p>Este segundo párrafo también forma parte del cuerpo</p>

</body>
</html>
```

Nota: A este documento básico le faltan todavía algunas cosas importantes que no queremos que nunca se te olviden. Sin embargo hablaremos de ellas en el siguiente artículo, dedicado a la [página HTML básica](#).

Las mayúsculas o minúsculas son indiferentes al escribir etiquetas

En HTML las mayúsculas y minúsculas son indiferentes. Quiere decir que las etiquetas pueden ser escritas con cualquier tipo de combinación de mayúsculas y minúsculas. Resulta sin embargo aconsejable acostumbrarse a escribirlas en minúscula ya que otras tecnologías que pueden convivir con nuestro HTML (XML por ejemplo) no son tan permisivas y nunca viene mal hacernos a las buenas costumbres desde el principio, para evitar fallos triviales en un futuro.

Salto de línea en HTML

Otra de las cosas importantes de conocer sobre la sintaxis básica del HTML es que los saltos de línea no importan a la hora de interpretar una página. Un salto de línea será simplemente interpretado como un separador de palabras, un espacio en blanco. Es por ello que para separar líneas necesitamos usar la etiqueta de párrafo comentada antes, o la etiqueta BR que significa un salto de línea simple.

```
Esto es una línea
<br>
Esto es otra línea

Ahora, aunque estoy escribiendo aparentemente en otra línea, no se verá el salto de línea porque no lo he separado por el BR (o P, o cualquier otra etiqueta que produzca el salto de línea)
```

Nota: La etiqueta BR no tiene su correspondiente cierre. Es un detalle que quizás te haya llamado la atención. Volveremos sobre ello más adelante.

Es un detalle que choca al principio de usar HTML, pero al que te acabas acostumbrando con rapidez. Iremos viendo ejemplos a lo largo de todo el [Manual de HTML](#), por lo que no debes preocuparte por ahora, sólo seguir leyendo.

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 29/09/2016
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Tu primera página

Vamos a ver cómo se hace una página muy sencilla en HTML, que sirva de práctica a los debutantes.

Solo hemos comenzado el [Manual de HTML](#), pero ya podemos hacer un primer ejemplo completo y, sobre todo, correcto. Con lo que sabemos ya casi estamos listos para practicar, aunque todavía tenemos que adquirir algún conocimiento adicional.

No te olvides ahora de practicar, así que usa tu editor de código preferido y crea por tu cuenta los ejercicios que vamos a ir realizando. Recuerda que la práctica es la mejor vía para afianzar los conocimientos.

En el anterior artículo ya adelantamos la forma de un documento HTML básico, con sus etiquetas de cabecera y cuerpo. No obstante, aún tenemos que agregar alguna cosa adicional para que todo funcione de la mejor manera.

Doctype

Reconozco que el "doctype" no es la etiqueta más intuitiva, pero debemos mencionarla ahora porque es el inicio de cualquier archivo HTML. Viene heredada del XML, que es un lenguaje precursor del HTML. En el pasado la etiqueta Doctype era bastante más compleja, pero afortunadamente con la llegada de HTML5 se simplificó para quedar simplemente como esto:

```
<!DOCTYPE html>
```

No pretendas entenderla mucho, piensa en ella como una ceremonia que debes realizar (algo que debes escribir) al inicio de todo documento HTML.

Juego de caracteres

El juego de caracteres es otro asunto que puede parecer un poco complejo, pero que tenerlo claro desde el principio te ayudará a no pasar en el futuro por diversos problemas.

Este juego de caracteres, o codificación, depende del sistema operativo que estás usando para crear tu archivo HTML. Mientras que unos sistemas como Linux o Mac usan por defecto un juego de caracteres llamado UTF-8, en Windows se usa de manera predeterminada otro juego de caracteres llamado ISO-8859-

1. Parece una información un tanto técnica y fuera de necesidad para introducir ahora que estamos comenzando, pero insistimos que nos ahorrará frustraciones al dar los primeros pasos, pero sobre todo en un futuro.

En HTML5 el juego de caracteres a usar es siempre UTF-8. Por lo que tendremos que tener especial atención si somos usuarios de Windows, para asegurarnos que usamos la codificación correcta. Es otro de los motivos por los que en pasados artículos recomendábamos Brackets o Atom como editores de código, ya que éstos trabajan siempre en UTF-8, independientemente del sistema operativo. Si no estás usando uno de esos editores, te recomendamos hacerlo ahora y si te empeñas en trabajar con tu propio editor infórmate sobre el juego de caracteres que produce y si existe alguna opción o configuración que te asegure usar siempre UTF-8.

Para definir qué juego de caracteres estamos usando en un documento HTML se tiene que escribir una etiqueta en la cabecera de la página, en el HEAD, llamada META. Realmente las etiquetas META las trataremos más adelante, porque sirven para varias cosas interesantes. Pero de momento nos aseguraremos que tenemos esta etiqueta en el head.

```
<meta charset="UTF-8">
```

Igualmente, no pretendemos hablar mucho de esta etiqueta por el momento, solo que te la tomes como un contrato a cumplir para tener un documento correcto. Te ampliamos esta información en el artículo [Documento básico HTML5](#).

Un documento HTML correcto

A continuación tienes un documento básico con las etiquetas necesarias para comenzar con buen pie.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Mi primera página</title>
</head>
<body>
  <p><b>Bienvenido,</b></p>
  <p>Estás en la página <b>Comida para Todos</b>.</p>
  <p>Aquí aprenderás recetas fáciles y deliciosas.</p>
</body>
</html>
```

Puedes copiar y pegarlo en tu editor de código. Ahora guarda ese archivo con extensión .html o .htm en tu disco duro. Para ello accedemos al menú Archivo y seleccionamos la opción Guardar como. En la ventana elegimos el directorio donde deseamos guardarlo y colocaremos su nombre, por ejemplo mi_pagina.html

Consejo: Utiliza nombres en tus archivos que tengan algunas normas básicas para ahorrarte disgustos y líos. Nuestro consejo es que no utilices acentos ni espacios ni otros caracteres raros. También te ayudará escribir siempre las letras en minúsculas. Esto no quiere decir que debes hacer nombres de archivos

cortos, es mejor hacerlos descriptivos para que te aclaren lo que hay dentro. Algún caracter como el guión "-" o el guión bajo "_" te puede ayudar a separar las palabras. Por ejemplo quienes_somos.html

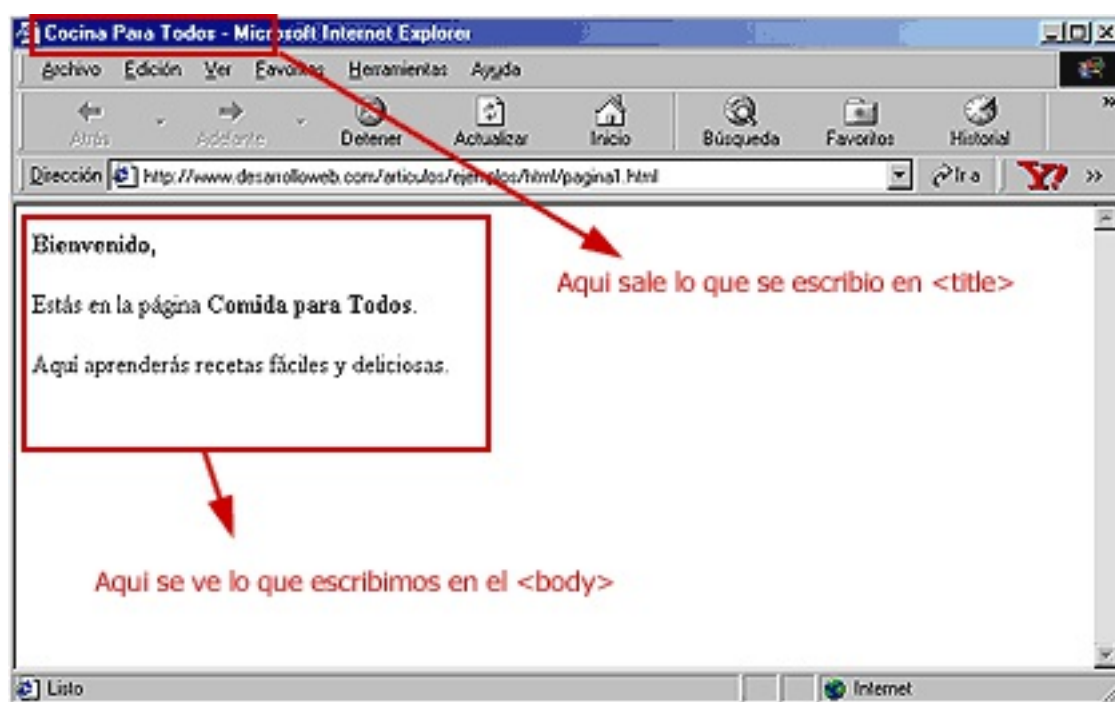
Con el documento HTML creado, podemos ver el resultado obtenido a partir de un navegador. Es conveniente, llegado a este punto, hacer hincapié en el hecho de que no todos los navegadores son idénticos a la hora de interpretar un documento. Desgraciadamente, los resultados de nuestro código pueden cambiar de uno a otro por lo que resulta aconsejable visualizar la página en varios clientes web. Generalmente se usan Chrome, Internet Explorer y Firefox como referencias ya que son los más extendidos.

A decir verdad, en el momento que estas líneas son escritas, Google Chrome acapara la mayoría de usuarios y Firefox e Internet Explorer/Edge están relegados a un segundo plano. Esto no quiere decir que lo debemos dejar totalmente de lado ya que incluso una minoría que puede proporcionarnos puede resultar muy importante para nosotros.

Ten en cuenta que el archivo debe tener codificación UTF-8, como hemos mencionado antes. Una vez guardado el fichero con extensión .html, para abrir la página en el navegador, simplemente tienes que acceder a la carpeta donde has guardado el archivo y darle un doble clic. Se trata de una tarea sencilla que estamos seguros que podrás realizar. Si no lo consigues, fíjate que la mayoría de los navegadores tienen un menú. En el menú de "Archivo" de tu navegador preferido encontrarás una opción como "Abrir archivo", desde donde también podrás abrir una página realizada por ti con tu editor de código.

Una vez abierto el archivo podréis ver vuestra primera página web. Algo sencillita pero por algo se empieza. Ya veréis como en poco tiempo seremos capaces de mejorar sensiblemente.

Fijaos en la parte superior izquierda de la ventana del navegador. Podréis comprobar la presencia del texto delimitado por la etiqueta TITLE. Esta es una de las funciones de esta etiqueta, cuyo principal cometido es el de servir de referencia en los motores de búsqueda como Google.



Por otro lado, los elementos que colocamos entre la etiqueta BODY, y su cierre, se pueden ver en el espacio

reservado para el cuerpo de la página.

Si ahora hacéis click con el botón derecho sobre la página y elegís "Ver código fuente de la página" (o View page source) veréis como en una ventana accesoria aparece el código de nuestro archivo HTML. Este recurso es de extrema importancia, ya que nos permite ver el tipo de técnicas empleadas por otros para la confección de sus páginas.

Con todo esto asimilado ya estamos en condiciones de adentrarnos un poco más en la descripción de algunas de las etiquetas más empleadas del HTML, en los próximos capítulos de este [manual de HTML](#).

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 29/09/2016
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Formatos básicos con HTML

Cómo realizar el formato de textos que se colocan en una página web. Aprende a utilizar tus primeras etiquetas HTML y atributos para definir los contenidos de la página y aplicar un formato básico.

Formato de párrafos en HTML

Cómo colocar párrafos y saltos de línea en páginas web. También vemos los encabezados como párrafos que sirven de título.

En los capítulos anteriores hemos presentado a título de ejemplo algunas etiquetas que permiten dar formato a nuestro texto. En este capítulo veremos con más detalle las más ampliamente utilizadas y ejemplificaremos algunas de ellas posteriormente.

Formatear un texto pasa por tareas tan evidentes como definir los párrafos, justificarlos, introducir viñetas, numeraciones o bien poner en negrita, itálica...

Hemos visto que para definir los párrafos nos servimos de la etiqueta P que introduce un salto y deja una línea en blanco antes de continuar con el resto del documento.

Podemos también usar la etiqueta BR, de la cual no existe su cierre correspondiente (/BR), para realizar un simple retorno de carro con lo que no dejamos una línea en blanco sino que solo cambiamos de línea.

Nota: Existen otras etiquetas que no tienen su correspondiente de cierre, como IMG para las imágenes, que veremos más adelante. Esto ocurre porque un salto de línea o una imagen no empiezan y acaban más adelante sino que sólo tienen presencia en un lugar puntual.

Podéis comprobar que cambiar de línea en nuestro documento HTML sin introducir alguna de estas u otras etiquetas no implica en absoluto un cambio de línea en la página visualizada. En realidad el navegador introducirá el texto y no cambiara de línea a no ser que esta llegue a su fin o bien lo especifiquemos con la etiqueta correspondiente.

Los párrafos delimitados por etiquetas P pueden ser fácilmente justificados a la izquierda, centro o derecha especificando dicha justificación en el interior de la etiqueta por medio de un atributo "align". Un atributo no es más que un parámetro incluido en el interior de la etiqueta que ayuda a definir el funcionamiento de la etiqueta de una forma más personalizada. Veremos a lo largo de este manual cantidad de atributos muy útiles para todo tipo de etiquetas.

Nota: Ten muy en cuenta lo siguiente, que ya hemos comentado anteriormente. **El HTML se usa**

para definir el contenido. Por tanto, los atributos align que vamos a conocer a continuación se están metiendo a una parcela que no le corresponde al HTML, porque están definiendo la forma con la que un párrafo debe de representarse, su estilo, y no el contenido. Es importante señalarlo para que aprendas que estas cosas se deben hacer mediante el lenguaje [CSS](#), que sirve para definir el estilo, la forma. En la revisión de este texto en 2016 hemos decidido mantener estos ejemplos, a pesar que no es el uso más correcto del HTML, porque así nos sirve para introducir los atributos de las etiquetas, que no hemos visto hasta ahora. No obstante, tú lo tendrás en cuenta cuando realices tus páginas y [aprenderás CSS](#) para ver cómo se deben de aplicar estos formatos.

Así, si deseásemos introducir un **texto alineado a la izquierda** escribiríamos:

```
<p align="left">Texto alineado a la izquierda</p>
```

Para una **justificación al centro**:

```
<p align="center">Texto alineado al centro</p>
```

Para **alinear a la derecha**:

```
<p align="right">Texto alineado a la derecha</p>
```

Los anteriores párrafos con sus alineaciones se verían más o menos así:

Texto alineado a la izquierda

Texto alineado al centro

Texto alineado a la derecha

Como veis, en cada caso el atributo align toma determinados valores que son escritos entre comillas. En algunas ocasiones necesitamos especificar algunos atributos para el correcto funcionamiento de la etiqueta. En otros casos, el propio navegador toma un valor definido por defecto. Para el caso de align, el valor por defecto es left.

Nota: Los atributos tienen sus valores indicados entre comillas ("), pero si no los indicamos entre comillas también funcionará en la mayoría de los casos. Sin embargo, es aconsejable que pongamos siempre las comillas para acostumbrarnos a utilizarlas, por dar homogeneidad a nuestros códigos y para evitar errores futuros en sistemas más quisquillosos.

El atributo align no es exclusivo de la etiqueta P. Otras etiquetas muy comunes, que veremos más adelante, entre las cuales se introducen texto o imágenes, suelen hacer uso de este atributo de una forma habitual.

Imaginemos un texto relativamente largo donde todos los párrafos están alineados a la izquierda (por ejemplo). Una forma de simplificar nuestro código y de evitar introducir continuamente el atributo align

sobre cada una de nuestras etiquetas es utilizando la etiqueta DIV.

Esta etiqueta, DIV, por si sola no sirve para nada, salvo producir un salto de línea simple. Para que realmente se vea tiene que estar acompañada de algún estilo definido en el CSS o de atributos del HTML como align y lo que nos permite es alinear cualquier elemento (párrafo o imagen) de la manera que nosotros deseemos.

Así, el código:

```
<p align="left">Parrafo1</p>
<p align="left"> Parrafo3</p>
<p align="left"> Parrafo2</p>
```

es equivalente a:

```
<div align="left">
<p>Parrafo1</p>
<p>Parrafo2</p>
<p>Parrafo3</p>
</div>
```

Nota: Recuerda que align tampoco sería correcto de usar en una etiqueta DIV, por el mismo motivo que no sería correcto de usar en un párrafo. Nos sirve para conocer facetas del HTML, que antes se usaban más y nos han quedado heredadas en las versiones actuales del lenguaje.

Como hemos visto, la etiqueta DIV marca divisiones en las que definimos un bloque de contenido, y a los que podríamos aplicar estilo de manera global, aunque lo correcto sería aplicar ese estilo del lado del CSS.

Ejemplo práctico:

Para practicar un poco lo que acabamos de ver vamos a proponer un ejercicio que podéis resolver en vuestros ordenadores. Simplemente queremos construir una página que tenga, por este orden:

2 Párrafos centrados 3 Párrafos alineados a la derecha Un salto de línea triple 1 párrafo alineado a la izquierda

El código fuente del ejercicio, con lo que sabemos hasta ahora, podría tener la siguiente forma:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>parrafos</title>
</head>
<body>
```

```
<p align="center">
Ejemplo de formatear parrafos
</p>
<p align="center">
Esto es el resultado:
</p>

<div align="right">
<p>
Que son los buscadores que no tienen porque mantener un índice y que tienen robots que constantemente recorren Internet en busca de nuevas
páginas para incluirlas en el buscador.
</p>
<p>
Imaginemos un texto relativamente largo donde todos los párrafos están alineados a la izquierda (por ejemplo).
</p>
<p>
Que son los buscadores que no tienen porque mantener un índice y que tienen robots que constantemente recorren Internet en busca de nuevas
páginas para incluirlas en el buscador. Estos buscadores suelen tener un formulario accesible desde la página inicial, con el enlace
correspondiente. No hay que navegar las categorías para acceder al formulario.
</p>
</div>
<br>
<br>
<br>
<p>
Esto es que acaba... hasta luego...
</p>

</body>
</html>
```

Al verlo en un navegador obtendríamos un resultado como el que sigue:

Ejemplo de formatear párrafos

Esto es el resultado:

Que son los buscadores que no tienen porque mantener un índice y que tienen robots que constantemente recorren Internet en busca de nuevas páginas para incluirlas en el buscador.

Imaginemos un texto relativamente largo donde todos los párrafos están alineados a la izquierda (por ejemplo).

Que son los buscadores que no tienen porque mantener un índice y que tienen robots que constantemente recorren Internet en busca de nuevas páginas para incluirlas en el buscador. Estos buscadores suelen tener un formulario accesible desde la página inicial, con el enlace correspondiente. No hay que navegar las categorías para acceder al formulario.

Esto es que acaba... hasta luego...

Encabezados

Existen otras etiquetas para definir párrafos especiales, que harán las veces de títulos. Son los encabezados o headings en inglés. Como decimos, son etiquetas que formatean el texto como un titular, pero el hecho de que cambien el formato no es lo que nos tiene que preocupar, sino el significado en sí de la etiqueta. Es cierto que los navegadores asignan un tamaño mayor de letra y colocan el texto en negrita, pero lo importante es que sirven para definir la estructura del contenido de un documento HTML. Así los navegadores para ciegos podrán informar a los invidentes que esta es una división nueva de contenido y que su titular es este o aquel. También motores de búsqueda sabrán interpretar mejor el contenido de una página en función de los titulares y subtítulos.

Hay varios tipos de encabezados, que se diferencian visualmente en el tamaño de la letra que utilizan. La etiqueta en concreto es la H1, para los encabezados más grandes, H2 para los de segundo nivel y así hasta H6 que es el encabezado más pequeño. Pero lo importante, insistimos es la estructura que denotan. Una página tendrá generalmente un encabezado de nivel 1 y dentro varios de nivel 2. Luego, dentro de los H2 encontraremos si acaso H3, etc. Nunca debemos usar los encabezados porque nos formateen el texto de una manera dada, sino porque nuestro documento lo requiera según su estructura.

Los encabezados implican también una separación en párrafos, así que todo lo que escribamos dentro de H1 y su cierre (o cualquier otro encabezado) se colocará en un párrafo independiente.

Podemos ver cómo se presentan algunos encabezados a continuación.

```
<h1>Encabezado de nivel 1</h1>
```

Los encabezados, como otras etiquetas de HTML, soportan el atributo align. Vemos un ejemplo de encabezado de nivel 2 alineado al centro, aunque repetimos que esta formatación debería hacerse en CSS.

```
<h2 align="center">Encabezado de nivel 2</h2>
```

Los encabezados se verán de esta manera en la página:

Encabezado de nivel 1

Encabezado de nivel 2

Encabezado de nivel 3

Encabezado de nivel 4

Encabezado de nivel 5

Encabezado de nivel 6

Otro ejercicio interesante es construir una página web que contenga todos los encabezados posibles. Se puede ver a continuación.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>headings</title>
</head>
<body>
  <h1>Encabezado de nivel 1</h1>
  <h2>Encabezado de nivel 2</h2>
  <h3>Encabezado de nivel 3</h3>
  <h4>Encabezado de nivel 4</h4>
  <h5>Encabezado de nivel 5</h5>
  <h6>Encabezado de nivel 6</h6>
</body>
</html>
```

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 30/09/2016
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Formateando el texto

Vemos como colocar negritas, itálicas, subrayados, subíndices y supréndices.

Además de todo lo relativo a la organización de los párrafos, uno de los aspectos primordiales del formateo

de un texto es el de la propia letra. Resulta muy común y práctico presentar texto resaltado en negrita, itálica y otros. Paralelamente el uso de índices, subíndices resulta vital para la publicación de textos científicos. Todo esto y mucho más es posible por medio del HTML a partir de multitud de etiquetas entre las cuales vamos a destacar algunas.

Pero antes de comenzar cabe hacer una reflexión sobre por qué son interesantes estas etiquetas y se siguen usando, a pesar que están entrando prácticamente en el terreno de CSS, ya que en la práctica están directamente formateando el aspecto de las fuentes. Son importantes porque las etiquetas en si no están para definir un estilo en concreto, sino una función de ciertas palabras dentro de un contenido. Por ejemplo, las negritas quieren decir que algo tiene más fuerza o importancia dentro de un texto y una itálica se puede usar para un texto que citado o algún énfasis particular. En cuanto a subíndices y superíndices todavía es más claro, ya que éstos especifican cosas que tiene que ver con el contenido y no con la presentación.

Negrita

Podemos escribir texto en negrita incluyéndolo dentro de las etiquetas B y su cierre (bold). Esta misma tarea es desempeñada por STRONG y su cierre, siendo ambas equivalentes. Nosotros nos inclinamos por la primeras por simple razón de esfuerzo.

Escribiendo un código de este tipo:

```
<b>Texto en negrita</b>
```

Obtenemos este resultado:

Texto en negrita

Nota: ¿Qué diferencia hay entre B y STRONG? Aunque las dos etiquetas hacen el mismo efecto, tienen una peculiaridad que las hace distintas. La etiqueta B indica negrita, mientras que la etiqueta STRONG indica que se debe escribir con fuerza. El HTML lo interpretan los navegadores según su criterio, es por eso que las páginas se pueden ver de distinta manera en unos browsers y en otros. La etiqueta H1 quiere decir "encabezado de nivel 1", es el navegador el responsable de formatear el texto de manera que parezca un encabezado de primer nivel. En la práctica los encabezados de los navegadores habituales son muy parecidos (tamaño de letra grande y en negrita), pero otro navegador podría colocar los encabezados con subrayado si le pareciese oportuno.

La diferencia entre b y STRONG se podrá entender ahora. Mientras que B significa simplemente negrita y todos los navegadores la interpretarán como negrita, STRONG es una etiqueta que significa que se tiene que resaltar fuertemente el texto y cada navegador es el responsable de resaltarlo como desee. En la práctica STRONG coloca el texto en negrita, pero podría ser que un navegador decidiese resaltar colocando negrilla, subrayado y color rojo en el texto.

Itálica

También en este caso existen dos posibilidades, una corta: I y su cierre (italic) y otra un poco más larga: EM

y su cierre. En este manual, y en la mayoría de las páginas que veréis por ahí, os encontraréis con la primera forma sin duda más sencilla a escribir y a acordarse.

He aquí un ejemplo de texto en itálica:

```
<i>Texto en itálica</i>
```

Que da el siguiente efecto:

Texto en itálica

Subrayado

El HTML nos propone también para el subrayado el par de etiquetas: U (underlined). Sin embargo, el uso de subrayados ha de ser aplicado con mucha precaución dado que los enlaces hipertexto van, a no ser que se indique lo contrario, subrayados con lo que podemos confundir al lector y apartarlo del verdadero interés de nuestro texto.

Además, cabe decir que la etiqueta U se ha quedado obsoleta, debido a que es algo que realmente se debe hacer del lado del [CSS](#), al ser básicamente un estilo.

Subíndices y supraíndices

Este tipo de formato resulta de extremada utilidad para textos científicos. Las etiquetas empleadas son:

```
<sup> y </sup> para los supraíndices  
<sub> y </sub> para los subíndices
```

Aquí tenéis un ejemplo:

```
La <sup>13</sup>CC<sub>3</sub>H<sub>4</sub>CINOS es un heterociclo alergeno enriquecido
```

El resultado:

La ¹³CC₃H₄CINOS es un heterociclo alergeno enriquecido

Anidar etiquetas

Todas estas etiquetas y por supuesto el resto de las vistas y que veremos más adelante pueden ser anidadas unas dentro de otras de manera a conseguir resultados diferentes. Así, podemos sin ningún problema crear texto en negrita e itálica embebiendo una etiqueta dentro de la otra:

```
<b>Esto sólo está en negrita <i>y esto en negrita e itálica</i></b>
```

Esto nos daría:

Esto sólo está en negrita y esto en negrita e itálica

Consejo: Cuando anides etiquetas HTML hazlo correctamente. Nos referimos a que si abres etiquetas dentro de otra más principal, antes de cerrar la etiqueta principal cierras las etiquetas que hayas abierto dentro de ella.

Debemos evitar códigos como el siguiente:

```
<b>Esto está en negrita e <i>itálica</b></i>
```

En favor de códigos con etiquetas correctamente anidadas:

```
<b>Esto está en negrita e <i>itálica</i></b>
```

Esto es muy aconsejable, aunque los navegadores entiendan bien las etiquetas mal anidadas, por dos razones:

1. Sistemas como XML no son tan permisivos con estos errores y puede que en el futuro nuestras páginas no funcionen correctamente.
2. A los navegadores les cuesta mucho tiempo de procesamiento resolver este tipo de errores, incluso más que construir la propia página y debemos evitarles que sufran por una mala codificación.

Todo lo que se ha visto hasta el momento en el Manual de HTML se puede ver en un vídeo para aprender visualmente. Si te interesa puedes acceder al [Videotutorial de HTML - Parte 1, documento básico y formato de texto](#).

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 11/10/2001
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Los colores y HTML

En este artículo aprenderás a crear colores en notación RGB con valores en hexadecimal, la manera más habitual de expresar un color en el lenguaje HTML. Explicamos la correcta utilización de los colores en el HTML.

En la composición de webs juegan un papel muy importante los colores. Usar una paleta de colores definida suele ayudar a la consistencia de un diseño y a transmitir ciertas sensaciones al usuario. Como parte de nuestro aprendizaje de HTML tenemos que detenernos a comprender cómo se expresan los colores en el

lenguaje.

En HTML se usa una notación específica de especificar un color, compuesta por tres valores "RGB": Red, Green, Blue. Rojo, Verde y Azul. Es decir, que para conseguir un color cualquiera mezclaremos cantidades de cada uno de esos colores. RGB es el modelo usado para la creación de colores de los monitores y televisores, así que es un excelente modo de expresar color en un medio digital como una web.

Los valores RGB en HTML se indican en numeración hexadecimal, en base 16. (Los dígitos pueden crecer hasta 16. Como no hay tantos dígitos numéricos, se utilizan las letras de la A a la F).

0=0	4=4	8=8	C=12
1=1	5=5	9=9	D=13
2=2	6=6	A=10	E=14
3=3	7=7	B=11	F=15

Para conseguir un color, mezclaremos valores asignando dos dígitos a cada valor RGB. De esta manera: "#RRGGBB"

Como has observado, colocamos también una almohadilla "#" al principio, para indicar que esa cadena es un valor de color en hexadecimal.

Más adelante en el artículo veremos ejemplos en una grande paleta, con sus valores en RGB. No obstante ejemplos podrían ser #000000 para el negro, #FFFFFF para el blanco, #660000 sería un rojo oscuro o #FF0000 un rojo brillante.

Nota: Habrás podido observar en alguna ocasión que los colores también se pueden expresar con algunas palabras, de hecho en el artículo sobre [Color y tipo de letra](#) ya lo comentamos. No obstante, es bastante más común escribir en RGB, porque es más versátil y podemos conseguir más fácilmente cualquier tonalidad deseada.

Atributos de color en etiquetas HTML

En HTML existen numerosas etiquetas que soportan atributos de color. Para que tengas una primera referencia, así se cambiaría la fuente para escribir en rojo:

```
<font color="#FF0000">Rojo</font>
```

Como ves, al Atributo **color** le damos un valor RGB en formato hexadecimal. El caracter # se coloca al principio de la cadena.

Nota: De nuevo tenemos que advertir sobre la necesidad de expresar todo lo que son estilos mediante [CSS](#). En HTML nos debemos centrar en lo que es escribir el contenido y en CSS en aplicar el estilo. Por supuesto, el color es más estilo que contenido, así que debería ir en el CSS. Es motivo por el cual toda la etiqueta FONT ha quedado en desuso, porque solamente nos servía para aplicar estilo. Para tu tranquilidad, en CSS los colores se pueden expresar de la misma manera que en HTML, por lo que no tendrás que aprender nada nuevo.

Por poner otro ejemplo, la etiqueta TABLE admite que se le exprese el color de fondo de la tabla. La veremos más adelante, pero lo consigues con el atributo **bgcolor**.

```
<table bgcolor="#ff8030">
```

Combinar otros colores

Al principio puede parecer difícil crear combinaciones de color con valores hexadecimales, pero con la práctica nos iremos acostumbrando y hasta seremos capaces de pensar un color y conseguir de cabeza un valor RGB aproximado. Nos vendrá bien tener en mente la rueda de colores:



Pero al final de lo que resulta más fácil echar mano es de un programa de diseño gráfico, que tiene selectores de color que nos suelen dar los valores RGB para que los podamos usar en cualquier programa. Algunos editores vienen con "color pickers" integrados para facilitar esta tarea, sin tener que cambiar de programa. La mayoría de los editores puede instalar de manera adicional plugins para implementar selectores de color, ya que es una demanda muy habitual de los desarrolladores.

Por ejemplo, otros colores RGB los puedes combinar así. Aunque al final de este artículo tienes una tabla de color completa.

Naranja	#FF8000
Verde turquesa	#339966
Azul oscuro	#000080

Colores seguros

Debemos estar preparados para recibir visitas desde todo tipo de dispositivos y a todos les debemos ofrecer una adecuada experiencia de usuario. En lo que respecta a los colores, no podemos saber a priori qué tipo

de pantalla va a tener la persona que nos visita y la resolución de color. Por eso una buena idea es usar aquellos colores considerados seguros: "Safe colors", colores compatibles con todos los sistemas.

Nota: Hoy la necesidad de usar colores seguros (aquellos que se verán bien en todos los monitores, independientemente de su paleta de color), no es tan grande como hace años, porque la tecnología ha evolucionado mucho y es raro encontrar un monitor que solo soporte 256 colores. No obstante es un conocimiento que resulta interesante por el hecho de remarcar la naturaleza universal de la web y la necesidad de construir páginas que sean capaces de adaptarse a cada medio donde va a ser consultada. En este manual de HTML no vamos a entrar en este tipo de detalles, pero si te interesan deberías leer el [Manual de Responsive Web Design](#).

La forma de conseguir colores seguros es limitando nuestros colores a los que se pueden conseguir utilizando los siguientes valores:

- 00
- 33
- 66
- 99
- AA
- CC
- FF

Ejemplos: #3366FF #FF9900 #666666

Es interesante comentar que, cuando usamos colores seguros, podemos resumir la notación RGB usando tres caracteres en vez de 6. Por ejemplo, #000 equivale a #000000. O #ABC equivale a #AABBCC.

Usando todas las combinaciones de "safe colors", conseguimos la siguiente paleta de colores:

#000000	#000033	#000066	#000099	#0000CC	#0000FF
#003300	#003333	#003366	#003399	#0033CC	#0033FF
#006600	#006633	#006666	#006699	#0066CC	#0066FF
#009900	#009933	#009966	#009999	#0099CC	#0099FF
#00CC00	#00CC33	#00CC66	#00CC99	#00CCCC	#00CCFF
#00FF00	#00FF33	#00FF66	#00FF99	#00FFCC	#00FFFF
#330000	#330033	#330066	#330099	#3300CC	#3300FF
#333300	#333333	#333366	#333399	#3333CC	#3333FF
#336600	#336633	#336666	#336699	#3366CC	#3366FF
#339900	#339933	#339966	#339999	#3399CC	#3399FF
#33CC00	#33CC33	#33CC66	#33CC99	#33CCCC	#33CCFF
#33FF00	#33FF33	#33FF66	#33FF99	#33FFCC	#33FFFF
#660000	#660033	#660066	#660099	#6600CC	#6600FF
#663300	#663333	#663366	#663399	#6633CC	#6633FF
#666600	#666633	#666666	#666699	#6666CC	#6666FF
#669900	#669933	#669966	#669999	#6699CC	#6699FF
#66CC00	#66CC33	#66CC66	#66CC99	#66CCCC	#66CCFF
#66FF00	#66FF33	#66FF66	#66FF99	#66FFCC	#66FFFF
#990000	#990033	#990066	#990099	#9900CC	#9900FF
#993300	#993333	#993366	#993399	#9933CC	#9933FF
#996600	#996633	#996666	#996699	#9966CC	#9966FF
#999900	#999933	#999966	#999999	#9999CC	#9999FF
#99CC00	#99CC33	#99CC66	#99CC99	#99CCCC	#99CCFF
#99FF00	#99FF33	#99FF66	#99FF99	#99FFCC	#99FFFF

#CC0000	#CC0033	#CC0066	#CC0099	#CC00CC	#CC00FF
#CC3300	#CC3333	#CC3366	#CC3399	#CC33CC	#CC33FF
#CC6600	#CC6633	#CC6666	#CC6699	#CC66CC	#CC66FF
#CC9900	#CC9933	#CC9966	#CC9999	#CC99CC	#CC99FF
#CCCC00	#CCCC33	#CCCC66	#CCCC99	#CCCCCC	#CCCCFF
#CCFF00	#CCFF33	#CCFF66	#CCFF99	#CCFFCC	#CCFFFF
#FF0000	#FF0033	#FF0066	#FF0099	#FF00CC	#FF00FF
#FF3300	#FF3333	#FF3366	#FF3399	#FF33CC	#FF33FF
#FF6600	#FF6633	#FF6666	#FF6699	#FF66CC	#FF66FF
#FF9900	#FF9933	#FF9966	#FF9999	#FF99CC	#FF99FF
#FFCC00	#FFCC33	#FFCC66	#FFCC99	#FFCCCC	#FFCCFF
#FFFF00	#FFFF33	#FFFF66	#FFFF99	#FFFFCC	#FFFFFF

Nota para la curiosidad: Este fue el primer artículo publicado en DesarrolloWeb.com. Aunque revisado en 2016 su publicación original es de 1999.

Este artículo es obra de *Miguel Angel Alvarez*.

Fue publicado por primera vez en 30/09/2016

Disponible online en <http://desarrolloweb.com/articulos/colores-html.html>

Atributos para páginas

Explicamos una serie de atributos que se aplican de manera global a toda la página, como el color de fondo el del texto, de los enlaces, márgenes, etc.

En este artículo nos metemos de nuevo en el terreno del CSS. Veremos todo tipo de estilos que se pueden aplicar a una página, colores o imágenes de fondo, colores para los enlaces, etc. Todo eso tiene que ir en el CSS. Si estás decidido a [aprender CSS](#) a continuación de aprender HTML (que deberías), puedes saltarte este texto tranquilamente. Ahora bien, si quieres seguir aprendiendo cosas del HTML y te apetece empezar con lo que sabes a poner un poco de color a la página, lee a continuación.

Las páginas HTML pueden construirse con variedad de atributos que le pueden dar un aspecto a la página muy personalizado. Podemos definir atributos como el color de fondo, el color del texto o de los enlaces. Estos atributos se definen en la etiqueta BODY y, como decíamos son generales a toda la página.

Lo mejor para explicar su funcionamiento es verlos uno por uno.

Atributos para fondos

bgcolor: especificamos un color de fondo para la página. En el [capítulo anterior](#) y en el [taller de los colores y HTML](#) hemos aprendido a construir cualquier color, con su nombre o su valor RGB. El color de fondo que podemos asignar con bgcolor es un color plano, es decir el mismo para toda la superficie del navegador.

background: sirve para indicar la colocación de una imagen como fondo de la página. La imagen se coloca haciendo un mosaico, es decir, se repite muchas veces hasta ocupar todo el espacio del fondo de la página. En capítulos más adelante veremos como se insertan imágenes con HTML y los tipos de imágenes que se pueden utilizar.

Ejemplo de fondo

Vamos a colocar esta imagen como fondo en la página.

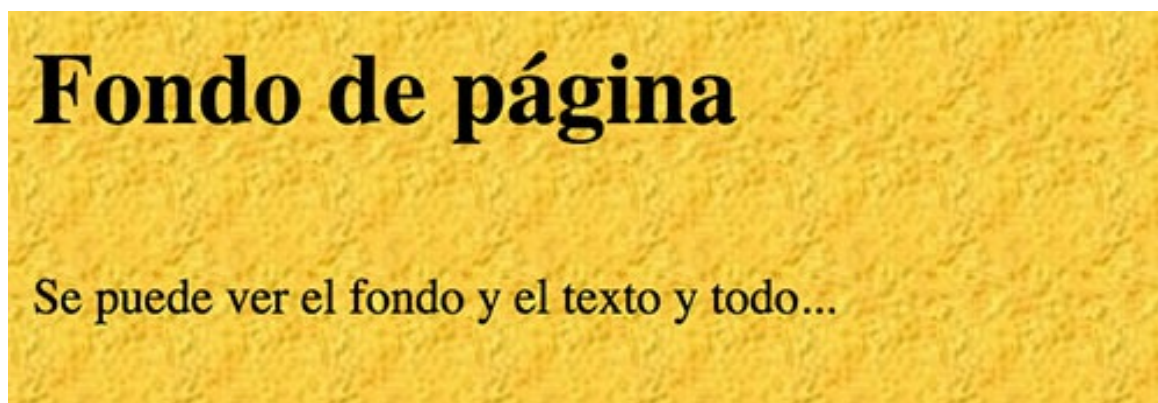


La imagen se llama fondo.jpg. Puedes guardarla en tu disco duro para practicar tú también con ella, mediante un clic con el botón derecho. Para trabajar con esta imagen vamos a colocarla en la misma carpeta donde está el HTML donde vamos a trabajar. Más adelante también hablaremos sobre cómo acceder a otros archivos que están en otras carpetas, mediante la composición de rutas un poco más complejas, pero por el momento suponemos que la imagen se encuentra en el mismo directorio que la página.

Para colocar esta imagen como fondo de mosaico, se escribiría la siguiente etiqueta BODY.

```
<body background="fondo.jpg">
```

Se puede ver el efecto que daría en la siguiente imagen como fondo.



Consejo: Siempre que coloquemos una imagen de fondo, debemos poner también un color de fondo cercano al color de la imagen.

Esto se debe a que, al colocar una imagen de fondo, el texto de la página debemos colocarlo en un color que contraste suficientemente con dicho fondo. Si el visitante no puede ver el fondo por cualquier cuestión (Por ejemplo tener desactivada la carga de imágenes) puede que el texto no contraste lo suficiente con el color de fondo por defecto de la web.

Creo que lo mejor será poner un ejemplo. Si la imagen de fondo es oscura, tendremos que poner un texto claro para que se pueda leer. Si el visitante que accede a la página no ve la imagen de fondo, le saldrá el fondo por defecto, que generalmente es blanco, de modo que al tener un texto con color claro sobre un fondo blanco, nos pasará que no podremos leer el texto convenientemente.

Ocurre parecido cuando se está cargando la página. Si todavía no ha llegado a nuestro sistema la imagen de fondo, se verá el fondo que hayamos seleccionado con bgcolor y es interesante que sea parecido al color de la imagen para que se pueda leer el texto mientras se carga la imagen de fondo.

Color del texto

text: este atributo sirve para asignar el color del texto de la página. Por defecto es el negro.

Además del color del texto, tenemos tres atributos para asignar el color de los enlaces de la página. Ya debemos saber que los enlaces deben diferenciarse del resto del texto de la página para que los usuarios puedan identificarlos fácilmente. Para ello suelen aparecer subrayados y con un color más vivo que el texto. Los tres atributos son los siguientes:

link: el color de los enlaces que no han sido visitados. (por defecto es azul clarito)

vlink: el color de los enlaces visitados. La "v" viene justamente de la palabra visitado. Es el color que tendrán los enlaces que ya hemos visitado. Por defecto su color es morado. Este color debería ser un poco menos vivo que el color de los enlaces normales.

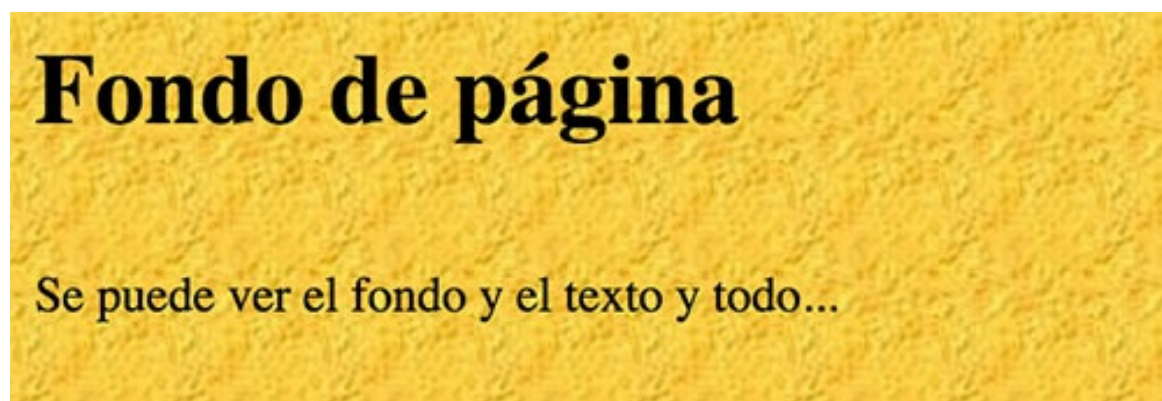
alink: es el color de los enlaces activos. Un enlace está activo en el preciso instante que se pulsa. A veces es difícil darse cuenta cuando un enlace está activo porque en el momento en el que se activa es porque lo estamos pulsando y en ese caso el navegador abandonará la página rápidamente y no podremos ver el enlace activo más que por unos instantes mínimos.

Ejemplo de color del texto

Vamos a ver una página donde el color de fondo sea negro, y los colores del texto y los enlaces sean claros. Pondremos el color de texto blanco y los enlaces amarillos, más resaltados los que no estén visitados y menos resaltados los que ya están visitados. Para ello escribiríamos la etiqueta BODY así:

```
<body bgcolor="#000000" text="ffffff" link="fffff33" alink="ffffcc" vlink="ffff00">
```

El efecto de esta definición de colores para la página se vería de la siguiente manera.



Márgenes

Con otros atributos de la etiqueta BODY se pueden asignar espacios de margen en las páginas, lo que es muy útil para eliminar los márgenes en blanco que aparecen a los lados, arriba y debajo de la página. Estos atributos son distintos para Internet Explorer y para otros navegadores, por lo que debemos utilizarlos todos si queremos que todos los clientes web los interpreten perfectamente.

leftmargin: para indicar el margen a los lados de la página. Válido para iexplorer.

topmargin: para indicar el margen arriba y debajo de la página. Para iexplorer.

marginwidth: la contrapartida de leftmargin para Firefox. (Margen a los lados)

marginheight: igual que topmargin, pero para Firefox. (Margen arriba y abajo)

Tenemos un artículo sobre la utilización de estos atributos para hacer [diseños avanzados con tablas en distintas definiciones de pantalla](#), que puede ser interesante de leer.

Un ejemplo de página sin margen es la propia página de DesarrolloWeb.com, que estás visitando actualmente. (Por lo menos a la hora de escribir este artículo) Además, vamos a ver otra página sin márgenes, por si alguien necesita ver el ejemplo en estas líneas.

```
<body topmargin=0 leftmargin=0 marginheight=0 marginwidth=0 bgcolor="ffffff">
<table width=100% bgcolor=ff6666><tr><td>
<h1>Hola amigos</h1>
<br>
<br>
Gracias por visitarme!
</td></tr></table>
</body>
```

Esta página tiene el fondo blanco y dentro una tabla con el fondo rojo. En la página podremos ver que la tabla ocupa el espacio en la página sin dejar sitio para ningún tipo de margen.

Todo lo que hemos visto hasta ahora en el [Manual de HTML](#) lo podemos encontrar en vídeo y en concreto las explicaciones de los últimos artículos se han recogido en el [Videotutorial de HTML - Fuentes, colores y estilos de BODY](#).

Por qué todos estos estilos deberían definirse en CSS

Como hemos dicho, todos estos estilos deberían indicarse en el CSS. Existen muchos motivos para ello pero uno de ellos seguro que ahora se podrá comprender. Imagina un sitio web con 30 páginas distintas (no tiene que ser muy grande para llegar a ese número). Imagina que llegado un día te cansas del color negro de fondo y lo quieres azul, y el color de los enlaces amarillo y lo quieres verde. Si tienes los estilos en el HTML tendrías que ir, página a página, cambiando los estilos "n" veces.

CSS, entre otras cosas, te permite tener los estilos definidos en un único lugar, un archivo con código en texto plano, y todas las páginas de tu sitio web usarían ese mismo archivo para definir su presentación. Así, si un día te cansas del color de fondo, el color del texto, el tipo de letra o su tamaño, entonces solo tienes que ir a un único lugar (el archivo CSS) y cambiarlo una única vez.

Todo eso lo veremos con detalle en el [Manual de CSS](#).

Este artículo es obra de *Miguel Angel Alvarez*
Fue publicado por primera vez en 04/03/2002
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Caracteres especiales

Una referencia útil, y una lista completa de los caracteres especiales del HTML.

En este artículo vamos a ver un tópico de frecuente consulta cuando se está aprendiendo a construir una página web. Son los caracteres especiales, unos códigos propios del lenguaje HTML que son capaces de entender los navegadores y que nos sirven para representar ciertos símbolos.

La necesidad de la existencia de los caracteres especiales responde a dos motivos que explicaremos a continuación, remarcando cuál de ellos es importante en la actualidad y por qué (actualizado en octubre de 2016). Al final del artículo también encontrarás unas tablas con un amplio conjunto de caracteres especiales y sus códigos, para que te sirva de referencia.

Luego entraremos en detalles, pero para ir viendo un ejemplo, la "á" (a minúscula acentuada) se escribe "á" de modo que la palabra página se podría escribir en una página HTML de este modo: página.

Por qué existen los caracteres especiales

Los motivos por los que debemos usar los caracteres especiales son los siguientes:

1. Una página web se ha de ver en países distintos, que usan conjuntos de caracteres distintos. El lenguaje HTML nos ofrece un mecanismo por el que podemos estar seguros que una serie de caracteres "raros" se van a ver bien en todos los ordenadores del mundo, independientemente de su juego de caracteres local.
2. Por otra parte, ciertos caracteres forman parte de las etiquetas como el "mayor qué" o "menor qué" que las delimitan. Mediante los caracteres especiales se definen unos códigos, que debemos usar cuando queremos poner uno de estos caracteres en una página,

De estos dos motivos, es verdaderamente importante el segundo, que es un asunto derivado del propio HTML y sus necesidades de codificación. Pero el primer motivo, aunque fue importante, hoy no lo es tanto porque la recomendación es usar UTF-8 y éste juego de caracteres sí que permite representar cualquiera de los símbolos que iremos a necesitar normalmente (todos los acentos, ñe, símbolo del Euro, etc.).

Caracteres especiales básicos

Estos son los caracteres especiales que se usan en HTML para no confundir un principio o final de etiqueta, unas comillas o un & con su correspondiente caracter.

<	>
&	"

Caracteres especiales del HTML 2.0

Ahora vamos a ver caracteres especiales, aunque muchos de ellos están disponibles en UTF-8, por lo que, si respetamos las recomendaciones de HTML5 para los juegos de caracteres, no necesitamos realmente

utilizarlos.

Á	Á	À	À
É	É	È	È
Í	Í	Ì	Ì
Ó	Ó	Ò	Ò
Ú	Ú	Ù	Ù
á	á	à	à
é	é	è	è
í	í	ì	ì
ó	ó	ò	ò
ú	ú	ù	ù
Ä	Ä	Â	Â
Ë	Ë	Ê	Ê
Ï	Ï	Î	Î
Ö	Ö	Ô	Ô
Ü	Ü	Û	Û
ä	ä	â	â
ë	ë	ê	ê
ï	ï	î	î
ö	ö	ô	ô
ü	ü	û	û
Ã	Ã	å	å
Ñ	Ñ	Å	Å
Õ	Õ	Ç	Ç
ã	ã	ç	ç
ñ	ñ	Ý	Ý
õ	õ	ý	ý
Ø	Ø	ÿ	ÿ

ø	ø	Þ	þ
Ð	Ð	þ	þ
ð	ð	Æ	Æ
ß	ß	æ	æ

Caracteres especiales del HTML 3.2

¼	¼	 	
½	½	¡	¡
¾	¾	£	£
©	©	¥	¥
®	®	§	§
ª	ª	¤	¤
²	²	¦	¦
³	³	«	«
¹	¹	¬	¬
¯	—	­	–
µ	μ	º	º
¶	¶	´	´
·	·	¨	¨
°	°	±	±
¸	,	»	»
¿	¿		

Otros caracteres especiales

×	×	¢	¢
÷	÷	€	€
“	"	™	™
”	"	‰	‰
Œ	Œ	ƒ	ƒ
‡	‡	†	†

Referencia: Hay un videotutorial en DesarrolloWeb.com que trata sobre [enlaces y caracteres especiales](#). No son temas muy relacionados entre sí, pero seguro que nos sirve como práctica para aprender todo lo que se ha tratado en este artículo.

Este artículo es obra de *Miguel Angel Alvarez*
Fue publicado por primera vez en 01/10/2016
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Listas en HTML

A continuación comenzaremos a explicar las listas de HTML que implican varias etiquetas para crear su estructura. Veremos varios tipos de listas que se pueden utilizar para diversos objetivos.

Listas en HTML: Listas desordenadas

Vemos lo que son las listas y señalamos los tres tipos que hay. Estudiamos las listas desordenadas.

Las posibilidades que nos ofrece el HTML en cuestión de tratamiento de texto son realmente notables. No se limitan a lo visto hasta ahora, sino que van más lejos todavía. Varios ejemplos de ello son las listas, que sirven para enumerar y definir elementos, los textos preformateados y las cabeceras o títulos.

Las listas originalmente están pensadas para citar, numerar y definir cosas a través de características, o al menos así lo hacemos en la escritura de textos. Sin embargo, las listas finalmente se utilizan para mucho más que enumerar una serie de puntos, en realidad son un recurso muy interesante para poder maquetar elementos diversos, como barras de navegación, pestañas etc. Pero esto lo veremos más adelante, cuando apliquemos estilos [CSS](#) a las listas.

De momento, en este [Manual de HTML](#), trataremos las listas desde el punto de vista de su construcción y veremos los diferentes tipos que existen, y que podemos utilizar para resolver nuestras distintas necesidades a la hora de escribir textos en HTML.

Referencia: Todos los tipos de listas los hemos explicado en vídeo y se pueden revisar en el [Videotutorial de HTML](#), más concretamente en la entrega siguiente: [Vídeo sobre las Listas HTML](#).

Podemos distinguir tres tipos de listas HTML:

- [Listas desordenadas](#)
- [Listas ordenadas](#)
- [Listas de definición](#)

Las veremos detenidamente una a una.

Listas desordenadas

Son delimitadas por las etiquetas UL y su cierre (unordered list). Cada uno de los elementos de la lista es citado por medio de una etiqueta LI (La LI tiene su cierre, aunque si no lo colocas el navegador al ver el siguiente LI interpretará que estás cerrando el anterior). La cosa queda así:


```
<p>Países del mundo</p>
<ul>
  <li>Argentina</li>
  <li>Perú</li>
  <li>Chile</li>
</ul>
```

El resultado:

Países del mundo

- Argentina
- Perú
- Chile

Podemos definir el tipo de viñeta empleada para cada elemento. Para ello debemos especificarlo por medio del atributo `type` incluido dentro de la etiqueta de apertura `UL`, si queremos que el estilo sea válido para toda la lista, o dentro de la etiqueta `LI` si queremos hacerlo específico de un solo elemento. La sintaxis es del siguiente tipo:

```
<ul type="tipo de viñeta">
```

donde tipo de viñeta puede ser uno de los siguientes:

- circle
- disc
- square

Nota: En algunos navegadores no funciona la opción de cambiar el tipo de viñeta a mostrar y por mucho que nos empeñemos, siempre saldrá el redondel negro.

En caso de que no funcione siempre podemos construir la lista a mano con la viñeta que queramos utilizando las tablas de HTML. Veremos más adelante cómo trabajar con tablas.

También es posible definir el tipo de viñeta o bullet por medio de CSS.

Vamos a ver un ejemplo de lista con un cuadrado en lugar de un redondel, y en el último elemento colocaremos un círculo. Para ello vamos a colocar el atributo `type` en la etiqueta `UL`, con lo que afectará a todos los elementos de la lista.

```
<ul type="square">
```

```
<li>Elemento 1
<li>Elemento 2
<li>Elemento 3
<li type="circle">Elemento 4
</ul>
```

Que tiene como resultado:

- Elemento 1
- Elemento 2
- Elemento 3
- Elemento 4

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 11/10/2001
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Listas en HTML: Listas ordenadas

Estudiamos otro tipo de listas: las listas ordenadas.

Estamos en el [Manual de HTML](#) y continuamos estudiando las listas de HTML, con las que crear estructuras atractivas para presentar la información. En el capítulo anterior vimos las [listas desordenadas](#) y ahora estudiaremos las listas ordenadas.

Listas ordenadas

Las listas ordenadas sirven también para presentar información, en diversos elementos o items, con la particularidad que éstos estarán predcidos de un número o una letra para enumerarlos, siempre por un orden.

Para realizar las listas ordenadas usaremos las etiquetas OL (ordered list) y su cierre. Cada elemento sera igualmente indicado por la etiqueta LI, que ya vimos en las listas desordenadas.

Pongamos un ejemplo:

```
<p>Reglas de comportamiento en el trabajo</p>
<ol>
<li>El jefe siempre tiene la razón
<li>En caso de duda aplicar regla 1
</ol>
```

El resultado es:

1. El jefe siempre tiene la razón
2. En caso de duda aplicar regla 1

Del mismo modo que para las listas desordenadas, las listas ordenadas ofrecen la posibilidad de modificar el estilo. En concreto nos es posible especificar el tipo de numeración empleado eligiendo entre números (1, 2, 3...), letras (a, b, c...) y sus mayúsculas (A, B, C,...) y números romanos en sus versiones mayúsculas (I, II, III,...) y minúsculas (i, ii, iii,...).

Para realizar dicha selección hemos de utilizar, como para el caso precedente, el atributo `type`, el cual será situado dentro de la etiqueta `OL`. Los valores que puede tomar el atributo en este caso son:

1 Para ordenar por números

a Por letras del alfabeto

A Por letras mayúsculas del alfabeto

i Ordenación por números romanos en minúsculas

I Ordenación por números romanos en mayúsculas

Nota: Recordamos que en algunos navegadores no funciona la opción de cambiar el tipo de viñeta a mostrar.

Puede que en algún caso deseemos comenzar nuestra enumeración por un número o letra que no tiene por qué ser necesariamente el primero de todos. Para solventar esta situación, podemos utilizar un segundo atributo, `start`, que tendrá como valor un número. Este número, que por defecto es 1, corresponde al valor a partir del cual comenzamos a definir nuestra lista. Para el caso de las letras o los números romanos, el navegador se encarga de hacer la traducción del número a la letra correspondiente.

Os proponemos un ejemplo usando este tipo de atributos:

```
<p>Ordenamos por números</p>

<ol type="1">
  <li>Elemento 1
  <li> Elemento 2
</ol>

<p>Ordenamos por letras</p>
<ol type="a">
  <li>Elemento a
  <li> Elemento b
</ol>
```

```
<p>Ordenamos por números romanos empezando por el 10</p>
```

```
<ol type="i" start="10">  
<li>Elemento x  
<li> Elemento xi  
</ol>
```

El resultado:

Ordenamos por números

Elemento 1
Elemento 2

Ordenamos por letras

Elemento a
Elemento b

Ordenamos por números romanos empezando por el 10

Elemento x
Elemento xi

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 11/10/2001
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Listas en HTML: listas de definición

Vemos las listas de definición y aprendemos a anidar listas para crear estructuras lista más complejas.

Terminamos el tema de listas en el [manual de HTML](#) de DesarrolloWeb.com estudiando las listas de definición. Veremos también la anidación de listas, que resultará un recurso interesante para estructurar datos un poco más complejos o enumerar elementos con una jerarquía.

Listas de definición

Las listas de definición sirven para hacer un conjunto de elementos con pares concepto-descripción. Es decir, se especificarán varios términos por su nombre y se escribirá una definición para cada uno. Cada

elemento es presentado junto con su definición, uno detrás de otro.

Nota: Este tipo de listas la verdad es que no se usan mucho. Es un buen recurso, porque permite aplicar semántica a los items de una lista, que quedan asociados a su definición, por lo que usarlas no será una mala idea, sin embargo casi nadie las usa en la práctica.

Para realizar una lista de definición, la etiqueta principal es DL y su cierre (definition list). Las etiquetas del elemento y su definición son DT (definition term) y DD (definition definition) respectivamente.

Aquí os proponemos un código que podrá aclarar este sistema:

```
<p>Diccionario de la Real Academia</p>
<dl>
  <dt>Brujula
  <dd>Señorula montada en una escóbula
  <dt>Oreja
  <dd>Sesenta minutejos
</dl>
```

El efecto producido:

Diccionario de la Real Academia

Brujula

Señorula montada en una escóbula

Oreja

Sesenta minutejos

Fijaos en que cada línea DD está desplazada hacia la izquierda. Este tipo de etiquetas son usadas a menudo con el propósito de crear textos más o menos desplazados hacia la izquierda.

El código:

```
<dl>
  <dd>Primer nivel de desplazamiento
    <dl>
      <dd>Segundo nivel de desplazamiento
        <dl>
          <dd>Tercer nivel de desplazamiento
        </dl>
    </dl>
</dl>
```

```
</dl>  
</dl>
```

El resultado:

Primer nivel de desplazamiento

Segundo nivel de desplazamiento

Tercer nivel de desplazamiento

Anidando listas

Nada nos impide utilizar todas estas etiquetas de forma anidada como hemos visto en otros casos. De esta forma, podemos conseguir listas mixtas como por ejemplo:

```
<p>Ciudades del mundo</p>  
<ul>  
  <li>Argentina  
    <ol>  
      <li>Buenos Aires  
      <li>Bariloche  
    </ol>  
  <li>Uruguay  
    <ol>  
      <li>Montevideo  
      <li>Punta del Este  
    </ol>  
</ul>
```

De esta forma creamos una lista como esta:

Ciudades del mundo

- Argentina
 1. Buenos Aires
 2. Bariloche
- Uruguay
 1. Montevideo
 2. Punta del Este

Referencia: Con esto hemos acabado el tema sobre Listas HTML, sin embargo, podemos completar las explicaciones en el [Videotutorial de HTML](#), en el [Video sobre las Listas HTML](#).

Este artículo es obra de *Rubén Álvarez*

Fue publicado por primera vez en 11/10/2001

Disponible online en <http://desarrolloweb.com/articulos/21.php>

Todo sobre los enlaces en HTML

Los enlaces son los elementos que nos permiten navegar por las páginas HTML y son tan importantes que la web no tendría sentido sin ellos. Dedicaremos varios capítulos a explorar los distintos tipos de enlaces, sus usos y diversos consejos para hacer páginas navegables.

Enlaces en HTML

Vemos qué son los enlaces en HTML y los distintos tipos.

Hasta aquí, hemos podido ver que una página web es un archivo HTML en el que podemos incluir, entre otras cosas, textos formateados a nuestro gusto e imágenes (las veremos con detalle enseguida). Del mismo modo, un sitio web podrá ser considerado como el conjunto de archivos, principalmente páginas HTML e imágenes, que constituyen el contenido al que el navegante tiene acceso.

Sin embargo, no podríamos hablar de navegante o de navegación si estos archivos HTML no estuviesen debidamente conectados entre ellos y con el exterior de nuestro sitio por medio de enlaces hipertexto. En efecto, el atractivo original del HTML radica en la posible puesta en relación de los contenidos de los archivos introduciendo referencias bajo forma de enlaces que permitan un acceso rápido a la información deseada. De poco serviría en la red tener páginas aisladas a las que la gente no puede acceder y desde las que la gente no puede saltar a otras.

Un enlace puede ser fácilmente detectado por el usuario en una página. Basta con deslizar el puntero del ratón sobre las imágenes o el texto y ver como cambia de su forma original transformándose por regla general en una mano con un dedo señalador. Adicionalmente, estos enlaces suelen ir, en el caso de los textos, coloreados y subrayados para que el usuario no tenga dificultad en reconocerlos.

Sintaxis de un enlace

Para colocar un enlace, nos serviremos de las etiquetas `A` y su cierre. Dentro de la etiqueta de apertura deberemos especificar asimismo el destino del enlace. Este destino será introducido bajo forma de atributo, el cual lleva por nombre "href".

La sintaxis general de un enlace es por tanto de la forma:

```
<a href="destino">contenido</a>
```

Siendo el "*contenido*" un texto o una imagen. Es la parte de la página que se colocará activa y donde deberemos pulsar para acceder al enlace. Por su parte, "*destino*" será una página, un correo electrónico o un archivo.

Por ejemplo, un enlace a la home de DesarrolloWeb, tendría esta manera

```
<a href="http://www.desarrolloweb.com/">Home de Desarrolloweb.com</a>
```

Ahora, si queremos que el contenido del enlace sea una imagen y no un texto, podremos colocar la correspondiente etiqueta IMG dentro de la etiqueta A.

```
<a href="http://www.escuela.it"></a>
```

Nota: Todavía no hemos explicado la etiqueta IMG, pero no te preocupes porque más adelante encontrarás un apartado donde se explica [cómo se trabaja con imágenes en HTML](#).

El aspecto de los enlaces

Nosotros mediante el HTML, y las hojas de estilo CSS, podemos definir el aspecto que tendrán los enlaces en una página. Sin embargo, ya de manera predeterminada el navegador los destaca para que los podamos distinguir. Generalmente encontraremos a los enlaces subrayados y coloreados en azul, aunque esta regla depende del navegador del usuario y de sus estilos definidos como predeterminados.

En el caso de las imágenes que sirvan de enlace, tradicionalmente aparecían encuadradas en un marco azul por defecto. Aunque ese estilo predeterminado también cambiará dependiendo del navegador y de hecho, en 2016, la mayoría de navegadores ya no ponen ese marco azul, así que tenemos un ejemplo de cómo los estilos predeterminados pueden cambiar con el tiempo y con versiones de navegador.

Por ese incierto estilo predeterminado siempre es interesante marcar por nosotros mismos el estilo que los enlaces deben tener en nuestra página. Ese estilo lo correcto es colocarlo en el código de CSS, pero también se puede definir en la etiqueta BODY. Eso es algo que se explicó en el artículo [Atributos generales de la página](#).

Tipos de enlaces

Para estudiar en profundidad los enlaces tenemos que clasificarlos por su tipo, porque dependiendo ese tipo algunas cosas cambiarán a la hora de construirlos.

En función del destino los enlaces son clásicamente agrupados del siguiente modo:

- [Enlaces internos](#): los que se dirigen a otras partes dentro de la misma página.
- [Enlaces locales](#): los que se dirigen a otras páginas del mismo sitio web.
- [Enlaces remotos](#): los dirigidos hacia páginas de otros sitios web.
- [Enlaces con direcciones de correo](#): para crear un mensaje de correo dirigido a una dirección.
- [Enlaces con archivos](#): para que los usuarios puedan hacer download de ficheros.

Todos estos tipos de enlaces los iremos repasando en los próximos artículos del [Manual de HTML](#).

Referencia: Para complementar las explicaciones sobre los enlaces HTML recomendamos ver el [videotutorial sobre enlaces](#).

Este artículo es obra de *Rubén Álvarez*

Fue publicado por primera vez en 02/10/2016

Disponible online en <http://desarrolloweb.com/articulos/21.php>

Enlaces internos

Los enlaces HTML que se hacen con otras partes de la misma página.

Quizás "Enlaces internos" puede ser un poco ambiguo, porque podríamos pensar tanto en enlaces internos dentro del mismo sitio web o enlaces internos dentro de la misma página. Nosotros en este manual nos referimos a los los enlaces que apuntan a un lugar diferente dentro de la misma página. Llamamos "Enlaces locales" a los [enlaces que apuntan a otra página dentro de mismo sitio web](#).

Este tipo de enlaces son esencialmente utilizados en páginas donde el acceso a los contenidos puede verse dificultado debido al gran tamaño del texto. Es un enlace poco habitual en páginas web como blogs o páginas comerciales, que presentan un producto o un servicio. Se encuentran más en páginas de referencia, donde además el contenido está dividido en diversas secciones y queremos poder navegar entre esas secciones que se encuentran dentro del mismo archivo HTML. Otro uso habitual de los enlaces internos es ofrecer al visitante la posibilidad de ir rápidamente al principio de la página, a la parte de arriba.

Enlace y ancla

Para crear un enlace de este tipo es necesario dos componentes, que para aclararnos los vamos a nombrar de la siguiente forma:

- El enlace: Sería el link, lo que aparecerá en la página para que el usuario haga clic. Sería el enlace de origen propiamente dicho.
- El ancla: Además se requiere una marca, para saber dónde se dirige el enlace. Es el destino donde nos llevará el navegador al pulsar el link. Le llamamos ancla porque nos permite anclar a esa posición otros enlaces.

Ambos elementos se crean con la misma etiqueta A, tanto el enlace como el ancla. Solo que usaremos distintos atributos dentro de esa etiqueta.

Sintaxis de los enlaces en la misma página

Veamos más claramente como funcionan estos enlaces con un ejemplo sencillo: Supongamos que queremos crear un enlace que apunte al final de la página. Lo primero será colocar nuestro enlace origen. Este enlace de origen es el que el usuario podrá hacer clic.

```
<a href="#abajo">Ir abajo</a>
```

Como podéis ver, el contenido del enlace es el texto "Ir abajo" y el destino, #abajo, es un punto de la misma página que todavía no hemos definido. Ojo al símbolo "#": es él quien especifica al navegador que el enlace apunta a una sección en particular, a un punto interno dentro de la misma página.

En segundo lugar, hay que generar un enlace en el destino, al que hemos llamado "ancla". Este enlace no llevará contenido, puesto que no queremos que nadie lo pulse, sino que nos sirva de ancla. Tampoco llevará el atributo "href", porque no tiene que apuntar a ningún lugar, sino que le apuntarán a él. Para poder distinguirlo de otros posibles enlaces realizados dentro de la misma página a cada ancla se le asigna un nombre por medio del atributo "name". En este caso, la etiqueta que escribiremos será ésta:

```
<a name="abajo"></a>
```

Para entender cómo crear los enlaces internos nos tenemos que fijar en el name="abajo" del ancla. Pues bien, si queremos crear un enlace interno a esta ancla, colocaremos en el enlace de origen el href="#abajo", o sea, el nombre del enlace más un "#" para que el navegador sepa que es un enlace interno.

Enlaces útiles pero no tan habituales

A decir verdad, estos enlaces, aunque útiles, no son los más extendidos de cuantos hay. La tendencia general es la de crear páginas (archivos) independientes con tamaños más reducidos enlazados entre ellos por [enlaces locales](#) (los veremos enseguida). De esta forma evitamos el exceso de tiempo de carga de un archivo y la introducción de exceso de información que pueda desviar la atención del usuario.

Una aplicación corriente de estos enlaces consiste en poner un pequeño índice al principio de nuestro documento donde introducimos enlaces origen a las diferentes secciones. Paralelamente, al final de cada sección introducimos un enlace que apunta al índice de manera que podamos guiar al navegante en la búsqueda de la información útil para él.

En el siguiente artículo veremos cómo se hacen los [enlaces locales](#), que son mucho más comunes en sitios web.

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 02/10/2016
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Enlaces locales

Cómo construir enlaces en HTML cuyo destino sean otras páginas dentro del mismo sitio web.

Como hemos dicho, un sitio web está constituido de páginas interconexas, que se relacionan mediante enlaces de hipertexto. Para abordar el estudio dividimos la materia por los distintos tipos de enlaces que nos podemos encontrar, atendiendo al tipo de destino. En el capítulo anterior vimos [cómo enlazar distintas secciones dentro de una misma página](#).

En este artículo nos pondremos con otros tipos de enlaces, a los que hemos llamado "Enlaces locales". Se

trata de un tipo de enlace mucho más común en el día a día del desarrollo. De hecho, es el tipo de enlace que más se produce en lo general. Estos enlaces locales nos permiten relacionar distintos documentos HTML que componen un sitio web. Gracias a los enlaces locales podremos convertir varias páginas sueltas en un sitio web completo, compuesto de varios documentos.

Para crear este tipo de enlaces, hemos de usar la misma etiqueta A que ya conocemos, de la siguiente forma:

```
<a href="archivo.html">contenido</a>
```

Rutas de los enlaces

Hacer un enlace en si no es para nada complejo. No requiere muchas explicaciones con lo que ya conocemos del [manual de HTML](#), sin embargo hay que abordar con detalle un tema importante: las rutas de los enlaces. Como rutas nos referimos al destino del enlace, o sea, lo que ponemos en el atributo "href" y es importante que nos paremos aquí porque nos puede dar algunos problemas al desarrollar, sobre todo para las personas que puedan tener menos experiencia en el trabajo con el ordenador.

Por regla general, para una mejor organización, los sitios suelen estar ordenados por directorios. Estos directorios suelen contener diferentes secciones de la página, imágenes, scripts, estilos... Es por ello que en muchos casos no nos valdrá con especificar el nombre del archivo, sino que tendremos que especificar además el directorio en el que nuestro archivo.html esta alojado.

Nota: Si habéis trabajado con MS-DOS o Linux por línea de comandos no tendréis ningún problema para comprender el modo de funcionamiento. Tan solo, para los usuarios de Windows hay que tener cuidado en usar la barra "/" en lugar de la contrabarra "\", pues las contrabarras usadas en Windows para separar componentes de la ruta no se deben usar nunca al especificar rutas en HTML.

Para aquellos que no saben como mostrar un camino de un archivo, aquí van una serie de indicaciones que os ayudaran a comprender la forma de expresarlos. No resulta difícil en absoluto y con un poco de practica lo haréis prácticamente sin pensar.

1. Hay que situarse mentalmente en el directorio en el que se encuentra la página donde vamos a crear el enlace.
2. Si la página destino está en el mismo directorio que el archivo desde donde vamos a enlazar podemos colocar simplemente el nombre del archivo de destino, ya que no hay necesidad de cambiar de directorio.
3. Si la página de destino está en una carpeta o subdirectorio interior al directorio donde está el archivo de origen, hemos de marcar la ruta enumerando cada uno de los directorios por los que pasamos hasta llegar al archivo de destino, separándolos por el símbolo barra "/". Al final obviamente, escribimos el nombre del archivo destino.
4. Si la página destino se encuentra en un directorio padre (superior al de la página del enlace), hemos de escribir dos puntos y una barra "../" tantas veces como niveles subamos en la arborescencia hasta dar con el directorio donde esta emplazado el archivo destino.
5. Si la página se encuentra en otro directorio no incluido ni incluyente del archivo origen, tendremos que subir como en la regla 3 por medio de "../" hasta encontrar un directorio que englobe el

directorio que contiene a la página destino. A continuación haremos como en la regla 2. Escribiremos todos los directorios por los que pasamos hasta llegar al archivo.

Se verá mejor enseguida con unos ejemplos.

Imagina que tienes la siguiente estructura de carpetas y archivos. La que aparece en la siguiente imagen.



1) Para hacer un enlace desde index.html hacia yyy.html:

```
<a href="seccion1/paginas/yyy.html">ir a yyy.html</a>
```

2) Para hacer un enlace desde xxx.html hacia yyy.html:

```
<a href="../../seccion1/paginas/yyy.html">Ir (también) a yyy.html</a>
```

3) Para hacer un enlace desde yyy.html hacia xxx.html:

```
<a href="../../../seccion2/xxx.html">Ir ahora a xxx.html</a>
```

Enlazar con una página diferente, pero en una sección interna

Los enlaces locales pueden, a su vez, apuntar ya no a la página en general sino más precisamente a una sección concreta. Este tipo de enlaces resultan ser un híbrido de [interno](#) y local. La sintaxis es de este tipo:

```
<a href="archivo.html#seccion">contenido</a>
```

Como para los enlaces internos, en este caso hemos de marcar la sección con un ancla:

```
<a name="seccion"></a>
```

Hasta aquí todo lo que necesitas saber sobre enlaces dentro del mismo sitio web. Aunque aún nos quedan por ver otros tipos de enlaces. En el siguiente artículo analizaremos los [enlaces externos](#), los [enlaces a direcciones de correo](#) y los [que llevan a archivos para descarga](#).

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 03/10/2016
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Enlaces externos, de correo y hacia archivos

Vemos tres tipos de enlaces. Los dirigidos a otras páginas de otros webs, a direcciones de correo y a ficheros externos.

A lo largo de los artículos anteriores del [Manual de HTML](#) hemos visto enlaces internos, dentro de la misma página, y los enlaces locales, que se dan cuando se referencian páginas dentro del mismo sitio web. Eran tipos de enlaces muy comunes, pero aun hay otros tipos que debemos de repasar para completar la información.

En este artículo abordaremos los últimos 3 tipos de enlaces que habíamos señalado. No tienen mucho que ver entre si, pero como son sencillos los agrupamos todos en el presente texto. Son enlaces a páginas externas (otros dominios), enlaces a direcciones de email y enlaces a archivos de descarga.

Enlaces remotos

Son los enlaces que se dirigen hacia páginas que se encuentran fuera de nuestro sitio web, es decir, cualquier otro documento que no forma parte de nuestro sitio. Generalmente nuestro sitio web estará en un dominio determinado, tipo example.com. Los enlaces remotos son los que van a páginas que estarían en otro dominio diferente.

Este tipo de enlaces es muy común y no representa ninguna dificultad. Simplemente colocamos en el atributo HREF de nuestra etiqueta A la URL o dirección de la página con la que queremos enlazar. Será algo parecido a esto.

```
<a href="http://www.guiarte.com">ir a guiarte.com</a>
```

Sólo cabe destacar que todas las direcciones web (URLs) empiezan por **http://**, o **https://** en el caso que la página de destino se sirva mediante un servidor seguro. Este tipo de rutas que comienzan por "http" también se conocen como "rutas absolutas". Cuando enlazas con páginas que están en otros dominios necesitas usar necesariamente rutas absolutas.

Nota: La parte por la que inician las direcciones de sitios web (http://) nos indica que el protocolo por el que se accede es HTTP, el utilizado en la web. No debemos olvidarnos de colocarlas, porque si no lo hacemos, los enlaces serán tratados como enlaces locales a nuestro sitio.

Otra cosa interesante es que no tenemos que enlazar con una página web con el protocolo HTTP necesariamente. También podemos acceder a recursos a través de otros protocolos como el FTP. En tal

caso, las direcciones de los recursos no comenzarán por http:// sino por ftp://.

Enlaces a direcciones de correo

Los enlaces a direcciones de correo son aquellos que al pincharlos nos abre un nuevo mensaje de correo electrónico dirigido a una dirección de mail determinada. Estos enlaces son muy habituales en las páginas web y resultan la manera más rápida de ofrecer al visitante una vía para el contacto con el propietario de la página.

Para colocar un enlace dirigido hacia una dirección de correo colocamos **mailto:** en el atributo href del enlace, seguido de la dirección de correo a la que se debe dirigir el enlace.

```
<a href="mailto:eugim@desarrolloweb.com">eugim@desarrolloweb.com</a>
```

Este enlace se puede ver en funcionamiento aquí: miguel@desarrolloweb.com

Consejo: Cuando coloques enlaces a direcciones de correo procura indicar en el contenido del enlace (lo que hay entre A y su cierre) la dirección de correo a la que se debe escribir. Esto es porque si un usuario no tiene configurado un programa de correo en su ordenador no podrá enviar mensajes, pero por lo menos podrá copiar la dirección de mail y escribir el correo a través de otro ordenador o un sistema web-mail.

Además de la dirección de correo del destinatario, también podemos colocar en el enlace el asunto del mensaje. Esto se consigue colocando después de la dirección de correo un interrogante, la palabra subject, un signo igual (=) y el asunto en concreto.

```
<a href="mailto:eugim@desarrolloweb.com?subject=contacto a través de la pagina">eugim@desarrolloweb.com</a>
```

Podemos colocar otros atributos del mensaje con una sintaxis parecida. En este caso indicamos también que el correo debe ir con copia a colabora@desarrolloweb.com.

```
<a href="mailto:miguel@desarrolloweb.com?subject=contacto a través de la pagina&cc=colabora@desarrolloweb.com">miguel@desarrolloweb.com</a>
```

Nota: El visitante de la página necesitará tener configurada una cuenta de correo electrónico en su sistema para enviar los mensajes. Lógicamente, si no tiene servicio de correo en el ordenador no se podrán enviar los mensajes y este sistema de contacto con el visitante no funcionará. Más adelante abordaremos el tema de los [formularios](#), mediante los cuales seremos capaces de implementar una serie de campos donde solicitar información de todo tipo, que luego se pueda enviar por email. Es una manera más usable en lo que respecta al contacto con el usuario. Ya fuera de este manual también tenemos un artículo en desarrolloweb que hace un análisis de las alternativas que existen para [contacto con el navegante](#).

Enlaces con archivos

Este no es un tipo de enlace propiamente dicho, pero lo señalamos aquí porque son un tipo de enlaces muy habitual y que presenta alguna complicación para el usuario novato.

El mecanismo es el mismo que hemos conocido en los enlaces locales y los enlaces remotos, con la única particularidad de que en vez de estar dirigidos hacia una página web está dirigido hacia un archivo de otro tipo.

Si queremos enlazar con un archivo `mi_fichero.zip` que se encuentra en el mismo directorio que la página se escribiría un enlace así.

```
<a href="mi_fichero.zip">Descarga mi_fichero.zip</a>
```

Si pinchamos un enlace de este tipo nuestro navegador descargará el fichero, haciendo la pregunta típica de "Qué queremos hacer con el archivo. Abrirlo o guardarlo en disco".

Consejo: No colocar en Internet archivos ejecutables directamente, sino archivos comprimidos. Por dos razones:

1. El archivo ocupará menos, con lo que será más rápida su transferencia.
2. Al preguntar al usuario lo que desea hacer con el fichero le ofrece la opción de abrirlo y guardarlo en disco. Nosotros generalmente desearemos que el usuario lo guarde en disco y no lo ejecute hasta que lo tenga en su disco duro. Si se decide a abrirlo en vez de guardarlo simplemente lo pondrá en marcha y cuando lo pare no se quedará guardado en su sistema. Si los archivos están comprimidos obligaremos al usuario a descomprimirlos en su disco duro antes de ponerlos en marcha, con lo que nos aseguramos que el usuario lo guarde en su ordenador antes de ejecutarlo.

Si queremos enlazar hacia otro tipo de archivo como un **PDF** o un **mundo VRML** (Realidad virtual para Internet) lo seguimos haciendo de la misma manera. El navegador, si reconoce el tipo de archivo, es el responsable de abrirlo utilizando el conector adecuado para ello. Así, si por ejemplo enlazamos con un PDF pondrá el programa Acrobat Reader en funcionamiento para mostrar los contenidos. Si enlazamos con un mundo VRML pondrá en marcha el plug-in que el usuario tenga instalado para ver los mundos virtuales.

Este sería un ejemplo de enlace a un documento PDF.

```
<a href="mi_documento.pdf">Descarga el PDF</a>
```

Este artículo es obra de *Miguel Angel Alvarez*
Fue publicado por primera vez en 29/10/2001
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Imágenes y formatos gráficos

Veremos todo lo que los creadores de webs deben conocer sobre las imágenes, no sólo cómo incluir imágenes en las páginas, sino también qué formatos gráficos son adecuados en cada caso y cómo podemos optimizar las imágenes para reducir el tiempo de carga de las webs.

Imágenes en HTML

Vemos cómo colocar una imagen en una página web y algunos atributos básicos para asignarle estilos a las imágenes en HTML.

Sin duda uno de los aspectos más vistosos y atractivos de las páginas web es el grafismo. La introducción en nuestro texto de imágenes puede ayudarnos a explicar más fácilmente nuestra información y darle un aire mucho más estético. El abuso no obstante puede conducirnos a una sobrecarga que se traduce en una distracción para el navegante, quien tendrá más dificultad en encontrar la información necesaria.

El uso de imágenes también tiene que ser realizado con cuidado porque aumentan el tiempo de carga de la página, lo que puede ser de un efecto nefasto si nuestro visitante no tiene una buena conexión o si es un poco impaciente. Por ello es recomendable siempre optimizar las imágenes para Internet, haciendo que su tamaño en bytes sea lo mínimo posible, para facilitar la descarga, pero sin que ello comprometa mucho su calidad.

En este capítulo no explicaremos como crear ni tratar las imágenes, únicamente diremos que para ello se utilizan aplicaciones como [Paint Shop Pro](#), [Photoshop](#) o [Gimp](#). Tampoco explicaremos las particularidades de cada tipo de archivo: GIF, JPG o PNG y la forma de optimizar nuestras imágenes. Un capítulo posterior al respecto será dedicado a este menester: [Formatos gráficos para páginas web](#).

Las imágenes son almacenadas en forma de archivos, principalmente GIF (para dibujos) o JPG (para fotos). Estos archivos los podemos obtener desde diversas vías, como por ejemplo nuestra cámara digital, aunque también pueden ser creados por nosotros mismos con algún editor gráfico o pueden ser descargados gratuitamente en sitios web especializados.

Así pues, en estos primeros capítulos nos limitaremos a explicar como insertar y alinear debidamente en nuestra página una imagen ya creada.

La etiqueta que utilizaremos para insertar una imagen es IMG (image). Esta etiqueta no posee su cierre correspondiente y en ella hemos de especificar obligatoriamente el paradero de nuestro archivo gráfico mediante el atributo src (source).

La sintaxis queda entonces de la siguiente forma:

```

```

Para expresar el camino, lo haremos de la misma [forma que vimos para los enlaces](#). Las reglas siguen siendo las mismas, lo único que cambia es que, en lugar de una página destino, el destino es un archivo gráfico. En el código anterior estamos enlazando con un archivo con extensión .jpg, pero podrá ser otro tipo de archivo como .gif o .png, tal como se explica en el mencionado artículo sobre los [formatos gráficos permitidos en una página web](#).

Aparte de este atributo, indispensable obviamente para la visualización de la imagen, la etiqueta IMG nos propone otra serie de atributos de mayor o menor utilidad, que listamos a continuación:

Atributo alt

Dentro de las comillas de este atributo colocaremos una brevísima descripción de la imagen. Esta etiqueta no es indispensable pero presenta varias utilidades. La sintaxis te quedaría de esta manera:

```

```

Primeramente, sirve para el posicionamiento de la página en buscadores. De los atributos alt el buscador puede extraer palabras clave y le ayuda a entender qué función o contenido tiene la imagen, y por lo tanto la página.

Otra utilidad importante la encontramos en determinadas aplicaciones, usadas por personas con discapacidad. Navegadores para ciegos, por ejemplo, no muestran las imágenes y por tanto los alt ofrecen la posibilidad de leerlas. Nunca está de más pensar en crear páginas accesibles.

Por último, aunque ya menos importante en 2016, durante el proceso de carga de la página y cuando la imagen no ha sido todavía cargada, el navegador podría mostrar esta descripción, con lo que el navegante se puede hacer una idea de lo que va en ese lugar. Si hubo problemas de conexión y no se pudo mostrar la imagen, también podría usarse ese alt para mostrar al menos su descripción. En el pasado incluso era normal que algunos usuarios navegasen por la red con una opción del navegador que desactiva el muestreo de imágenes, con lo que tales personas podrán siempre saber de qué se trata el gráfico y eventualmente cambiar a modo con imágenes para visualizarla.

En general podemos considerar como aconsejable el uso de este atributo, salvo para imágenes de poca importancia. Si la imagen es usada como cuerpo de un enlace todavía se hace más indispensable.

Atributos height y width

Estos atributos definen la altura y anchura respectivamente de la imagen en píxeles. Aunque estas dimensiones forman parte del estilo de la imagen, y por tanto podrían ir en el CSS, todavía puede ser interesante definir las dentro del HTML. De nuevo, en 2016 ya no es tan indispensable, puesto que muchos sitios creados con "[Responsive Web Design](#)" prefieren que las imágenes se adapten al tamaño de la pantalla donde se va a visualizar.

Todos los archivos gráficos poseen unas dimensiones de ancho y alto. Estas dimensiones pueden obtenerse a partir del propio diseñador gráfico o bien haciendo clic con el botón derecho sobre la imagen, vista desde el explorador de archivos de tu ordenador, para luego elegir "propiedades" o "información de la imagen" sobre el menú que se despliega.

Un ejemplo de etiqueta IMG con sus valores de anchura y altura declarados te quedaría así:

```

```

Aunque este punto actualmente no tiene tanta importancia, puesto que ahora contamos con conexiones más rápidas, el hecho de explicitar en nuestro código las dimensiones de nuestras imágenes ayuda al navegador a confeccionar la página de la forma que nosotros deseamos antes incluso de que las imágenes hayan sido descargadas. Cuando las dimensiones de las imágenes han sido proporcionadas, durante el proceso de carga, el navegador reservará el espacio correspondiente a cada imagen creando una maquetación correcta. El usuario podrá comenzar a leer tranquilamente el texto sin que este se mueva de un lado a otro cada vez que una imagen se cargue.

Además de esta utilidad, el alterar los valores de estos dos atributos, es una forma inmediata de redimensionar nuestra imagen. Este tipo de utilidad no es siempre aconsejable dado que, si lo que pretendemos es aumentar el tamaño, la pérdida de calidad de la imagen será sensible. Inversamente, si deseamos disminuir su tamaño, estaremos usando un archivo más pesado en KB de lo necesario para la imagen que estamos mostrando con lo que aumentamos el tiempo de descarga de nuestro documento innecesariamente.

Nota: Como ves, muchas cosas han cambiado sobre el tema de las imágenes en 2016. Este último punto merece una mención especial, puesto que en los últimos años han aparecido (y cada vez son más comunes) pantallas de una resolución de píxeles mayor. Son pantallas donde un píxel lógico se representa con varios píxeles físicos. A esto se llama la densidad de píxeles. Teléfonos móviles de alta gama y ordenadores también de alta gama suelen tener este tipo de pantallas para conseguir una mayor nitidez. En esos casos, aunque una imagen se reduzca y no se vea a su tamaño natural (por ejemplo, una imagen de 1000 píxel se redimensiona para que ocupe solo 400 píxel), todos los puntos de la imagen servirán para que se vean con mayor nitidez en las pantallas grandes. Si te interesa profundizar sobre este punto te recomendamos la lectura de los artículos [Imágenes responsive](#) o [Mejorar la experiencia de usuario en sitios Responsive](#).

Es importante hacer hincapié en este punto ya que muchos debutantes tienen esa mala costumbre de crear gráficos pequeños redimensionando la imagen por medio de estos atributos a partir de archivos de tamaño descomunal. Hay que pensar que el tamaño de una imagen con unas dimensiones de la mitad no se reduce a la mitad, sino que resulta ser aproximadamente 4 veces inferior.

Imágenes que son enlaces y el atributo border

Ni que decir tiene que una imagen, lo mismo que un texto, puede servir de enlace. Vista la estructura de los [enlaces en HTML](#), podemos muy fácilmente adivinar el tipo de código necesario:

```
<a href="archivo.html"></a>
```

El problema de hacer esto en ciertos navegadores es que se crea un borde en la imagen, del mismo color que el color configurado para los enlaces, lo que suele ser un efecto poco deseado.

Sin embargo, en HTML podemos indicar que una imagen tenga borde. Mediante el atributo "border" se define el tamaño en píxeles del cuadro que rodea la imagen. De esta forma podemos recuadrar nuestra imagen si lo deseamos. No es algo que se use mucho, pero resulta particularmente útil cuando deseamos eliminar el borde que aparece cuando la imagen sirve de enlace. En dicho caso tendremos que especificar `border="0"`.

Nota: El atributo border ya no es tan necesario, porque los enlaces que se hacen con contenido de imágenes ya no colocan ese borde feo en los gráficos. Esto en navegadores modernos, por lo que podría darse el caso que sí nos apareciera el borde en algunos casos. Aunque de cualquier modo, ese borde se puede eliminar igualmente con CSS y será la manera correcta de hacerlo, porque no deja de ser un estilo.

Atributos `vspace` y `hspace`

Sirven para indicar el espacio libre, en píxeles, que tiene que colocarse entre la imagen y los otros elementos que la rodean, como texto, otras imágenes, etc. Estos atributos forman parte también de la responsabilidad de las CSS, así que no sería recomendable usarlos.

Atributo `lowsrc`

Con este atributo podemos indicar un archivo de la imagen de baja resolución. Cuando el navegador detecta que la imagen tiene este atributo primero descarga y muestra la imagen de baja resolución (que ocupa muy poco y que se transfiere muy rápido). Posteriormente descarga y muestra la imagen de resolución adecuada (señalada con el atributo `src`, que se supone que ocupará más y será más lenta de transferir).

Este atributo está en desuso, aunque supone una ventaja considerable para que la descarga inicial de la web se realice más rápido y que un visitante pueda ver una muestra de la imagen mientras se descarga la imagen real. Lee el artículo de [imágenes con la etiqueta picture](#) para poder ver una alternativa más moderna.

Ejemplo práctico

Resultará obvio para los lectores hacer ahora una página que contenga una imagen varias veces repetida pero con distintos atributos.

- Una de las veces que salga debe mostrarse con su tamaño original y con un borde de 3 píxeles.
- En otra ocasión la imagen aparecerá sin borde, con su misma altura y con una anchura superior a la original
- También mostraremos la imagen sin borde, con su misma anchura y con una altura superior a la original
- Por último, mostraremos la imagen con una altura y anchura mayores que las originales, pero proporcionalmente igual que antes.

| Vamos a utilizar esta imagen para hacer el ejercicio: | |  |

Las dimensiones originales de la imagen son 28x21, así que este sería el código fuente:

```

<br>
<br>

<br>
<br>

<br>
<br>

```

Se puede ver en la siguiente imagen una muestra sobre cómo quedaría ese código al visualizarse en un navegador. Observa como se produce en algunos casos una deformación de las imágenes, debido a por un cambio no proporcional en las dimensiones.



Nota: A lo largo de los próximos artículos veremos muchas otras cosas sobre imágenes, pero si lo deseas, también puedes acceder a un vídeo donde se trata lo visto anteriormente y muchas otras cosas adicionales sobre las imágenes y la creación de webs: [Videotutorial HTML: imágenes](#).

Si quieres aprender a aplicar algunos estilos adicionales a las imágenes, usando solamente HTML puedes leer el artículo [Alineación de imágenes con HTML](#). Aunque hay que advertir que no es lo más recomendable, ya que sabemos que los estilos deberían indicarse con [CSS](#). Nosotros continuaremos el [Manual de HTML](#) hablando de [formatos gráficos compatibles con la web](#).

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 19/11/2001
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Formatos gráficos para páginas web

Presenta los formatos gráficos utilizados en las páginas web, el GIF, el JPG y PNG. Hace hincapié en los dos primeros, resumiendo sus características y enseñando a optimizar los ficheros.

El componente gráfico de las páginas web tiene mucha importancia, es el que hace que estas sean vistosas y el que nos permite aplicar nuestra creatividad para hacer del diseño de sitios una tarea agradable. Es también una herramienta para acercar los sitios al mundo donde vivimos, si embargo, es también el causante de errores graves en las páginas y hacer de estas, en algunos casos, un martirio para el visitante.

Las nociones básicas para el uso de archivos gráficos son sencillas, conocerlas, aunque sea ligeramente, nos ayudará a crear sitios agradables y rápidos. No cometer errores en el uso de las imágenes es fundamental, aunque no seas un diseñador y las imágenes que utilices sean feas, utilízalas bien y así estarás haciendo más agradable la visita a tus páginas.

Tipos de archivos

En Internet se utilizan principalmente dos tipos de archivos gráficos GIF y JPG, pensados especialmente para optimizar el tamaño que ocupan en disco, ya que los archivos pequeños se transmiten más rápidamente por la Red.

El formato de archivo GIF se usa para las imágenes que tengan dibujos, mientras que el formato JPG se usa para las fotografías. Los dos comprimen las imágenes para guardarlas. La forma de comprimir la imagen que utiliza cada formato es lo que los hace ideales para unos u otros propósitos.

Adicionalmente, se puede usar un tercer formato gráfico en las páginas web, el PNG. Este formato no tiene tanta aceptación como el GIF o JPG por varias razones, entre las que destacan el desconocimiento del formato por parte de los desarrolladores, que las herramientas habituales para tratar gráficos (como por ejemplo Photoshop) generalmente no lo soportaban y que los navegadores antiguos también tienen problemas para visualizarlas.

Sin embargo, el formato se comporta muy bien en cuanto a compresión y calidad del gráfico conseguido, por lo que resulta muy útil en infinidad de casos. Todos estos problemas han pasado y ya sólo Internet Explorer 6 tiene algunos fallos cuando trata con PNG, pero la aceptación actual es más que suficiente para incorporarlo a nuestras posibilidades reales de trabajo con formatos y optimización de archivos.

A continuación se puede ver una tabla comparativa de las principales características de los formatos gráficos para crear páginas web:

Formatos gráficos para páginas web		
GIF	JPG	PNG
<ul style="list-style-type: none"> - Compresión sin pérdida - Comprime bien los dibujos - Paleta de colores variable - Hasta 256 colores - Permite transparencia - Permite animación - Alta compatibilidad 	<ul style="list-style-type: none"> - Compresión con pérdida - Comprime bien las fotos - Paleta de color real - Hasta 16 Millones colores - Sin transparencia - Sin animación - Alta compatibilidad 	<ul style="list-style-type: none"> - Compresión sin pérdida - Comprime bien los dibujos - Paleta de colores variable - Hasta millones de colores - Permite transparencia - Sin animación - Menor compatibilidad
OPTIMIZACIÓN: <ul style="list-style-type: none"> - Reducir paleta de colores 	OPTIMIZACIÓN: <ul style="list-style-type: none"> - Alterar calidad de la imagen 	OPTIMIZACIÓN: <ul style="list-style-type: none"> - Reducir paleta y más

Formato GIF

Aparte de ser un archivo ideal para las imágenes que estén dibujadas tiene muchas otras características que son importantes y útiles.

Compresión: Es muy buena para dibujos, como ya hemos dicho. Incluso puede ser interesante si la imagen es muy pequeña, aunque sea una foto.

Transparencia: es una utilidad para definir ciertas partes del dibujo como transparentes. De este modo podemos colocar las imágenes sobre distintos fondos sin que se vea el cuadrado donde está inscrito el dibujo, viendose en cambio la silueta del dibujo en cuestión.

Para crear un gif transparente debemos utilizar un programa de diseño gráfico, con el podemos indicar qué colores del dibujo queremos que sean transparentes. Generalmente, definimos la transparencia cuando vamos a guardar el gráfico.

Colores: Con este formato gráfico podemos utilizar paletas, conjuntos, de 256 colores o menos. Este es un detalle muy importante, puesto que cuantos menos colores utilicemos en la imagen, por lo general, menos ocupará el archivo. En ocasiones, aunque utilicemos menos colores en un gráfico, este no pierde mucho en calidad, llegando a ser inapreciable a la vista.

En algunos programas podemos modificar la cantidad de colores al guardar el archivo, en otros lo hacemos mientras creamos el gráfico.



En esta imagen, tomada con distintas paletas de colores, se puede apreciar como con pocos colores se ve bien el gráfico y como pierde un poco a medida que le restamos colores.

Formato JPG

Veamos ahora cuales son las características fundamentales del formato JPG:

Compresión: Tal como hemos dicho anteriormente, su algoritmo de compresión hace ideal este formato para guardar fotografías. Además, con JPG podemos definir la calidad de la imagen, con calidad baja el fichero ocupará menos, y viceversa.

Transparencia: Este formato no tiene posibilidad de crear áreas transparentes. Si deseamos colocar una imagen con un área que parezca transparente procederemos así: con nuestro programa de diseño gráfico haremos que el fondo de la imagen sea el mismo que el de la página donde queremos colocarla. En muchos casos los fondos de la imagen y la página parecerán el mismo.

Colores: JPG trabaja siempre con 16 millones de colores, ideal para fotografías.

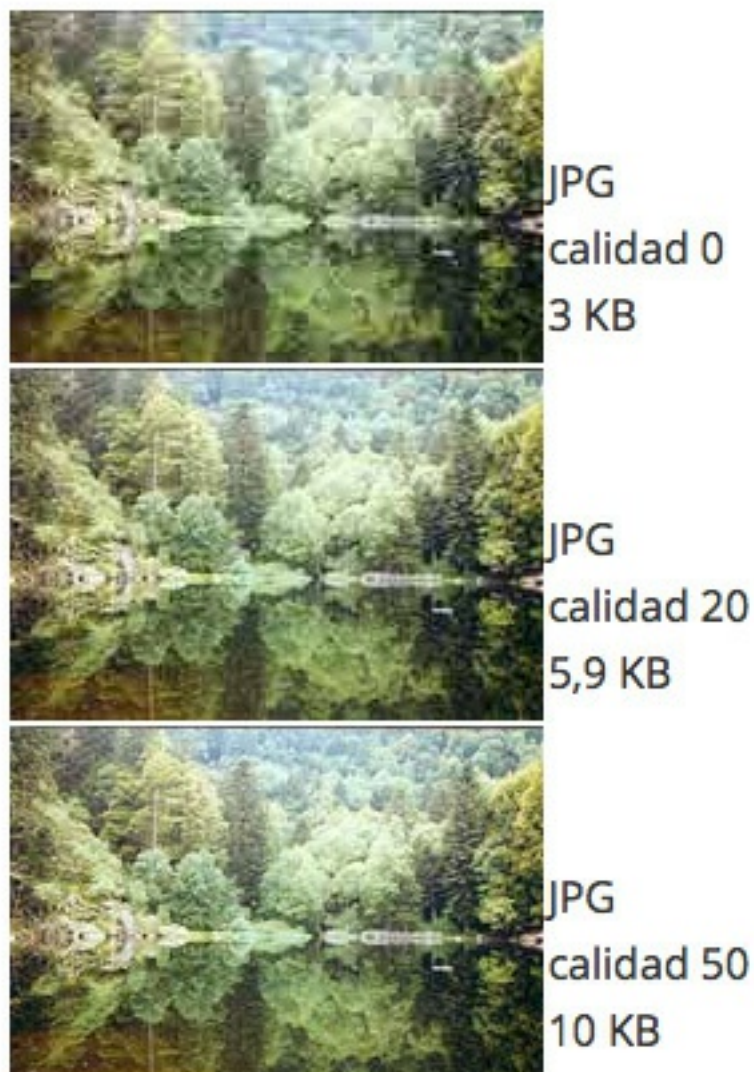
Optimizar ficheros:

Para que las imágenes ocupen lo menos posible y se transfieran rápidamente por la Red debemos aprender a optimizar los ficheros gráficos. Para ello debemos hacer lo siguiente:

Para los archivos GIF: Reduciremos el número de colores de nuestra paleta. Esto se hace con nuestro editor gráfico, en muchos casos podremos hacerlo al guardar el archivo.



Para los archivos JPG: Ajustaremos la calidad del archivo cuando lo estemos guardando. Este formato nos permite bajar mucho la calidad de la imagen sin que esta pierda mucho en su aspecto visual.



Es imprescindible disponer para optimizar la imagen de una herramienta buena que nos permita configurar estas características de la imagen con libertad y fácilmente. Photoshop incorpora una opción que se llama "Guardar para Web" con la que podemos definir los colores del gif, calidad del JPG y otras opciones en varias muestras a la vez. Así con todas las opciones configurables, viendo los resultados a la vez que el tamaño del archivo podemos optimizar la imagen de una manera precisa con los resultados que deseamos.

También existen en el mercado otros programas que nos permiten optimizar estas imágenes de manera sorprendente. Una vez hemos creado la imagen la pasamos por estos programas y nos comprimen aun más el archivo, haciéndolo rápido de transferir y, por tanto, más optimo para Internet. Al ser estas utilidades tan especializadas los resultados suelen ser mejores que con los programas de edición gráfica.

Ejemplos de optimizadores gráficos: [WebGraphics Optimizer](#) [ProJPG](#), [GIF Imantion](#) Y con versiones Online: [GIF Wizard](#)

Nota: Si te interesa reforzar todos los conceptos tratados en este artículo y ver cómo optimizamos nosotros imágenes para la web, te recomendamos ver el [Vídeo sobre Formatos gráficos para páginas web](#).

Este artículo es obra de *Miguel Angel Alvarez*
Fue publicado por primera vez en 01/01/2001
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Mapas de imágenes con HTML

Explicamos detalladamente el proceso para crear mapas de imágenes, osea, imágenes que tienes varios enlaces asociados en distintas áreas.

En capítulos anteriores hemos podido adentrarnos en el elemento básico de navegación del web: El enlace hipertexto. Hemos visto que estos enlaces son palabras, textos o imágenes que, al pinchar sobre ellos, nos envían a otras páginas o zonas.

Los mapas de imágenes es un nuevo planteamiento de navegación que incorpora una serie de enlaces dentro de una misma imagen. Estos enlaces son definidos por figuras geométricas y funcionan exactamente del mismo modo que los otros enlaces. Podéis [ver el funcionamiento de uno en este enlace](#).

En un principio, estos mapas no eran directamente reconocidos por los navegadores y recurrían a [tecnologías de lado del servidor](#) para ser visualizados. Hoy en día pueden ser implementados por medio de código HTML tal y como veremos en este capítulo.

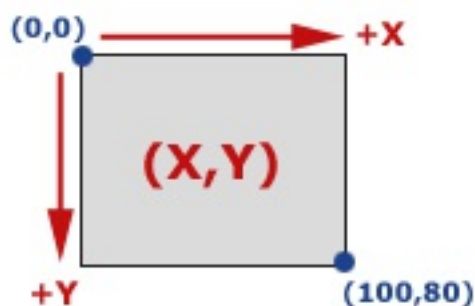
Nota: Los mapas de imágenes, aunque populares en otra época y todavía disponibles en el estándar HTML5, hoy prácticamente no se usan. Si estás leyendo el [Manual de HTML](#) para comenzar a desarrollar páginas web y tienes interés en avanzar rápidamente para luego introducirte en lenguajes también necesarios como [CSS](#), te recomendamos saltarte este artículo.

Podemos utilizar estos mapas, por ejemplo, en portadas donde damos a conocer cada una de las secciones del sitio por medio de una imagen. También puede ser muy práctico en mapas geográficos donde cada ciudad, provincia o punto cualquiera representa un enlace a una página.

En cualquier caso, el uso de estos mapas ha de estar sistemáticamente acompañado de un texto explicativo que dé a conocer al usuario la posibilidad de hacer clic sobre los distintos puntos de la imagen. Frases como "Haz clic sobre tal icono para acceder a tal información" resultan muy indicativas a la hora de hacer intuitiva la navegación por los mapas de imágenes. Por otro lado, no esta de más introducir esa misma explicación en el [atributo alt de la imagen](#).

Así pues, un mapa de imagen esta compuesto de dos partes:

- La imagen propiamente dicha que estará situada como de costumbre dentro de la etiqueta BODY de nuestro documento HTML.
- Un código, situado en el interior de la etiqueta MAP, que delimitara por medio de líneas geométricas imaginarias cada una de las áreas de los enlaces presentados en la imagen.



El recuadro es una supuesta imagen de 100x80 pixels

Las líneas geométricas que delimitan los enlaces, es decir, las áreas de los enlaces, han de ser definidas por medio de coordenadas. Cada imagen es definida por unas dimensiones de ancho (X) y alto (Y) y cada punto de la imagen puede ser definido por tanto diciendo a qué altura (x) y anchura (y) nos encontramos. De este modo, la esquina superior izquierda corresponde a la posición 0,0 y la esquina inferior derecha corresponde a las coordenadas X,Y. Si deseamos saber qué coordenadas corresponden a un punto concreto de nuestra imagen, lo mejor es utilizar un programa de diseño gráfico como Photoshop o Paint Shop Pro.

La mejor forma de explicar el funcionamiento de este tipo de mapas es a partir de un ejemplo práctico. Supongamos que tenemos una imagen con un mapa como esta:



Pulsa en los círculos para acceder a las secciones!

Dentro de ella queremos introducir un enlace a cada uno de los elementos que la componen. Para ello, definiremos nuestros enlaces como zonas circulares de pequeño tamaño que serán distribuidas a lo largo y ancho de la imagen.

Veamos a continuación el código que utilizaremos:

```
<table border=0 width=450><tr><td align="center">
<map name="mapa1">
<area alt="Pulsa para ver la página de mis amigos" shape="CIRCLE" coords="44,36,29" href="#">
<area alt="Pulsa para ver mi novia" shape="CIRCLE" coords="140,35,31" href="#">
<area alt="Pulsa para conocer a mi Familia" shape="circle" coords="239,37,30" href="#">
<area alt="Pulsa para conocer mi trabajo" shape="CIRCLE" coords="336,36,31" href="#">
</map>

<br>
Pulsa en los círculos para acceder a las secciones!
</td></tr></table>
```

Nota: Los href de las áreas van a # Este es un ejemplo parcial de utilización de los mapas, faltaría colocar los href con valores reales y no con la #. Cada uno de los enlaces de las áreas -atributo href de la etiqueta AREA- deberían llevar a una página web. El ejemplo quedaría completo si creásemos todas las páginas donde enlazar las áreas y colocásemos los href dirigidos hacia dichas páginas. Como no hemos hecho las páginas "destino" hemos colocado enlaces que no llevan a ningún sitio, que, como puedes ver, se indica con el caracter "#".

Podéis observar, tal y como hemos explicado antes, que nuestro mapa consta de dos partes principales: la imagen y la etiqueta MAP que define las áreas de cada enlace.

Cada área se indica con una etiqueta AREA que tiene los siguientes atributos:

alt Para indicar un texto que se mostrará cuando situemos el ratón en el área.

shape Indica el tipo de área.

coords Las coordenadas que definen el área. Serán un grupo de valores numéricos distintos dependiendo del tipo de área (shape) que estemos definiendo.

href Para indicar el destino del enlace correspondiente al área.

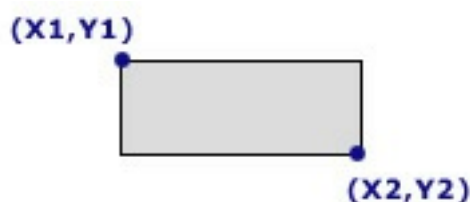
En este caso hemos utilizado unas áreas circulares (shape="CIRCLE"), que se definen indicando el centro del círculo -una coordenada (X,Y) y el radio, que es un número entero que se corresponde con el número de pixels desde el centro hasta el borde del círculo.

Tipos de áreas: shape distintas

Existen tres tipos de áreas distintas, suficientes para hacer casi cualquier tipo de figura. Las detallamos a continuación.

shape="RECT" Crea un área rectangular. Para definirla se utilizan las coordenadas de los puntos de la esquina superior izquierda y la esquina inferior derecha. Tal como están nombradas dichas coordenadas en nuestro dibujo, el área tendría la siguiente etiqueta:

```
<area shape="RECT" coords="X1,Y1,X2,Y2" href="#">
```



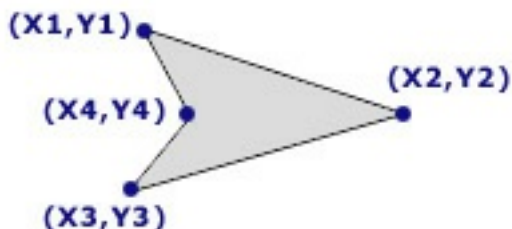
shape="CIRCLE" Crea un área circular, que se indica con la coordenada del centro del círculo y el radio. A la vista de nuestro dibujo, la etiqueta de un área circular tendría esta forma:


```
<area shape="CIRCLE" coords="X1,Y1,R" href="#">
```



shape="POLY" Este tipo de área, poligonal, es la más compleja de todas. Un polígono queda definido indicando todos sus puntos, pero atención, los tenemos que indicar en orden, siguiendo el camino marcado por el perímetro del polígono. A la vista del dibujo y los nombres que hemos dado a los puntos del polígono, la etiqueta AREA quedaría de esta forma.

```
<area shape="POLY" coords=" X1,Y1, X2,Y2, X3,Y3, X4,Y4" href="#">
```



Este artículo es obra de *Rubén Álvarez*
 Fue publicado por primera vez en 18/04/2002
 Disponible online en <http://desarrolloweb.com/articulos/21.php>

Tablas en HTML

Las tablas fueron muy importantes en una época para maquetar páginas web. Hoy lo adecuado es utilizarlas sólo para presentar información tabulada, es decir, colocada en una rejilla de filas y columnas. En los siguientes artículos aprenderemos todo sobre las tablas en HTML.

Tablas en HTML

Vemos lo que son las tablas, para que sirven y en qué casos podemos utilizarlas. Vemos la tabla más simple posible.

Una tabla en un conjunto de celdas organizadas dentro de las cuales podemos alojar distintos contenidos.

HTML dispone de una gran variedad de etiquetas para crear tablas, con sus atributos, de las cuales veremos una introducción en este artículo.

En un principio nos podría parecer que las tablas son raramente útiles y que pueden ser utilizadas principalmente para listar datos como agendas, resultados y otros datos de una forma organizada. En general, sirven para representar información tabulada, en filas y columnas. Esto es una realidad en los últimos años, desde que las tablas se han descartado para fines relacionados con la maquetación.

Nota: Durante un tiempo, gran parte de los diseñadores de páginas basaron su maquetación en este tipo de artilugios. En efecto, una tabla nos permite organizar y distribuir los espacios de la manera más adecuada. Nos puede ayudar a generar texto en columnas como los periódicos, prefijar los tamaños ocupados por distintas secciones de la página o poner de una manera sencilla un pie de foto a una imagen.

Estamos en la segunda década del 2000 y hablar de las tablas como solución para maquetación ha pasado a la historia. Las webs de primera y segunda generación utilizaban este recurso intensivamente para maquetar contenidos en páginas web, además de otras barbaridades, como los píxeles transparentes, para conseguir efectos como márgenes o espacios en blanco. Sin embargo, las webs actuales, a partir de la tercera generación, han acabado con todas esas técnicas que no hacían más que ensuciar el código fuente de las páginas web, mezclando presentación y contenido. Actualmente toda la maquetación de una página se organiza con [CSS](#), lo que nos da un mayor control de todos los elementos de la página y la posibilidad de separar todos los estilos para definir el aspecto de una web en un fichero aparte del HTML.

Por ello, en el momento actual las tablas se utilizan mucho menos que en el pasado y realmente la recomendación es usarlas solo en los casos en los que necesitemos incluir en una página información tabulada, es decir, dispuesta en filas y columnas. Todo uso basado en tablas para procurar colocar elementos en determinadas posiciones de la página sería incorrecto en las técnicas actuales de diseño de páginas web.

Como veremos a continuación, existen diversas etiquetas que se deben utilizar en una forma determinada

para la creación de tablas. Por ello, puede que en un principio nos resulte un poco complicado trabajar con estas estructuras pero, con un poco de práctica podremos crear tablas con absoluta soltura. Si deseamos mostrar datos de una manera sencilla de leer, dispuestos en filas y columnas, tarde o temprano observaremos que las tablas son la mejor solución y apreciaremos las posibilidades nos ofrecen.

Etiquetas básicas para tablas en HTML

Para empezar, nada más sencillo que por el principio: las tablas son definidas por las etiquetas `TABLE` y su cierre.

Dentro de estas dos etiquetas colocaremos todas las otras etiquetas de las tablas, hasta llegar a las celdas. Dentro de las celdas ya es permitido colocar textos e imágenes que darán el contenido a la tabla.

Las tablas son descritas por líneas de arriba a abajo (y luego por columnas de izquierda a derecha). Cada una de estas líneas, llamada fila, es definida por otra etiqueta y su cierre: `TR`

Asimismo, dentro de cada línea, habrá diferentes celdas. Cada una de estas celdas será definida por otra etiqueta: `TD`. Dentro de ésta y su cierre será donde coloquemos nuestro contenido, el contenido de cada celda.

Aquí tenéis un ejemplo de estructura de tabla:

```
<table>
<tr>
  <td>Celda 1, linea 1</td>
  <td> Celda 2, linea 1</td>
</tr>
<tr>
  <td> Celda 1, linea 2</td>
  <td> Celda 2, linea 2</td>
</tr>
</table>
```

El resultado:

Celda 1, linea 1	Celda 2, linea 1
Celda 1, linea 2	Celda 2, linea 2

Nota: Hasta aquí hemos visto todas las etiquetas que necesitamos conocer para crear tablas. Existen otras etiquetas, pero lo que podemos conseguir con ellas se puede conseguir también usando las que hemos visto.

Por poner un ejemplo, señalamos la etiqueta `TH`, que sirve para crear una celda cuyo contenido esté formateado como un título o cabecera de la tabla. En la práctica, lo que hace es poner en negrita y centrado el contenido de esa celda, lo que se puede conseguir aplicando las correspondientes etiquetas dentro de la celda. Así:

```
<td align="center"><b>contenido de la celda</b></td>
```

Sin embargo, cuando estudies la semántica del HTML te darás cuenta que, aunque la presentación sea la misma, la semántica no lo es. Esto es un tema más avanzado por el que de momento (si estás comenzando con HTML) no necesitas preocuparte. Puedes encontrar algo más de información en el artículo de las [etiquetas semánticas del HTML5](#).

Atributos para tablas, filas y celdas

A partir de esta idea simple y sencilla, las tablas adquieren otra magnitud cuando les incorporamos toda una batería de atributos aplicados sobre cada tipo de etiquetas que las componen.

En cuanto a atributos para tabla hay unos cuantos. Muchos los conoces ya de otras etiquetas, como width, height, align, etc. Hay otros que son especialmente creados para las etiquetas TABLE.

- **cellspacing:** es el espacio entre celdas de la tabla.
- **cellpadding:** es el espacio entre el borde de la celda y su contenido.
- **border:** es el número de píxeles que tendrá el borde de la tabla.
- **bordercolor:** es el rgb que le vas a asignar al borde de la tabla.

En cuanto a las etiquetas "interiores" de una tabla, nos referimos a TR y TD, ten en cuenta:

- Podemos usar prácticamente cualquier tipo de etiqueta dentro de la etiqueta TD para, de esta forma, escribir su contenido.
- Las etiquetas situadas en el interior de la celda no modifican el resto del documento.
- Las etiquetas de fuera de la celda no son tenidas en cuenta por ésta.

Así pues, podemos especificar el formato de nuestras celdas a partir de etiquetas introducidas en su interior o mediante atributos colocados dentro de la etiqueta de celda TD o bien, en algunos casos, dentro de la etiqueta TR, si deseamos que el atributo sea valido para toda la línea. La forma más útil y actual de dar forma a las celdas es a partir de las [hojas de estilo en cascada](#) que ya tendréis la oportunidad de abordar más adelante.

Veamos a continuación algunos atributos útiles para la construcción de nuestras tablas. Empecemos viendo atributos que nos permiten modificar una celda en concreto o toda una línea:

- **align:** Justifica el texto de la celda del mismo modo que si fuese el de un párrafo.
- **valign:** Podemos elegir si queremos que el texto aparezca arriba (top), en el centro (middle) o abajo (bottom) de la celda.
- **bgcolor:** Da color a la celda o línea elegida.
- **bordercolor:** Define el color del borde.

Otros atributos que pueden ser únicamente asignados a una celda y no al conjunto de celdas de una línea son:

- **background:** Nos permite colocar un fondo para la celda a partir de un enlace a una imagen.
- **height:** Define la altura de la celda en pixels o porcentaje.

- **width:** Define la anchura de la celda en pixels o porcentaje.
- **colspan:** Expande una celda horizontalmente.
- **rowspan:** Expande una celda verticalmente.

Nota: El atributo height no funciona en todos los navegadores, además, su uso no está muy extendido. Las celdas por lo general tienen el alto que necesitan para que quepa todo el contenido que se le haya insertado, es decir, crecen lo suficiente para que quepa lo que hemos colocado dentro.

El atributo width sí que funciona en todos los navegadores y lo tendréis que utilizar constantemente. Si le asignamos un ancho a la celda, el ancho será respetado y si dicha celda tiene mucho texto o cualquier otro contenido, la celda crecerá hacia abajo todo lo necesario para que quepa lo que hemos colocado.

Un matiz al último párrafo. Se trata de que si definimos una celda de un ancho 100 por ejemplo, y colocamos en la celda un contenido como una imagen que mida más de 100 píxeles, la celda crecerá en horizontal todo lo necesario para que la imagen quepa. Si el elemento, aunque más ancho, fuera divisible (como un texto) el ancho sería respetado y el texto crecería hacia abajo o lo que es lo mismo, en altura, como señalábamos en el anterior párrafo.

Estos últimos cuatro atributos descritos son de gran utilidad. Concretamente, height y width nos ayudan a definir las dimensiones de nuestras celdas de una forma absoluta (en pixels o puntos de pantalla) o de una forma relativa, es decir por porcentajes referidos al tamaño total de la tabla. Podéis leer un [artículo interesante a propósito de estas dos modalidades de diseño](#) en nuestro [manual de usabilidad](#).

A título de ejemplo:

```
<td width="80">
```

Dará una anchura de 80 pixels a la celda. Sin embargo,

```
<td width="80%">
```

Dará una anchura a la celda del 80% de la anchura de la tabla.

Hay que tener en cuenta que, definidas las dimensiones de las celdas, el navegador va a hacer lo que buenamente pueda para satisfacer al programador. Esto quiere decir que puede que en algunas ocasiones el resultado que obtengamos no sea el esperado. Concretamente, si el texto presenta una palabra excesivamente larga, puede que la anchura de la celda se vea aumentada para mantener la palabra en la misma línea. Por otra parte, si el texto resulta muy largo, la celda aumentará su altura para poder mostrar todo su contenido.

Análogamente, si por ejemplo definimos dos anchuras distintas a celdas de una misma columna, el navegador no sabrá a cuál hacer caso. Es por ello que resulta conveniente tener bien claro desde un principio como es la tabla que queremos diseñar. No está de más si la prediseñamos en papel si la complejidad es importante. El HTML resulta en general fácil pero las tablas pueden convertirse en un verdadero quebradero de cabeza si no llegamos a comprenderlas debidamente.

Los atributos rowspan y colspan son también utilizados frecuentemente. Gracias a ellos es posible expandir

celdas fusionando éstas con sus vecinas. El valor que pueden tomar estas etiquetas es numérico. El número representa la cantidad de celdas fusionadas.

Así:

```
<td colspan="2">
```

Fusionara la celda en cuestión con su vecina derecha.

Esta celda tiene un colspan="2"

Celda normal Otra celda

Del mismo modo,

```
<td rowspan="2">
```

Expandirá la celda hacia abajo fusionándose con la celda inferior.

Esta celda tiene rowspan="2", Celda
por eso tiene fusionada la Normal
celda de abajo. Otra
celda
normal

El resto de los atributos presentados presentan una utilidad y uso bastante obvios. Los dejamos a vuestra propia investigación.

Podemos continuar las explicaciones de tablas en los artículos [Atributos de la tabla y conclusión](#) y [Bordes de tabla en HTML 4](#).

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 19/12/2001
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Atributos para tablas HTML

Conocemos los atributos principales que le podemos asignar a las tablas en el lenguaje HTML de

modo general. Además vemos varios ejemplos prácticos de construcción de tablas.

En este artículo vamos a seguir hablando de tablas, con información de diverso tipo. A la parte de los atributos para tablas puedes prestarle poca atención porque básicamente lo que hacen es aplicar estilo a la propia tabla o a sus celdas y estas definiciones actualmente se recomienda incluir en el CSS. Luego pasaremos a ver las tablas anidadas y realizar unas prácticas de representación de información tabulada, que sí merece la pena estudiar para afianzar el conocimiento.

A continuación encuentras los más importantes de los atributos específicos para tablas, aunque recuerda que todos están ya en desuso desde la versión del estándar HTML5.

align: Alinea horizontalmente la tabla con respecto a su entorno. **background:** Nos permite colocar un fondo para la tabla a partir de un enlace a una imagen. **bgcolor:** Da color de fondo a la tabla. **border:** Define el número de pixels del borde principal. **bordercolor:** Define el color del borde. **cellpadding:** Define, en pixels, el espacio entre los bordes de la celda y el contenido de la misma. **cellspacing:** Define el espacio entre los bordes (en pixels). **height:** Define la altura de la tabla en pixels o porcentaje. **width:** Define la anchura de la tabla en pixels o porcentaje.

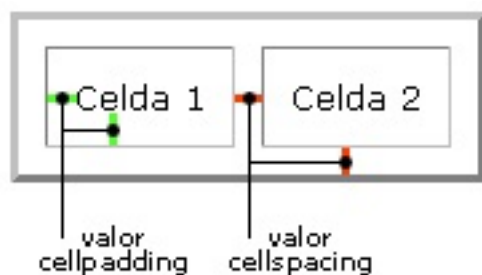
Los atributos que definen las dimensiones, height y width, funcionan de una manera análoga a la de las celdas tal y como hemos visto en el capítulo anterior. Contrariamente, el atributo align no nos permite justificar el texto de cada una de las celdas que componen la tabla, sino más bien, justificar la propia tabla con respecto a su entorno.

Vamos a poner tres ejemplos de alineado de tablas, centradas, alineadas a la derecha y a la izquierda.

Ejemplo de tabla centrada	Esta tabla está centrada (align="center"). Solo tiene una celda. Este sería un texto cualquiera colocado al lado de una tabla centrada
Ejemplo de tabla alineada a la derecha	Para que se vea el efecto de alineado a la tabla debemos colocar un texto al lado y el texto rodeará la tabla, igual que ocurría con las imágenes alineadas a un lado.
Ejemplo de tabla alineada a la izquierda	Esta tabla está alineada a la izquierda (align="left"). Solo tiene una celda.
	Para que se vea el efecto de alineado a la tabla debemos colocar un texto al lado y el texto rodeará la tabla, igual que ocurría con las imágenes alineadas a un lado.

Los atributos cellpadding y cellspacing nos ayudaran a dar a nuestra tabla un aspecto más estético. En un principio puede pareceros un poco confuso su uso pero un poco de practica será suficiente para hacerse con ellos.

En la siguiente imagen podemos ver gráficamente el significado de estos atributos.



Con un poco de práctica podéis comprobar vosotros mismos que los atributos definidos para una celda tienen prioridad con respecto a los definidos para una tabla. Podemos definir, por ejemplo, una tabla con color de fondo rojo y una de las celdas de color de fondo verde y se verá toda la tabla de color rojo menos la celda verde. Del mismo modo, podemos definir un color azul para los bordes de la tabla y hacer que una celda particular sea mostrada con un borde rojo. (Aunque esto no funcionará en todos los navegadores debido a que algunos no reconocen el atributo bordercolor).

Tablas anidadas

Muy útil también es el uso de tablas anidadas. De la misma forma que podíamos incluir listas dentro de otras listas, las tablas pueden ser incluidas dentro de otras. Así, podemos incluir una tabla dentro de la celda de otra. El modo de funcionamiento sigue siendo el mismo aunque la situación puede complicarse si el número de tablas embebidas dentro de otras es elevado.

Vamos a ver un código de anidación de tablas. Veamos primero el resultado y luego el código, así conseguiremos entenderlo mejor.

Celda de la tabla principal **Tabla anidada, celda 1** **Tabla anidada, celda 2**
Tabla anidada, celda 3 **Tabla anidada, celda 4**

Este sería el código:

```
<table cellspacing="10" cellpadding="10" border="3">
<tr>
  <td align="center">
    Celda de la tabla principal
  </td>
  <td align="center">
    <table cellspacing="2" cellpadding="2" border="1">
      <tr>
        <td>Tabla anidada, celda 1</td>
        <td>Tabla anidada, celda 2</td>
      </tr>
      <tr>
        <td>Tabla anidada, celda 3</td>
        <td>Tabla anidada, celda 4</td>
      </tr>
    </table>
  </td>
</tr>
```

```
</tr>
</table>
```

Ejemplos prácticos

Hasta aquí la información que pretendíamos transmitir sobre las tablas en HTML. Sería importante ahora realizar algún ejemplo de realización de una tabla un poco compleja. Por ejemplo la siguiente:

Animales en peligro de extinción			
Nombre	Cabezas	Previsión 2010	Previsión 2020
Ballena	6000	4000	1500
Oso Pardo	50	0	
Lince	10		
Tigre	300	210	

Se puede ver el código fuente para generar esa tabla. Pero antes intenta realizarla por ti mismo, que es esencial para poder afianzar el conocimiento. Ten en cuenta también que ciertos estilos colocados en la tabla pueden no funcionar en todos los navegadores. Además, lo que hemos repetido ya innumerables veces: los estilos forman parte de la responsabilidad del [CSS](#).

```
<table align="center" cellspacing="2" cellpadding="2" border="1" bgcolor=dddddd>
<tr>
  <td colspan="4" align="center" bgcolor="666666"><font color="#FFFFFF"><strong>Animales en peligro de extinción</strong></font></td>
</tr>
<tr bgcolor="aaaaaa">
  <td>Nombre</td>
  <td align="center">Cabezas</td>
  <td align="center">Previsión 2010</td>
  <td align="center">Previsión 2020</td>
</tr>
<tr>
  <td>Ballena</td>
  <td align="center">6000</td>
  <td align="center">4000</td>
  <td align="center">1500</td>
</tr>
<tr>
  <td>Oso Pardo</td>
  <td align="center">50</td>
  <td rowspan="2" colspan="2" align="center" bgcolor="red">0</td>
</tr>
<tr>
  <td>Lince</td>
  <td align="center">10</td>
</tr>
```

```
<td>Lince</td>
<td align="center">10</td>
</tr>
<tr>
<td>Tigre</td>
<td align="center">300</td>
<td colspan="2" align="center">210</td>
</tr>
</table>
```

Otro ejemplo de tabla con el que podemos practicar. En este caso hemos implementado una anidación de tablas, es decir, dentro de un TD hemos colocado un TABLE completo. Es un buen ejemplo para seguir aprendiendo

Climas de América del Sur			
Parte de arriba de América del Sur. Países como:	Venezuela	Parte de abajo de América del Sur. Países como:	Argentina
	Colombia		Chile
	Ecuador		Uruguay
	Perú		Paraguay
Bosque tropical, clima de sabana, clima marítimo con inviernos secos.		Climas marítimos con veranos secos, con inviernos secos, climas fríos, clima de estepa, clima desértico.	

También podemos ver su código fuente. Inténtalo tú antes de revelar la solución!

```
<table cellspacing="4" cellpadding="4" border="1" width=400 bgcolor=dddddd>
<tr>
<td colspan="2" bgcolor="666666" align="center"><font color="#FFFFFF"><strong>Climas de América del Sur</strong></font></td>
</tr>
<tr>
<td width="50%">
<table align="right" cellspacing="1" cellpadding="1" border="1">
<tr>
<td bgcolor="#cccccc" align="center">Venezuela</td>
</tr>
<tr>
<td bgcolor="#cccccc" align="center">Colombia</td>
</tr>
<tr>
<td>
```

```

        <td bgcolor="#cccccc" align="center">Ecuador</td>
    </tr>
    <tr>
        <td bgcolor="#cccccc" align="center">Perú</td>
    </tr>
</table>
Parte de arriba de América del Sur. Países como:
</td>
<td width="50%">
    <table align="right" cellspacing="1" cellpadding="1" border="1">
    <tr>
        <td bgcolor="#cccccc" align="center">Argentina</td>
    </tr>
    <tr>
        <td bgcolor="#cccccc" align="center">Chile</td>
    </tr>
    <tr>
        <td bgcolor="#cccccc" align="center">Uruguay</td>
    </tr>
    <tr>
        <td bgcolor="#cccccc" align="center">Paraguay</td>
    </tr>
    </table>
Parte de abajo de América del Sur. Países como:
</td>
</tr>
<tr>
    <td bgcolor="#358391">Bosque tropical, clima de sabana, clima marítimo con inviernos secos.</td>
    <td bgcolor="#358391">Climas marítimos con veranos secos, con inviernos secos, climas frios, clima de estepa, clima desértico.</td>
</tr>
</table>

```

No debes maquetar con tablas

En HTML (antes de la popularización del lenguaje CSS) se utilizan las tablas para maquetar páginas, aparte de mostrar información tabulada como hemos visto en este artículo. Como maquetar nos referimos al proceso por el cual se posicionan contenidos atendiendo a una estructura. Se conoce como maquetación y a la estructura muchas veces se la conoce como layout.

Con las tablas podemos generar una serie de columnas, espacios como cabecera o pie donde podemos mostrar contenidos estructurados que den la sensación de un diseño bien realizado, dividido en columnas y filas, como la maquetación de una revista o un portal. Sin embargo, **usar las tablas NO es una práctica recomendada**. La [maquetación por tablas la comentamos en un taller de HTML](#). Puedes analizar ese artículo para estudiar cómo se hacían las cosas antes y para practicar con HTML, pero hoy no se hacen las cosas así.

Lo correcto hoy, en páginas actuales que además tienen capacidades medianamente avanzadas y con información bien estructurada, se usa el lenguaje CSS y sus múltiples herramientas para producir un contenido correctamente maquetado. Si te interesa profundizar sobre este tema te recomendamos la lectura del [Manual de Maquetación CSS](#), aunque antes debes aprender el propio [CSS](#).

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 19/12/2001
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Agrupar filas o columnas de tablas con HTML 4

En HTML 4.0 podemos agrupar filas de una tabla, o columnas. Sirve para especificar estilos específicos a esas filas o columnas.

Con HTML 4 existen dos etiquetas que nos permiten agrupar filas o columnas de una tabla, para crear agrupaciones a las que se les puede definir un estilo de una sola vez, y no fila a fila o celda a celda. No son etiquetas que se usen mucho, además ahora menos que las tablas han sido sustituidas por otros tipos de herramientas para presentar la información maquetada en una rejilla, pero siguen estando dentro del estándar.

Son etiquetas para agrupación de los elementos de una tabla, que se diferencian en el tipo de elementos que pretenden agrupar.

Para agrupar conjuntos de filas se usa la etiqueta TBODY

```
<tbody>
```

:

Para agrupar conjuntos de columnas se usa la etiqueta COLGROUP.

```
<colgroup>
```

:

Ambas etiquetas tienen soporte a partir de HTML 4, por lo tanto están disponibles en todos los navegadores modernos más comúnmente utilizados.

Agrupar filas con TBODY

Hacer grupos de filas nos sirve para especificar estilos a determinadas filas de la tabla, de una sola vez. El modo de uso es el siguiente:

```
<table cellspacing="4" cellpadding="4" border="2">
<tr>
  <td>1</td>
  <td>2</td>
  <td>3</td>
</tr>
<tbody align="center" bgcolor="#ff8800" valign="top">
```

```
<tr>
  <td>4</td>
  <td>
    Esta es una celda
  <br>
  5
</td>
<td>6</td>
</tr>
<tr>
  <td>7</td>
  <td>8</td>
  <td>9</td>
</tr>
</tbody>
</table>
```

Como se puede ver, se tiene una tabla de 3 filas. Se ha marcado un grupo de dos filas con TBODY, justamente las dos últimas. Para esas filas hemos definido, mediante los atributos de la etiqueta TBODY, un centrado, un color de fondo y una alineación vertical superior. Los atributos que podríamos utilizar con la etiqueta TBODY son un grupo reducido de los que podríamos asignar a etiquetas TR o TD: align, bgcolor y valing, class, id, además de manejadores de eventos.

Podemos ver cómo se mostraría esta tabla:

1	2	3
4	Esta es una celda	6
	5	
7	8	9

De manera similar, se pueden asignar atributos de hojas de estilo en cascada, utilizando el atributo HTML style, como se puede ver a continuación:

```
<table cellspacing="2" cellpadding="2" border="2">
<tbody style="font-size:150%;">
<tr>
  <td>1</td>
  <td>2</td>
  <td>3</td>
</tr>
<tr>
  <td>4</td>
  <td>5</td>
  <td>6</td>
</tr>
</tbody>
```

```
<tr>
  <td>7</td>
  <td>8</td>
  <td>9</td>
</tr>
</table>
```

Aunque hay que decir que el grupo de atributos de hojas de estilo que son interpretados, cuando los colocamos en TBODY, es bastante reducido. Internet Explorer da mejor variedad de atributos aceptados.

Agrupar columnas con COLGROUP

Veamos ahora cómo se pueden agrupar varias filas con la etiqueta COLGROUP. El objetivo es básicamente el mismo que el de agrupar columnas, es decir, definir de una sola vez estilos específicos para un determinado conjunto de columnas de una tabla. El uso de la etiqueta, no obstante, es un poco diferente.

```
<table cellpadding="2" cellspacing="2" border="2">
<colgroup span=2 width="40">
</colgroup>
<tr>
  <td>1</td>
  <td>2</td>
  <td>3</td>
</tr>
<tr>
  <td>4</td>
  <td>5</td>
  <td>6</td>
</tr>
<tr>
  <td>7</td>
  <td>8</td>
  <td>9</td>
</tr>
</table>
```

Como se puede ver, COLGROUP se utiliza después de abrir la tabla y antes de empezar a meter los contenidos de filas y celdas.

El atributo span indica el número de columnas que se desean agrupar, empezando por la primera. En nuestro ejemplo se han agrupado las dos primeras columnas. Los otros atributos que podemos colocar en COLGROUP son los siguientes: align, id, class, style, valign, width y manejadores de eventos Javascript. Aunque Internet Explorer acepta otros atributos como bgcolor.

El ejemplo de colgroup se muestra a continuación:

1	2	3
4	5	6
7	8	9

Además, también podemos definir estilos CSS para las agrupaciones de columnas. De hecho lo correcto justamente es aplicar los estilos con CSS

Cuando queremos definir estilos para cada una de las columnas de la tabla, de manera que toda columna tenga sus estilos propios, también utilizamos COLGROUP. En este caso no se debe utilizar el atributo span, sino que se debe de agregar la etiqueta COL, una por cada columna a la que pretendemos asignar estilos. De esta manera:

```
<table cellspacing="2" cellpadding="2" border="2">
<colgroup>
<col>
<col width=100>
<col style="width: 200px;">
</colgroup>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
</table>
```

En nuestra tabla, que tenía tres columnas, hemos colocado la etiqueta COLGROUP y dentro de esta, tres etiquetas COL, cada una con sus estilos propios.

En el primer COL, como se puede ver, no hay ningún atributo. Eso quiere decir que no estoy asignando ningún estilo a la primera columna de la tabla. El segundo COL ha definido una anchura de 100 pixels. El tercer COL, también hemos definido una anchura, pero esta vez con CSS mediante el atributo style.

Los atributos CSS que acepta esta etiqueta también son bastante reducidos en Firefox, aunque Internet Explorer acepta bastantes más.

Para terminar, mostramos la tabla del último ejemplo:

12	3
45	6
78	9

Este artículo es obra de *Miguel Angel Alvarez*
Fue publicado por primera vez en 24/07/2008
Disponible online en <http://desarrolloweb.com/articulos/agrupar-filas-columnas-tabla-html.html>

Formularios en HTML

El trabajo con formularios es uno de los principales puntos que debemos aprender en HTML. Hacen posible muchas de las utilidades clave de una web, como el contacto de los creadores de las páginas con los visitantes, así como ciertos niveles de interacción básica y avanzada con el usuario.

Formularios HTML

Empezamos la explicación de la creación de formularios con el lenguaje HTML.

Hasta ahora hemos visto la forma en la que el HTML gestiona y muestra la información, esencialmente mediante texto, imágenes y enlaces. Nos queda por ver de qué forma podemos intercambiar información con nuestro visitante. Desde luego, este nuevo aspecto resulta primordial para gran cantidad de acciones que se pueden llevar a cabo mediante el Web: Comprar un artículo, rellenar una encuesta, enviar un comentario al autor...

Hemos visto anteriormente que podíamos, mediante los enlaces a direcciones de email, contactar directamente con un correo electrónico. Sin embargo, esta opción puede resultar en algunos casos poco versátil, si lo que deseamos es que el navegante nos envíe una información bien precisa y además requiere que el visitante tenga instalado en su ordenador algún correo electrónico en un programa como Outlook Express. Es por ello que el HTML propone otra solución mucho más amplia: Los formularios.

Los formularios son esas famosas cajas de texto y botones que podemos encontrar en muchas páginas web. Son muy utilizados para realizar búsquedas o bien para introducir datos personales por ejemplo en sitios de comercio electrónico. Los datos que el usuario introduce en estos campos son enviados al correo electrónico del administrador del formulario o bien a un programa que se encarga de procesarlo automáticamente.

Qué se puede hacer con un formulario

Usando HTML podemos únicamente enviar el contenido del formulario a un correo electrónico, es decir, construir un formulario con diversos campos y, a la hora pulsar el botón de enviar, generar una ventana de redacción de un email con los datos que el usuario haya escrito en cada uno de esos campos.

A menudo desearemos hacer cosas más complejas con los formularios, como que se envíe automáticamente el correo a un email sin necesidad que el contenido pase por ningún programa de email. Para ello tendremos que procesar el formulario mediante un programa.

La cosa puede resultar un poco más compleja, ya que tendremos que emplear otros lenguajes más sofisticados que el propio HTML. En este caso, la solución más sencilla es utilizar los programas prediseñados que nos ofrecen un gran número de servidores de alojamiento y que nos permiten almacenar y procesar los datos en forma de archivos u otros formatos. Si vuestras páginas están alojadas en un servidor que no os propone este tipo de ventajas, siempre podéis recurrir a servidores de terceros que ofrecen este u

otro tipo de [servicios gratuitos para webs](#). Por supuesto, existe otra alternativa que es la de aprender lenguajes como [ASP](#) o [PHP](#) que nos permitirán, entre otras cosas, el [tratamiento de formularios](#).

Así pues, en resumen, con HTML podremos construir los formularios, con diversos tipos de campos, como cajas de texto, botones de radio, cajas de selección, menús desplegables, etc. Sin embargo, debe quedar claro que desde HTML no se puede enviar directamente el correo, sino que se generará un email en el ordenador del visitante, que éste tendrá que enviar "manualmente" por medio de su programa de correo. Si queremos que el formulario se envíe automáticamente o se procese en el servidor para generar otro tipo de respuesta, necesitaremos lenguajes de programación. En este [Manual de HTML](#) nos limitaremos a explicar la creación de formularios y os proponemos buscar otras formas de proceso de los mismos a través de otros artículos en DesarrolloWeb.com.

Cómo hacer un formulario en HTML

Los formularios son definidos por medio de las etiquetas FORM y su cierre. Entre estas dos etiquetas colocaremos todos los campos y botones que componen el formulario. Dentro de esta etiqueta FORM debemos especificar algunos atributos:

action: define el tipo de acción a llevar a cabo con el formulario. Como ya hemos dicho, existen dos posibilidades:

- El formulario es enviado a una dirección de correo electrónico
- El formulario es enviado a un programa o script que procesa su contenido

En el primer caso, el contenido del formulario es enviado a la dirección de correo electrónico especificada por medio de una sintaxis de este tipo:

```
<form action="mailto:direccion@correo.com" ...>
```

Si lo que queremos es que el formulario sea procesado por un programa, hemos de especificar la dirección del archivo que contiene dicho programa. La etiqueta quedaría en este caso de la siguiente forma:

```
<form action="dirección del archivo" ...>
```

La forma en la que se expresa la localización del archivo que contiene el programa es la misma que la [vista para los enlaces](#).

method: Este atributo se encarga de especificar la forma en la que el formulario es enviado. Los dos valores posibles que puede tomar esta atributo son post y get. A efectos prácticos y, salvo que se os diga lo contrario, daremos siempre el valor post.

enctype: Se utiliza para indicar la forma en la que viajará la información que se mande por el formulario. En el caso más corriente, enviar el formulario por correo electrónico, el valor de este atributo debe de ser "text/plain". Así conseguimos que se envíe el contenido del formulario como texto plano dentro del email.

Si queremos que el formulario se procese automáticamente por un programa, generalmente no utilizaremos este atributo, de modo que tome su valor por defecto, es decir, no incluiremos enctype dentro de la etiqueta

FORM.

Ejemplo de etiqueta FORM completa

Así, para el caso más habitual -el envío del formulario por correo- la etiqueta de creación del formulario tendrá el siguiente aspecto:

```
<form action="mailto:direccion@correo.com (o nombre del archivo de proceso)" method="post" enctype="text/plain">
```

Entre esta etiqueta y su cierre colocaremos el resto de etiquetas que darán forma a nuestro formulario, las cuales serán vistas en capítulos siguientes.

Referencia: Mandar formulario por correo electrónico Los formularios se utilizan habitualmente para implementar un tipo de contacto con el navegante, que consiste en que éste pueda mandarnos sus comentarios por correo electrónico a nuestro buzón.

Para este tipo de utilización de los formularios hemos publicado hace tiempo en DesarrolloWeb.com un artículo que puede resultar muy interesante para los que deseen un referencia extremadamente rápida para construir un formulario que envíe los datos por correo electrónico al desarrollador de la página.

El artículo en cuestión se llama [contacto con el navegante](#).

Para continuar, vamos a ver cómo insertar cada uno de los campos posibles en un formulario HTML, comenzando por los [campos de texto](#).

Referencia: Si deseas la formación en vídeo, puedes ver este [videotutorial sobre formularios](#), publicado en DesarrolloWeb.com.

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 12/01/2002
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Elementos de Formularios. Campos de texto

Vemos detenidamente los distintos elementos de formulario que sirven para introducir texto.

El lenguaje HTML nos propone una gran diversidad de alternativas a la hora de crear nuestros formularios, es decir, una gran variedad de elementos para diferentes propósitos. Estas van desde la clásica caja de texto, hasta la lista de opciones en un menú desplegable, pasando por las cajas de validación, etc.

En el artículo anterior del [Manual de HTML](#) ya vimos cómo [iniciar nuestro formulario con la etiqueta FORM](#) y los distintos atributos que tenemos que utilizar para configurar su funcionamiento.

En el presente artículo veremos las etiquetas que tenemos que utilizar para crear campos de texto, que pueden ser de dos tipos. Veamos en qué consiste cada una de estas modalidades y como podemos

implementarlas en nuestro formulario.

Etiqueta INPUT para texto corto

Las cajas de texto son colocadas por medio de la etiqueta INPUT. Dentro de esta etiqueta hemos de especificar el valor de dos atributos: **type** y **name**.

La etiqueta tendrá la siguiente forma:

```
<input type="text" name="nombre">
```

De este modo expresamos nuestro deseo de crear una caja de texto cuyo contenido será llamado "nombre" (por ejemplo, en el caso de la etiqueta anterior, pero podemos poner distintos nombres a cada uno de los campos de texto que habrán en los formularios). El aspecto de este tipo de cajas es de sobra conocido, aquí lo podéis ver:

El nombre del elemento del formulario es de gran importancia para poder identificarlo en nuestro programa de procesamiento o en el mail recibido. Por otra parte, es importante indicar el atributo type, ya que, como veremos, existen otras modalidades de elementos de formulario que usan esta misma etiqueta INPUT.

El empleo de estas cajas esta fundamentalmente destinado a la toma de datos breves: palabras o conjuntos de palabras de longitud relativamente corta. Veremos más adelante que existe otra forma de tomar textos más largos a partir de otra etiqueta.

Además de estos dos atributos, esenciales para el correcto funcionamiento de nuestra etiqueta, existen otra serie de atributos que pueden resultarnos de utilidad pero que no son imprescindibles:

size: define el tamaño de la caja de texto, en número de caracteres visibles. Si al escribir el usuario llega al final de la caja, el texto que escriba a continuación también cabrá dentro del campo pero irá desfilando, a medida que se escribe, haciendo desaparecer la parte de texto que queda a la izquierda.

maxlength: indica el tamaño máximo del texto, en número de caracteres, que puede ser escrito en el campo. En caso que el campo de texto tenga definido el atributo maxlength, el navegador no permitirá escribir más caracteres en ese campo que los que hayamos indicado.

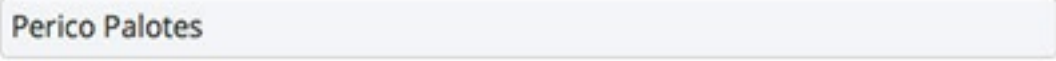
Nota: Es importante no confundir el atributo maxlength con el atributo size. Mientras size define el tamaño visible de la caja de texto, maxlength indica el tamaño máximo real del texto que se puede escribir. Podemos tener una caja de texto con un tamaño aparente (size) que es menor que el tamaño máximo (maxlength). Lo que ocurrirá en este caso es que, al escribir, si sobrepasamos el espacio marcado por size, el texto ira desfilando dentro de la caja hasta que lleguemos a su tamaño máximo definido por maxlength, momento en el cual nos será imposible continuar escribiendo.

value: en algunos casos puede resultarnos interesante asignar un valor definido al campo en cuestión. Esto

puede ayudar al usuario a rellenar más rápidamente el formulario o darle alguna idea sobre la naturaleza de datos que se requieren. Este valor inicial del campo puede ser expresado mediante el atributo value. Veamos su efecto con un ejemplo sencillo:

```
<input type="text" name="nombre" value="Perico Palotes">
```

Genera un campo de este estilo:



Veremos posteriormente que este atributo puede resultar bastante relevante en determinadas situaciones.

Nota: estamos obligados a utilizar la etiqueta FORM

Aunque esperamos que haya quedado claro a medida que se lee en estos capítulos sobre formularios, hemos querido remarcarlo para que quede muy claro: Cuando queremos utilizar, en cualquier situación elementos de formulario, debemos escribirlos siempre entre las etiquetas FORM y su cierre. De lo contrario, los elementos se verán perfectamente en Explorer pero no en Netscape. (Actualizado: en estos momentos la mayoría de los navegadores pueden interpretar bien los campos de texto sin que estén en una etiqueta FORM, sin embargo, la etiqueta FORM sigue siendo imprescindible, porque indica qué se desea hacer con los campos de texto, como el acción a realizar, y engloba qué elementos pertenecen a qué formularios) Dicho de otra forma, en Netscape no se visualizan los elementos de formulario a no ser que estén colocados entre las correspondientes etiquetas de inicio y fin de formulario. Es por ello que para mostrar un campo de texto no vale con poner la etiqueta INPUT, sino que habrá que ponerla dentro de un formulario. Así:

```
<form>
<input type="text" name="nombre" value="Perico Palotes">
</form>
```

Etiqueta INPUT, modalidad de texto oculto

Hay determinados casos en los que podemos desear esconder el texto escrito en el campo INPUT, por medio asteriscos, de manera que aporte una cierta confidencialidad. Este tipo de campos son análogos a los de texto, con una sola diferencia: remplazamos el atributo type="text" por type="password":

```
<input type="password" name="nombre">
```

En este caso, podéis comprobar que, al escribir dentro del campo, en lugar de texto veréis asteriscos.

Estos campos son ideales para la introducción de datos confidenciales, principalmente códigos de acceso o claves. Se ve en funcionamiento a continuación.



Etiqueta TEXTAREA para texto largo

Si deseamos poner a la disposición de usuario un campo de texto donde pueda escribir cómodamente sobre un espacio compuesto de varias líneas, hemos de invocar una nueva etiqueta: TEXTAREA y su cierre correspondiente.

Este tipo de campos son prácticos cuando el contenido a enviar no es un nombre, teléfono, edad o cualquier otro dato breve, sino más bien, un comentario, opinión, etc. en los que existe la posibilidad que el visitante desee rellenar varias líneas.

Dentro de la etiqueta textarea deberemos indicar, como para el caso visto anteriormente, el atributo name para asociar el contenido a un nombre que será asemejado a una variable en los programas de proceso. Además, podemos definir las dimensiones del campo a partir de los atributos siguientes:

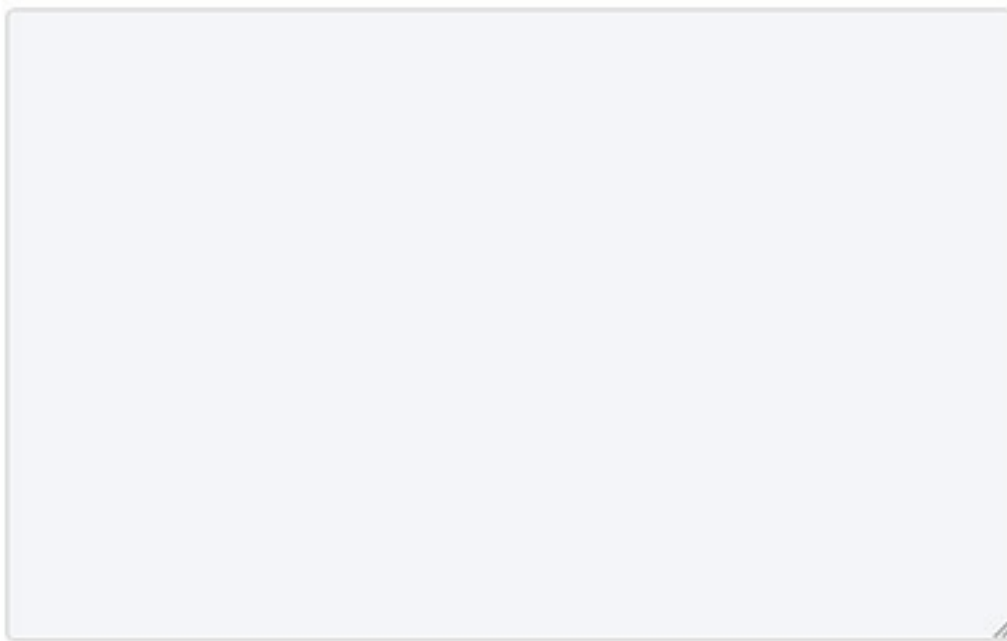
rows: define el número de líneas del campo de texto.

cols: define el número de columnas del campo de texto.

La etiqueta queda por tanto de esta forma:

```
<textarea name="comentario" rows="10" cols="40"></textarea>
```

El resultado es el siguiente:



Asimismo, es posible predefinir el contenido del campo. Para ello, no usaremos el atributo value, sino que escribiremos dentro de la etiqueta el contenido que deseamos atribuirle. Veámoslo:

```
<textarea name="comentario" rows="10" cols="40">Escribe tu comentario...</textarea>
```

Dará como resultado:



Como se podrá imaginar, los campos de texto son de vital importancia para los formularios, pero no son los únicos tipos de elementos que podemos colocar dentro de éstos. En el siguiente artículo veremos [otros tipos de elementos para formularios](#).

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 12/01/2002
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Otros elementos de formulario

Explicamos la sintaxis y el funcionamiento de las cajas y listas de selección, casillas de verificación y botones de radio.

En el [Manual de HTML](#) de DesarrolloWeb.com ya hemos publicado artículos sobre la creación de formularios y sobre los [campos de texto](#) en todas sus modalidades. Seguramente hayamos percibido que los textos son un manera muy practica de hacernos llegar la información del navegante. No obstante, en muchos casos, permitir al usuario que escriba cualquier texto permite demasiada libertad y puede que la información que éste escriba no sea la que nosotros estamos necesitando. Por otra parte, para determinados casos, los textos libres son dificilmente adaptables a programás que puedan procesarlos debidamente. Es por ello que, en determinados casos, puede resultar más efectivo proponer una elección al navegante a partir del planteamiento de una serie de opciones disponibles y definidas por nosotros.

Por ejemplo, pensemos que queremos que el usuario indique su país de residencia. En ese caso podríamos ofrecer una lista de países para que seleccione el que sea. Este mismo caso se puede aplicar a gran variedad de informaciones, como el tipo de tarjeta de crédito para un pago, la puntuación que da a un recurso, si quiere recibir o no un boletín de novedades, etc...

Este tipo de opciones predefinidas por nosotros pueden ser expresadas por medio de diferentes campos de formulario. Veamos a continuación cuales son:

Listas de opciones

Las listas de opciones son ese tipo de menús desplegables que nos permiten elegir una (o varias) de las múltiples opciones que nos proponen. Para construirlas emplearemos una etiqueta `SELECT`, con su respectivo cierre:

Como para los casos ya vistos, dentro de esta etiqueta definiremos su nombre por medio del atributo `name`. Cada opción será incluida en una línea precedida de la etiqueta `OPTION`.

Podemos ver, a partir de estas directivas, la forma más típica y sencilla de esta etiqueta:

```
<select name="estacion">
<option>Primavera</option>
<option>Verano</option>
<option>Otoño</option>
<option>Invierno</option>
</select>
```

El resultado que obtenemos mediante este código es el que se ilustra en la siguiente imagen:



Esta estructura puede verse modificada principalmente a partir de otros dos atributos:

size: Indica el número de valores mostrados a la vez en la lista. Lo típico es que no se incluya ningún valor en el atributo `size`, en ese caso tendremos un campo de opciones desplegable, pero si indicamos `size` aparecerá un campo donde veremos las opciones definidas por `size` y el resto podrán ser vistos por medio de la barra lateral de desplazamiento.

multiple: Permite la selección de más varios elementos de la lista. La elección de más de un elemento se hace como con el explorador de Windows, a partir de las teclas `ctrl` o mayúsculas (la flecha hacia arriba, también llamada `shift`). Este atributo se expresa sin valor alguno, es decir, no se utiliza con el igual: simplemente se pone para conseguir el efecto, o no se pone si queremos una lista desplegable común.

Consejo: Si es posible, no uses multiple. No recomendamos especialmente la puesta en practica de esta opción ya que el manejo de las teclas ctrl o shift para elegir varias opciones puede ser desconocido para el navegante. Evidentemente, siempre cabe la posibilidad de explicarle como funciona aunque no dejara de ser una complicación para más para el visitante.

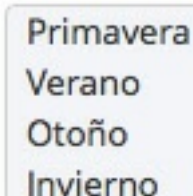
Veamos cual es el efecto producido por estos dos atributos cambiando la línea:

```
<select name="estacion">
```

por:

```
<select name="estacion" size="3" multiple>
```

La lista quedara de esta forma:



La etiqueta **OPTION** puede asimismo ser matizada por medio de **otros atributos**

selected: Del mismo modo que multiple, este atributo no toma ningún valor sino que simplemente indica que la opción que lo presenta esta elegida por defecto.

Así, si cambiamos la línea del código anterior:

```
<option>Otoño</option>
```

por:

```
<option selected>Otoño</option>
```

El resultado será:



El desplegable aparecerá
con su opción predeterminada

value: Define el valor de la opción que será enviado al programa o correo electrónico si el usuario elige esa opción. Este atributo puede resultar muy útil si el formulario es enviado a un programa para su procesamiento, puesto que a cada opción se le puede asociar un número o letra, lo cual es más fácilmente manipulable que una palabra o texto. podríamos así escribir líneas del tipo:

```
<option value="1">Primavera</option>
```

De este modo, si el usuario elige primavera, lo que le llegara al programa (o correo) es una variable llamada estacion que tendrá con valor 1. En el correo electrónico recibiríamos:

```
estacion=1
```

Botones de radio

Existe otra alternativa para plantear una elección, en este caso, obligamos al internauta a elegir únicamente una de las opciones que se le proponen.

La etiqueta empleada en este caso es INPUT en la cual tendremos el atributo type ha de tomar el valor radio. Veamos un ejemplo:

```
<input type="radio" name="estacion" value="1">Primavera  
<br>  
<input type="radio" name="estacion" value="2">Verano  
<br>  
<input type="radio" name="estacion" value="3">Otoño  
<br>  
<input type="radio" name="estacion" value="4">Invierno
```

Nota: Hay que fijarse que la etiqueta INPUT type="radio" sólo coloca la casilla pinchable en la página. Los textos que aparecen al lado, así como los saltos de línea los colocamos con el correspondiente texto en el código de la página y las etiquetas HTML que necesitamos.

El resultado es el siguiente:

☐Primavera ☐Verano ☐Otoño ☐Invierno

Como puede verse, a cada una de las opciones se le atribuye una etiqueta input dentro de la cual asignamos el mismo nombre (name) para todas las opciones y un valor (value) distinto. Si el usuario elige supuestamente Otoño, recibiremos en nuestro correo una línea tal que esta:

```
estacion=3
```

Cabe señalar que es posible preseleccionar por defecto una de las opciones. Esto puede ser conseguido por medio del atributo **checked**:

```
<input type="radio" name="estacion" value="2" checked>Verano
```

Veamos el efecto:

☐ Primavera ☒ Verano ☐ Otoño ☐ Invierno

Cajas de validación

Este tipo de elementos pueden ser activados o desactivados por el visitante por un simple clic sobre la caja en cuestión. La sintaxis utilizada es muy similar a las vistas anteriormente:

```
<input type="checkbox" name="paella">Me gusta la paella
```

El efecto:

☐ Me gusta la paella

La única diferencia fundamental es el valor adoptado por el atributo type.

Del mismo modo que para los botones de radio, podemos activar la caja por medio del atributo **checked**.

El tipo de información que llegara a nuestro correo (o al programa) será del tipo:

paella=on (u off dependiendo si ha sido activada o no).

Este artículo es obra de *Rubén Álvarez*
Fue publicado por primera vez en 12/01/2002
Disponible online en <http://desarrolloweb.com/articulos/21.php>

Envío, borrado y demás en formularios HTML

Enseñamos la manera de colocar botones de envío y borrado en formularios HTML. También conocemos los campos invisibles y los botones normales. Además, hacemos un ejemplo práctico.

Siguiendo con la explicación de todo lo relativo a formularios que estamos ofreciendo en el [Manual de HTML](#), ha llegado el momento de explicar cómo podemos hacer un botón para provocar el envío del formulario, entre otras cosas.

Como podremos imaginarnos, en formularios no solamente habrá elementos o campos donde solicitar información del usuario, sino también habrá que implementar otra serie de funciones. Concretamente, han de permitirnos su envío mediante un botón. También puede resultar práctico poder proponer un botón de

borrado o bien acompañar el formulario de datos ocultos que puedan ayudarnos en su procesamiento.

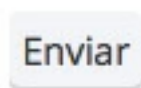
En este capítulo, para terminar la saga de formularios, daremos a conocer los medios de instalar todas estas funciones y acabaremos mostrando un ejemplo de formulario completo.

botón de envío de formulario (botón de submit)

Para dar por finalizado el proceso de relleno del formulario y hacerlo llegar a su gestor, el navegante ha de enviarlo por medio de un botón previsto a tal efecto. La construcción de dicho botón no reviste ninguna dificultad una vez familiarizados con las etiquetas INPUT ya vistas:

```
<input type="submit" value="Enviar">
```

Con este código generamos un botón como este:



Como puede verse, tan solo hemos de especificar que se trata de un botón de envío (`type="submit"`) y hemos de definir el mensaje que queremos que aparezca escrito en el botón por medio del atributo `value`. Este tipo de campos INPUT, para envío de formularios, a menudo se conocen simplemente como "botones de submit".

Nota: Al enviar el formulario se creará un mensaje con tu programa de correo, que se debe enviar con ese propio programa de correo, para que llegue al destinatario. Este es el comportamiento típico de los formularios que se programan con HTML, que requiere que el usuario tenga un programa de correo instalado y configurado para que funcione.

Una duda típica es cómo realizar el formulario para que se envíe directamente desde la página web, sin que el usuario deba tener un programa de correo, sino que se pulse el botón de enviar y se genere y envíe el mensaje automáticamente. Para ello es necesario realizar algo de programación, aparte del propio formulario en HTML, en un lenguaje avanzado, que sea del lado del servidor, como PHP, ASP... En DesarrolloWeb.com tienes todo lo que necesitas para aprender a conseguir el envío automático de correos, con explicaciones detalladas para obtener los resultados por varias vías. Te recomendamos leer el manual [Envío de formularios avanzado](#).

botón de borrado (botón de reset)

Este botón nos permitirá borrar el formulario por completo, en el caso de que el usuario desee rehacerlo desde el principio. Su estructura sintáctica es análoga a la anterior:

```
<input type="reset" value="Borrar">
```

A diferencia del botón de envío, indispensable en cualquier formulario, el botón de borrado resulta meramente optativo y no es utilizado frecuentemente. Hay que tener cuidado de no ponerlo muy cerca del

botón de envío y de distinguir claramente el uno del otro, para que ningún usuario borre el contenido del formulario que acaba de rellenar por error.

Datos ocultos (campos hidden)

En algunos casos, aparte de los propios datos rellenados por el usuario, puede resultar práctico enviar datos definidos por nosotros mismos que ayuden al programa en su procesamiento del formulario. Este tipo de datos, que no se muestran en la página pero si pueden ser detectados solicitando el código fuente, no son frecuentemente utilizados por páginas construidas en HTML, son más bien usados por páginas que emplean tecnologías de servidor. No os asustéis, veremos más adelante qué quiere decir esto. Tan solo queremos dar constancia de su existencia y de su modo creación. He aquí un ejemplo:

```
<input type=hidden name="sitio" value="www.desarrolloweb.com">
```

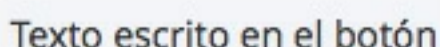
Esta etiqueta, incluida dentro de nuestro formulario, enviara un dato adicional al correo o programa encargado de la gestión del formulario. podríamos, a partir de este dato, dar a conocer al programa el origen del formulario o algún tipo de acción a llevar a cabo (una redirección por ejemplo).

Botones normales

Dentro de los formularios también podemos colocar botones normales, pulsables como cualquier otro botón. Igual que ocurre con los campos hidden, estos botones por si solos no tienen mucha utilidad pero podremos necesitarlos para realizar acciones en el futuro. Su sintaxis es la siguiente.

```
<input type=button value="Texto escrito en el botón">
```

Quedaría de esta manera:



El uso más frecuente de un botón es en la programación en el cliente. Utilizando lenguajes como [Javascript](#) podemos definir acciones a tomar cuando un visitante pulse el botón de una página web.

Ejemplo completo de formulario

Con este capítulo finalizamos el tema de formularios. Pasemos ahora a ejemplificar todo lo aprendido a partir de la creación de un formulario que consulta el grado de satisfacción de los usuarios de una línea de autobuses ficticia. El formulario está construido para que envíe los datos por correo electrónico a un buzón determinado.

Vemos el formulario en esta página. Vosotros tratar de construirlo para ver si habéis entendido bien los temas sobre formularios.

Nombre

Email

Población

Sexo ☒ Hombre ☐ Mujer Frecuencia de los viajes Comentarios sobre su satisfacción personal

☒ Deseo recibir notificación de las novedades en las líneas de autobuses.

El código del formulario se puede ver a continuación. Pero antes de analizarlo te recomendamos construir el formulario por tu propia cuenta para practicar.

```
<form action="mailto:colabora@desarrolloweb.com" method="post" enctype="text/plain">
Nombre <input type="text" name="nombre" size="30" maxlength="100">
<br>
Email <input type="text" name="email" size="25" maxlength="100" value="@">
<br>
Población <input type="text" name="poblacion" size="20" maxlength="60">
<br>
Sexo
<br>
<input type="radio" name="sexo" value="Varon" checked> Hombre
<br>
<input type="radio" name="sexo" value="Hembra"> Mujer
<br>
<br>
Frecuencia de los viajes
<br>
<select name="utilizacion">
  <option value="1">Varias veces al dia
  <option value="2">Una vez al dia
  <option value="3">Varias veces a la semana
  <option value="4">varias veces al mes
</select>
<br>
<br>
Comentarios sobre su satisfacción personal
```

```
<br>
<textarea cols="30" rows="7" name="comentarios"></textarea>
<br>
<br>
<input type="checkbox" name="recibir_info" checked> Deseo recibir notificación de las novedades en las líneas de autobuses.
<br>
<br>
<input type="submit" value="Enviar formulario">
<br>
<br>
<input type="Reset" value="Borrar todo">
</form>
```

Referencias:

Hemos publicado un taller de HTML con un formulario para valorar la página web. Muy sencillo y práctico. Puede ser interesante para afianzar estos conocimientos: [Taller con formularios](#).

Además, también te recomendamos [ver el videotutorial sobre formularios HTML](#), donde repasamos todo lo visto hasta el momento sobre la creación de formularios en páginas web.

Este artículo es obra de *Rubén Álvarez*.

Fue publicado por primera vez en 12/01/2002

Disponible online en <http://desarrolloweb.com/articulos/21.php>

Etiquetas FIELDSET y LEGEND de formularios

Las etiquetas de HTML FIELDSET y LEGEND sirven para crear bloques de elementos dentro de formularios.

En el [Manual de HTML](#) de DesarrolloWeb.com ya hemos visto varios artículos para la [creación de formularios en HTML](#). Pero nos quedaba por ver un par de etiquetas, que no siendo imprescindibles para la realización de formularios, sí nos pueden ayudar a estructurarlos, mejorando la interfaz de usuario de nuestras páginas.

Los atributos FIELDSET y LEGEND se utilizan en conjunto y sirven respectivamente para definir y etiquetar grupos lógicos de elementos de formularios. Realmente no afectan a la operativa del formulario, pero sirven para agrupar elementos en diferentes áreas, de modo que se clarifique la entrada de datos del usuario. Al formar varios grupos de elementos se puede crear una estructura mucho más fácil de asimilar por el usuario, sobre todo si se trata de formularios que tengan muchos elementos.

Etiqueta FIELDSET

La etiqueta FIELDSET sirve para agrupar los elementos. Se utiliza con su respectiva etiqueta de cierre y lo que hace es crear un recuadro que rodea a los elementos de formulario colocados dentro de ella.

Por ejemplo, se podría usar de esta manera:

```
<fieldset>
```

```
Elemento de formulario: <input type="text" name="elemento1">
<br>
Otro elemento: <input type="text" name="otro">
</fieldset>
```

Simplemente creará un cuadrado que agrupará los dos elementos del formulario incluidos dentro del FIELDSET. Podemos [ver el resultado en una página aparte](#).

Etiqueta LEGEND

LEGEND sirve para nombrar o etiquetar un grupo creado con FIELDSET. Añade simplemente una nota aclaratoria sobre qué tipo de información se está agrupando en el recuadro. Tampoco sirve para nada en especial, de no ser porque queda bonita y porque puede servir para ayudar al usuario y mejorar la interfaz y la claridad de los formularios.

La etiqueta LEGEND se coloca después de la etiqueta FIELDSET. Tiene su propia etiqueta de cierre. Entre LEGEND y su cierre colocamos el texto con el que queremos marcar el recuadro definido con FIELDSET.

A la etiqueta LEGEND se le puede poner el atributo align para indicar el lugar donde debe aparecer la leyenda. Por ejemplo podríamos indicar align="right" para que apareciera en la parte de la derecha, en lugar de la izquierda, que es donde aparece por defecto.

Veamos ahora un ejemplo sencillo de utilización de las etiquetas FIELDSET y LEGEND en conjunto.

```
<form>
<fieldset>
<legend align="right">Datos personales</legend>
Nombre: <input type="text" name="nombre">
<br>
Edad: <input type="text" name="edad" size="2">
<br>
Dirección: <input type="text" name="direccion">
</fieldset>
<br>
<fieldset>
<legend align="right">Datos de tu ordenador</legend>
Modelo de ordenador: <input type="text" name="modelo">
<br>
Sistema que te da el problema:
<select>
<option value=cpu>CPU
<option value=impresora>Impresora
</select>
</fieldset>
<br>
<fieldset>
<legend align="right">Descripción del problema</legend>
<textarea cols="55" rows="8" name="descripcion"></textarea>
</fieldset>
</form>
```

El [ejemplo en marcha tendría este aspecto](#).

Podremos comprobar como aparecen tres bloques en el formulario, producidos por tres etiquetas FIELDSET, con varios campos de formulario incluidos en cada una. Además, cada uno de los FIELDSET tienen dentro un LEGEND que sirve para nombrar con una leyenda cada uno de los tres bloques.

Etiqueta LABEL

Aunque no forma parte del objetivo de este artículo, queremos nombrar también otra etiqueta llamada LABEL que sí tiene una utilidad especial en la creación de formularios, además de la estética. Sirve para poner texto al lado de los elementos de formulario y que tal texto esté asociado al propio elemento. Ese texto, que pondremos con el tag LABEL, se asocia a un elemento concreto con el atributo FOR, colocando como valor del atributo el identificador del campo que se está asociando.

```
<label for="edad">Edad</label> <input type="text" name="edad" id="edad">
```

Como vemos, hemos creado un LABEL y hemos colocado en el atributo FOR el nombre del campo de formulario que estamos asociando a ese texto. El resultado es que el texto colocado dentro de LABEL es un elemento interactivo, al que podemos hacer clic y sería como si hiciésemos clic en el propio campo asociado al LABEL.

Para acabar, comentamos que estas etiquetas se hallan relatadas en otro artículo de DesarrolloWeb.com, con explicaciones escritas por otro autor, que podrían complementar y ampliar la presente información. Si te interesa, accede al artículo [Las nuevas etiquetas de HTML 4.0 \(2\)](#).

Este artículo es obra de *Miguel Angel Alvarez*
Fue publicado por primera vez en 06/08/2008
Disponible online en <http://desarrolloweb.com/articulos/etiquetas-fieldset-legend-formularios.html>

Otras etiquetas y nuevos estándares del HTML

En los capítulos finales de este manual de HTML nos centraremos en ver otras etiquetas que han ido apareciendo en el estándar. Además acabamos explicando algunas cosas básicas de lo que es el HTML5, la última versión de HTML.

Etiqueta Iframe

Explicamos detenidamente la etiqueta IFRAME de HTML y todos sus atributos, con algún ejemplo de uso.

Los frames (frame en inglés significa marco) son unas herramientas que han tenido una historia dilatada en el desarrollo de páginas web con HTML. De ser una etiqueta no estándar ha pasado a ser soportada por todos los navegadores y formar parte de las especificaciones de HTML, para luego retirarse de nuevo del estándar en HTML5. No obstante, ha permanecido en uso y dentro del estándar una etiqueta hermana IFRAME que vamos a ver en este artículo, que todavía hoy tiene mucha utilidad.

En concreto iframe sirve para crear un espacio dentro de la página donde se puede incrustar otra web. Es un cuadrado cuyas dimensiones debe especificar el desarrollador en la propia página, incluidas por los atributos width y height en la propia etiqueta IFRAME.

El iframe tiene asociada una página web, que se carga en el espacio y operará de manera totalmente independiente. Esa página web tendrá sus propios contenidos y estilos. Además será perfectamente funcional: si tiene enlaces se mostrarán en ese mismo espacio y si tiene scripts o aplicaciones dentro se ejecutarán también de manera autónoma en el espacio reservado al iframe.

Nota: Si andas buscando información sobre los frames tradicionales, en DesarrolloWeb.com ya hemos hablado mucho sobre ellos y aunque ya no están dentro del estándar actual del HTML todavía los navegadores los interpretan, por compatibilidad con las páginas antiguas. Puedes comenzar la lectura en el artículo [Frames en HTML](#).

En este artículo vamos a hablar la etiqueta "hermana" que es a día de hoy mucho más usada, porque resulta más útil y menos problemática que los propios frames. IFRAME fue un tag introducido en las especificaciones de HTML 4.0.

Donde resulta de utilidad el IFRAME

Iframe se utiliza en muchos contextos. Dentro de un iframe podemos mostrar contenidos de otras páginas, como si estuvieran en la nuestra, por lo que sirven para ejemplos como:

- Códigos de banner, que se invocan por medio de un iframe pidiendo los datos del banner generalmente a un servidor de banners que puede estar en otra red.
- Visualizar contenidos de terceros, como bloques de noticias o novedades que ofrecen en otras webs.
- Interfaces de usuario, en el que ciertas actividades se realizan de forma autónoma y el procesamiento está en otra página web.

Uso de iframe frente a frame

Actualmente la etiqueta iframe se utiliza más a menudo que la etiqueta frame, porque no da tantos problemas como esta. La diferencia principal está basada en que la etiqueta iframe no necesita una declaración de los espacios de los frames o frameset, porque se incrusta en el código HTML de la página. El iframe, por tanto, no provoca problemas de navegación, como los que ocurren con los frames normales si no se entra correctamente a través del frameset.

También, ya que no existe el frameset en los iframe, no adolece de los problemas del uso de frames, sobre todo a la hora en que la página es indexada en los motores de búsqueda.

Por decirlo de alguna manera, trabajar con iframe o frames flotantes es tan sencillo como hacer una tabla, que se codifica dentro de la maqueta HTML, con su espacio reservado dentro de la página. Así, la única diferencia con respecto a una tabla es que el contenido del iframe se extrae de otra página web.

Construcción de la etiqueta iframe

Como decimos, el iframe se coloca directamente en el código HTML, en el lugar donde queremos que aparezca.

Se colocaría con un código como este:

```
<iframe src="pagina_fuente.html" width=290 height=250>Texto para cuando el navegador no conoce la etiqueta iframe</iframe>
```

Como se ve, los atributos principales de iframe son la página web que se va a mostrar en el espacio y el ancho y alto del recuadro que reservemos para el frame flotante.

Como se puede ver, la etiqueta iframe tiene su correspondiente etiqueta de cierre. Todo el texto que coloquemos entre la etiqueta de inicio y la de cierre es texto alternativo, que sólo se mostrará en caso que el navegador del visitante no acepte la etiqueta iframe.

Todos los atributos de iframe

Ahora vamos a ver cuáles serían los atributos disponibles para la etiqueta iframe. No obstante, cabe ya señalar que algunos de los atributos que vamos a ver se engloban más en el terreno de los estilos y por tanto se podrían, y sería más correcto, especificar dentro de las CSS.

src: Para indicar la página web que se mostrará en el espacio del frame flotante.

width: Para definir la anchura del recuadro del iframe

height: Para definir la altura del iframe

name: Para especificar el nombre del frame, que podemos utilizar luego para referirnos a él con el target de los links, o mediante javascript.

id: Para indicar el identificador del iframe, y poder referirnos a él desde javascript.

frameborder: para definir si queremos o no que haya un borde en el frame. Los valores posibles son 0 | 1. frameborder=0 indicaría que no queremos borde y frameborder=1 que sí.

scrolling: indica si se quiere que aparezcan barras de desplazamiento para ver los contenidos del iframe completo, en el caso que no aparezcan en el espacio reservado para el iframe. Los valores posibles son: yes | no | auto. El valor "yes" es para que aparezcan siempre las barras de desplazamiento o barras de scroll, "no" sirve para que no aparezcan nunca y "auto" es para que aparezcan sólo cuando son necesarias (es el valor por defecto)

marginwidth: Para definir el margen a izquierda y derecha que debe tener la página que va dentro del iframe, con respecto al borde. Este margen va en pixels, pero prevalecerá el margen que pueda tener asignada la página web que mostremos en el frame flotante.

marginheight: lo mismo que marginwidth, pero en este caso para el tamaño del margen por la parte de arriba y abajo.

margin: para especificar alineación del frame, igual que se especifica para las imágenes.

style y class: los atributos para definir el aspecto del iframe por medio de hojas de estilo css.

Para acabar, aquí vemos otro ejemplo de iframe con unos cuantos atributos más:

```
<iframe name=miiframeflotante src="colabora.htm" width=400 height=550 frameborder="0" scrolling=yes marginwidth=2 marginheight=4 align=left>Tu navegador no soporta frames!!</iframe>
```

Este artículo es obra de *Miguel Angel Alvarez*
Fue publicado por primera vez en 17/03/2008
Disponible online en <http://desarrolloweb.com/articulos/etiqueta-iframe.html>

Las nuevas etiquetas de HTML 4.0

En este artículo veremos estas nuevas etiquetas del estandar HTML 4.0.

Cuando Internet empezaba su imparable escalada, la versión del estándar HTML que circulaba era la 2.0, el cuál siguen soportando los navegadores más actuales. Pero las herramientas de que se disponía no ofrecían un control preciso de los documentos.

Pero como por aquel entonces el objetivo de Internet estaba fundamentalmente orientado al ámbito académico y no al de diseño, no se le dio demasiada importancia a la cuestión de lanzar una versión mejorada del estándar hasta que Netscape, que por aquel entonces era la empresa líder en el sector, tomó la iniciativa de incluir nuevas etiquetas pensadas para mejorar el aspecto visual de las páginas web.

Por este motivo el IETF (Internet Engineering Task Force), Grupo de Trabajo en Ingeniería de Internet, comenzó a elaborar nuevos estándares, los cuales dieron como fruto el HTML 3.0, que resultó ser demasiado grande para las infraestructuras que había en ese momento, lo cual dificultó su aceptación.

Así pues, una serie de compañías (entre las que estaban Netscape, Sun Microsystems o Microsoft, entre otras), se unieron para crear lo que hoy se denomina W3C (o lo que es lo mismo, Consorcio para la World Wide Web), que fue fundado en octubre de 1.994 para conducir a la World Wide Web a su máximo potencial, desarrollando protocolos de uso común, para normalizar el uso de la web en todo el mundo.

El compromiso del W3C de encaminar a la Web a su máximo potencial incluye promover un alto grado de accesibilidad para las personas con discapacidades. El grupo de trabajo permanente Web Accessibility Initiative (WAI, Iniciativa para la Accesibilidad de la Red), en coordinación con organizaciones alrededor de todo el mundo, persigue la accesibilidad de la Web a través de cinco áreas de trabajo principales: Tecnología, directrices, herramientas, formación, difusión, e investigación y desarrollo.

De esta iniciativa nació el borrador de HTML 3.2 y en su versión definitiva se introdujeron cambios esenciales para las posibilidades que empezaban a ofrecer los navegadores, estas inclusiones fueron las tablas, los applets, etc.

En julio de 1.997 nace el borrador del HTML 4.0 y finalmente se aprueba en diciembre de 1.997 este estándar incluía como mejoras los marcos (frames), las hojas de estilo y la inclusión de scripts en páginas web, entre otras cosas.

Etiquetas para definir bloques de texto

Veamos una serie de etiquetas para englobar el texto cuya función indica principalmente la función que desempeña ese texto dentro de las páginas HTML. Daremos una breve descripción de cada una de ellas.

Entre el estándar del HTML 3.2 al 4.0 se introdujeron ocho nuevas etiquetas de las cuales daremos una breve explicación.

Q

La etiqueta Q actúa de forma muy parecida a BLOCKQUOTE pero con la particularidad de que añade un sangrado en párrafos más pequeños y sin necesidad de romper el párrafo.

Según el W3C, la etiqueta BLOCKQUOTE es para añadir sangrados largos y Q, para sangrados más pequeños, sin necesidad de romper el párrafo.

Nota: En el HTML 4.0 es imprescindible poner la etiqueta de apertura y la de clausura de Q

ACRONYM

Las etiquetas ACRONYM y su cierre, indican que hay un acrónimo en el texto. Un acrónimo es un pequeño texto que ayuda a explicar la estructura del texto una frase.

Nota: La etiqueta ACRONYM ha sido retirada del estándar con la versión HTML5.

INS y DEL

Utilice INS para marcar las partes de un documento que se han agregado desde la versión pasada del documento. DEL marca de manera similar un texto de un documento que se ha suprimido desde la versión anterior.

COLGROUP

Se utiliza para tener un mejor control sobre un el formato de las tablas especificando las características que comparten como: anchura, altura y alineación. Cada tabla debe tener por lo menos un COLGROUP sin especificar ninguna característica de COLGROUP. HTML 4.0 asume que una tabla contiene un solo grupo de columnas y que este contiene todas las columnas de una tabla. Por ejemplo, esto nos serviría para crear una tabla con una celda en la que puede incluirse una descripción y después seguido de check boxes para seleccionar las opciones deseadas.

Podemos ver un ejemplo de Código:

```
<TABLE> <COLGROUP span="10" width="30"> <COLGROUP span="1" width="0*"> <THEAD> <TR>... </ TABLE>
```

De esta forma, COLGROUP proporciona un formato más agradable a las celdas, sin necesidad de especificar propiedades idénticas para cada una por separado.

Nuevas etiquetas para uso en formularios

En el presente texto vamos a hablaros sobre las etiquetas del HTML 4.0 relacionadas con la creación de formularios, que muchas veces son desconocidas, pero no por ello dejan de ser interesantes. El paseo por esta nueva serie de etiquetas del HTML viene de la mano de Luciano Moreno y con revisión en 2011 de Miguel Angel Alvarez.

FIELDSET

Hasta ahora, no disponíamos de ninguna manera de agrupar visualmente varios controles, si no echábamos mano de elementos que no son del formulario, como tablas o capas (divisiones o elementos DIV, como prefieras llamarles). Ahora, si encerramos una parte de un formulario dentro de la etiqueta FIELDSET, se mostrara un rectángulo alrededor de los campos englobados por dicha etiqueta.

Además, podemos indicar un título en el rectángulo creado por FIELDSET y para ello utilizamos la etiqueta LEGEND, que admite el parámetro align="left / center / right / top / bottom", lo que nos permite alinear el título horizontal y verticalmente.

Para aclarar el aspecto de la agrupación de campos con FIELDSET podemos ver el siguiente efecto

obtenido al agrupar un par de elementos de formulario.

Datos personales

Nombre:

Edad:

Nota: Ten en cuenta que puedes aplicar estilo a esta organización de campos de formulario usando las CSS sobre la etiqueta FIELDSET, en el caso que sepamos usar las Hojas de Estilo en Cascada.

Ejemplo:

```
<form action="#" method="post" enctype="text/plain" name="miform">
  <fieldset>
    <legend align="left">
      Caja de texto
    </legend>
    pon tu nombre:
    <input type="text" size="15">
  </fieldset>
</form>
```

Referencia: Para los interesados, cabe señalar que en DesarrolloWeb.com existe un artículo que explica estas etiquetas desde otro punto de vista y con las palabras de otro autor, que se puede leer en el link [Etiquetas FIELDSET y LEGEND de formularios](#).

LABEL

Hasta no hace mucho los textos que ponemos al lado de los campos de formulario no estaban asociados a dichos campos. Es decir, el texto que colocamos al lado de un elemento de formulario, para especificar qué debe escribir el usuario en el campo, no tiene ninguna relación real con el propio elemento de formulario.

Por ejemplo, si tenemos un código como este:

```
Dirección: <input type="text" name="direccion">
```

El texto "Dirección" no está asociado para nada con el campo INPUT. Por ello, al pulsar sobre el texto "Dirección" no ocurre nada. Esto es así también con otros campos de formulario, como las cajas de checkbox o botones de radio.

```
<input type="checkbox" name="interesado"> Estoy interesado
```

Si pulsamos sobre el texto que hay al lado del campo de confirmación "Estoy interesado", ¡no sucede nada! Pero ahora, con la utilización de la etiqueta LABEL podemos conseguir que, haciendo clic en el texto "Estoy interesado", el control checkbox cambie de estado.

Ejemplo:

```
<form action="#" method="post" enctype="text/plain" name="un ejemplo más">
  <label>
    <input type="checkbox" name="email">
    Recibir email
  </label>
</form>
```

Ese ejemplo de LABEL es perfectamente válido y asocia el texto "Recibir email" al campo checkbox de formulario, de manera que si pulsamos sobre "Recibir email" cambiará el estado del campo checkbox asociado. Sin embargo, en la etiqueta LABEL podemos utilizar un atributo llamado FOR, que sirve para indicar explícitamente a qué campo de formulario se está asociando ese texto. Para ello colocamos como valor del atributo FOR el identificador del campo que estamos asociando a ese LABEL. Esto nos permite una mayor versatilidad a la hora de componer el HTML de nuestra página. Veamos el siguiente ejemplo:

```
<form>
  <label for="hombre">Hombre</label>
  <input type="radio" name="sexo" id="hombre" value="hombre">
  <br>
  <label for="mujer">Mujer</label>
  <input type="radio" name="sexo" id="mujer" value="mujer">
</form>
```

Si ponemos este ejemplo en marcha, veremos que pulsando en el texto "Hombre" se activa el botón de radio "hombre". Del mismo modo, si pulsamos sobre el texto "Mujer" se activará la opción del radio button "mujer". Podemos ver como quedaría ese código en marcha a continuación, aunque es una imagen, no podrás hacerlo funcionar:

Hombre



Mujer



BUTTON

A partir de la implementación de los estándares HTML 4.0 contamos con varias etiquetas nuevas para construir formularios, siendo **BUTTON** una de ellas, bastante útil por cierto. **BUTTON** se encuentra ampliamente soportada en la actualidad.

Esta etiqueta proporciona un método único para la implementación de cualquier tipo de botón de formulario. Sus principales atributos son:

- `type="tipo"`, que puede tomar los ya conocidos valores `submit` (por defecto), `reset` y `button`.
- `name="nombre"`, que asigna un nombre identificador único al botón.
- `value="texto"`, que define el texto que va a aparecer en el botón.

La principal ventaja que aporta estas etiquetas es que ahora vamos a poder introducir dentro de ellas cualquier elemento de HTML, como imágenes y tablas.

Podemos ver un ejemplo a continuación.

```
<form action="#" method="post" enctype="text/plain" name="miform">

<button name="boton_1" type="button">
<table width="10" cellspacing="0" cellpadding="2" border="1">
<tr>
<td>uno</td>
<td>dos</td>
</tr>
<tr>
<td>tres</td>
<td>cuatro</td>
</tr>
</table>
</button>
</form>
```

Referencia: Si deseas aprender algo más sobre la etiqueta BUTTON te recomendamos leer el artículo [Botones HTML con la etiqueta BUTTON](#).

Este artículo es obra de *Christian Santalucía*

Fue publicado por primera vez en 03/02/2003

Disponible online en <http://desarrolloweb.com/articulos/21.php>

Etiqueta META robots

Explicación de la etiqueta META robots y diferentes posibilidades de configuración.

Mediante las diferentes etiquetas META que podemos colocar en un sitio web disponemos de una variedad amplia de metainformaciones para comunicar a cualquier sistema que lea nuestra página web. En este artículo vamos a presentar una etiqueta interesante para definir cómo se tienen que comportar los motores de búsqueda a la hora de visitar nuestra página y mostrarla entre los resultados de búsquedas realizados en el buscador. Se trata la etiqueta meta de robots.

La etiqueta META de Robots sirve para personalizar el comportamiento de robots de indexación, tipo Google, a la hora de procesar nuestra página web. Cada una de las páginas de nuestro sitio puede tener una declaración de la etiqueta meta de robots distinta, con lo que podemos incluso definir de manera independiente cómo deseamos que se trate cada una de las páginas que componen el web.

En DesarrolloWeb.com hemos publicado anteriormente informaciones acerca de [distintas etiquetas META](#) en artículos dispersos. Además, tenemos un Generador de Etiquetas Meta que también puede resultar de interés para los lectores.

Etiqueta ROBOTS de META Tags

Como hemos dicho, la etiqueta robots, dentro de las posibles etiquetas con Metainformaciones acerca de un documento web, sirve para llevar un control exhaustivo de lo que puede o no puede hacer un robot de indexación cuando visita nuestro sitio web. Los comportamientos más típicos que podemos definir son permitir o no indexar una página y seguir o no sus enlaces.

Nota: Conviene recordar que también se puede definir el comportamiento de los robots de búsqueda con nuestro sitio, a la hora por ejemplo de permitir o no indexar las distintas páginas, mediante el [archivo robots.txt](#).

Ahora veamos cómo se define esta etiqueta META de robots.

```
<META name="robots" content="NOINDEX">
```

Como se puede ver, se define el etiqueta META y se acompaña de dos atributos esenciales:

Name: que para la etiqueta META que controla los comportamientos en motores de indexación el valor es "robots".

Content: se indica las directivas que queremos que apliquen los motores de indexación cuando visitan la página.

Valores posibles de la etiqueta META ROBOTS

En el atributo Content de la etiqueta meta debemos colocar las directrices que deseemos para buscadores, tantas como deseemos, separadas por comas. Las distintas directrices a aplicar son las siguientes:

INDEX / NOINDEX Sirve para indicar si se desea o no permitir la indexación de la página por los motores de búsqueda.

FOLLOW / NOFOLLOW Con esta directriz se indica si se debe o no permitir a los motores de búsqueda recorrer o seguir recorriendo la web a través de los enlaces que encuentre en el cuerpo del documento.

ARCHIVE / NOARCHIVE Esto permite decir si deseamos o no que el motor de búsqueda archive el contenido del sitio web en su caché interna. Como habremos podido ver, buscadores como Google tienen una caché y podemos ver las páginas web tal como las tiene cacheadas el buscador. Para ello, en los resultados de las búsquedas aparece un enlace que pone caché. Si decimos que no archive la página, no debería mostrar ese enlace de caché. Esto en realidad, según Google, no evita que se guarde en caché la página, sino que no permite verla a los usuarios del buscador y por lo tanto no muestra el enlace.

SNIPPET / NOSNIPPET Esta directriz en principio no resulta muy útil, al menos a primera vista. Sirve para que el motor de búsqueda no muestre ninguna descripción de un sitio, sólo su título. Si utilizas NOSNIPPET automáticamente defines un NOARCHIVE, por lo que la página tampoco se mostrará en caché.

ODP / NOODP Sirve para decirle al buscador que debe, o no, mostrar el título y descripción de la página iguales a los que se encuentra en el Open Directory Project. En algunos casos, algunos buscadores muestran como título y descripción de una web los que se han publicado en el ODP (ENLACE A <http://www.dmoz.org/>).

YDIR / NOYDIR Es básicamente lo mismo que ODP / NOODP, con la diferencia que es para que no se pueda, o si, mostrar la descripción y título que aparece en el directorio de Yahoo.

Cuando no existe esta etiqueta los buscadores interpretan las condiciones más favorables para ellos, es decir, que pueden hacer todo lo que suelen hacer con otras páginas a la nuestra, como indexarla, seguir sus enlaces, archivarla, etc.

Ejemplos de etiquetas META ROBOTS

A la hora de utilizar la META ROBOTS básicamente lo que podemos hacer es restringir las posibilidades de los motores de búsqueda, puesto que las posibilidades por defecto son las menos restrictivas.

Esto quiere decir que una etiqueta como la siguiente es irrelevante, porque el buscador siempre va a indexar la página y seguir sus enlaces de manera predeterminada:

```
<META name="robots" content="INDEX,FOLLOW">
```

Podemos definir entonces casos más restrictivos como estos:

```
<META name="robots" content="INDEX,NOFOLLOW">
```

Para indicar que se desea que se indexe la página, pero no se sigan los enlaces. Dada que la opción INDEX es la que se sobreentiende por defecto, esta etiqueta tendría el mismo valor que la siguiente:

```
<META name="robots" content="NOFOLLOW">
```

Para indicar que no queremos que se sigan los enlaces de la página.

```
<META name="robots" content="NOINDEX,NOFOLLOW">
```

Para indicar que no queremos que se indexe la página ni se sigan los enlaces que pueda contener.

```
<META name="robots" content="NOARCHIVE">
```

Lo único que indicamos es que no se muestre el enlace para ver la página en la caché del buscador.

```
<META name="robots" content="NOINDEX,NOFOLLOW,NOARCHIVE,NOODP,NOSNIPPET">
```

Con esta restrictiva etiqueta forzamos para que no se indexe la página, no se sigan los enlaces, no se muestre el link de caché, no se muestre el título y descripción del Open Directory Project y sólo se muestre el título de la página en los resultados de las búsquedas.

Este artículo es obra de *Miguel Angel Alvarez*
Fue publicado por primera vez en 19/08/2008
Disponible online en <http://desarrolloweb.com/articulos/etiqueta-meta-robots.html>

El futuro del desarrollo web: HTML 5

HTML 5 es una tecnología creada para modernizar la web y el desarrollo de aplicaciones web, online y offline, que aun tiene bastante camino por recorrer para ser una realidad.

HTML 5 es el más nuevo estándar del lenguaje HTML en estos momentos. Durante muchos años estuvo en fase de borrador pero ya desde hace tiempo es una realidad. Para las personas que han estudiado el [Manual de HTML de DesarrolloWeb](#), básicamente indica que ahora disponemos de nuevas etiquetas en el

lenguaje, así como se han marcado como obsoletas varias otras. No obstante, con lo que sabes hasta este punto ya tienes la base necesaria para poder entender de una manera rápida cualquier uso del lenguaje de marcación.

El Manual de HTML está terminando por aquí, pero no queremos perder la oportunidad de hablar de HTML5 y motivar a las personas para que continúen el estudio de HTML acercándose a esta nueva versión. Este artículo de Desarrolloweb.com pretende ofrecer algunas pistas sobre el lenguaje de marcación y HTML5 en general, así como resumir el camino que ha realizado para convertirse en una realidad. Te adelantaremos algunas de las novedades más importantes que nos ofrece.

Por qué se crea HTML 5

Los que nos dedicamos a hacer páginas web sabemos que un sitio web es como un puzzle de tecnologías que operan entre sí. Para hacer una página, en principio, se necesita simplemente HTML, pero esta afirmación hoy tiene poco que ver con la realidad y las necesidades de los desarrolladores para crear una rica experiencia de usuario.

Es que hoy pocos sitios web se basan únicamente en HTML. Quien más quien menos utiliza CSS para definir el aspecto de la página, algún tipo de script del lado del cliente, en Javascript, vídeos en diversos formatos, etc. Para poder beneficiarse e integrar todas estas tecnologías, existen multitud de etiquetas que se han ido creando al paso, según se iban necesitando, y las cuales no han pasado por el filtro de los normalizadores de tecnologías como el W3C.

Por tanto, más de 10 años después que se publicase la última especificación del HTML, resulta primordial para el futuro de la web la creación de un nuevo estándar que recoja y solucione de alguna manera, las necesidades de los desarrolladores que se han ido creando a lo largo de todo este tiempo.

Esfuerzos en diversas vertientes para la creación de HTML 5

Sin duda ya hace tiempo que se necesitaba esta nueva especificación, así que hoy nos preguntamos ¿por qué ha pasado tanto tiempo sin publicarse esta nueva versión del lenguaje? La respuesta es que HTML 5 se ha convertido en un proyecto muy ambicioso, donde hay muchas personas, empresas e instituciones que tienen mucho que opinar. Es de vital importancia, por tanto, coordinar a todos los implicados para crear un único frente común, que asegure esta vez un éxito de la tecnología como un estándar.

En principio, los encargados de regular los estándares de Internet son los integrantes del W3C, que han estado trabajando durante bastante tiempo en otros lenguajes como XML. No se puede decir que hayan dejado de lado HTML, pero de alguna manera estaban creando otros estándares más rígidos que sustituyesen al lenguaje. Ante esta falta de interés en HTML y las necesidades reales de los desarrolladores de webs por parte del W3C, se creó en 2004 una comunidad de personas interesadas en mejorar y modernizar el lenguaje de marcación. Este nuevo grupo, llamado **WHATWG** (Web Hypertext Application Technology Working Group), se creó a raíz de una conferencia del W3C con personas integrantes de los equipos de desarrollo de Apple, la Fundación Mozilla y Opera, a la que se han ido agregando personal de Microsoft y otras empresas implicadas en el mundo web.

El WHATWG, que funciona de manera independiente del W3C, tiene como objetivo principal trabajar en la nueva especificación del HTML 5 y a ellos debemos muchas de los avances que están por llegar con relación al lenguaje. Es una organización abierta, donde cualquiera puede participar libre gratuitamente. De

hecho, según comentan en su web, están realmente interesados en las opiniones e intereses de las personas que trabajan con el desarrollo web, para crear unas especificaciones que respondan a las necesidades reales de los profesionales de Internet.

Qué es verdaderamente HTML5

Cuando nos referimos a HTML5 en principio podríamos pensar que es solo una nueva versión del HTML, pero realmente con este término también se engloba a otras tecnologías que están estrechamente relacionadas con la plataforma web. Es el caso de CSS y las API de Javascript que dependen del navegador.

Por lo que respecta al lenguaje de marcación, HTML en sí, se ha procurado eliminar todo aquello que servía exclusivamente para aplicar estilo, e introducir toda una serie de etiquetas nuevas que aportan valor semántico al contenido. Si te interesa deberías leer el manual sobre las [novedades de HTML5 como lenguaje de marcas](#).

Por lo que respecta a CSS, se ha presentado la versión 3 del lenguaje, que puedes aprender en el [Manual de CSS3](#). Esta versión incluye muchas maneras nuevas de aplicar estilos, que vienen a resolver las diversas demandas de los diseñadores y a evitar que tengamos que hacer trucos diversos para ir un poco más allá de lo que te permitía anteriormente CSS.

Y ya en la parcela de Javascript, HTML5 no entra en el lenguaje en sí (que es regulado por otra organización externa al W3C y compañía), sino en todo lo que ofrecen los navegadores para poder trabajar con Javascript. Nos referimos a una serie de APIs para trabajar con la plataforma web, osea, lo que la web te ofrece. Todos esos conjuntos de funciones sirven para cosas tan variadas como geolocalización, almacenamiento local, trabajo a pantalla completa y un innumerable set de funcionalidades.

Enlaces relacionados

Para la documentación de los lectores, publicamos los enlaces a los borradores de especificaciones del HTML 5 y las organizaciones que trabajan en ellos:

W3C HTML Working Group: <http://www.w3.org/html/>

WHATWG <http://www.whatwg.org>

Última versión publicada del borrador de HTML 5 <http://www.w3.org/TR/html5/>

Borrador del HTML 5 por la WHATWG <http://www.whatwg.org/specs/web-apps/current-work/multipage/>

Conclusión

Hemos podido comprobar que existen numerosos esfuerzos para la creación de las nuevas especificaciones del HTML 5, llevados a cabo por distintas organizaciones, independientes, pero que trabajan en un frente común.

En DesarrolloWeb.com encontrarás cantidad de artículos que tratan sobre cosas concretas del HTML5. Usa el buscador interno para localizarlos. No obstante, queremos darte una referencia para seguir conociendo lo básico: [qué es HTML 5](#).

Este artículo es obra de *Miguel Angel Alvarez*
Fue publicado por primera vez en 14/10/2009
Disponible online en <http://desarrolloweb.com/articulos/html5-futuro-desarrollo.html>

Optimización de la página web

Aunque no es algo que tiene que ver directamente con el lenguaje HTML, es importante que comencemos a poner el foco en la optimización. Explicamos cómo se puede optimizar una página o un sitio web, para que tarde menos en cargar.

Introducción a la optimización de webs

Qué es la optimización de páginas web, cuáles son los principales recursos en una página web susceptibles de recibir mayor optimización y en qué parámetros podemos optimizarlos.

En el mundo de la web cada segundo de carga importa. Incluso cada milisegundo podríamos decir. La explicación es bien sencilla, pues se dice que un usuario estará esperando la carga de un sitio web por 5 segundos máximo, si pasado ese tiempo no se obtiene respuesta alguna del servidor, simplemente abandonará el intento de acceso a la página.

Este detalle se aprecia mejor todavía cuando hablamos de cifras. Por ejemplo según Google, medio segundo de retardo de carga de una página supone un decremento de 20% de su tráfico. Según Amazon, 100 milisegundos de retraso de carga de su web, afecta en 1% la caída de ventas en sus productos. Sin duda datos reveladores. ¿no?

Por este motivo, la optimización de una web es un detalle que no debemos pasar por alto. En este artículo aprenderás los criterios más importantes para conseguir optimizar los archivos y el código de tu sitio web.



Qué recursos son capaces de ofrecer mayor optimización

La optimización de una página depende de varios factores. Obviamente uno de ellos es el servidor donde la tengas publicada. Un servidor rápido te ofrecerá menores tiempos de entrega de tu contenido, pero en este artículo nos vamos a centrar más en la parte que afecta al código de la página.

Nota: no entramos en la configuración del servidor, pues en este momento de nuestro aprendizaje y en el marco del Manual de HTML, la configuración del servidor no forma parte de nuestras

responsabilidades como desarrolladores. Generalmente usaremos servidores de alojamiento compartido, por lo que no tendremos muchas opciones de configurar el servidor.

Entonces, en lo que respecta al código de nuestros archivos en una página web, la optimización la podremos realizar en estos tres principales recursos:

- Imágenes
- Código Javascript
- Código CSS

Además la importancia de la optimización de un sitio también sigue por lo general este orden, siendo las imágenes el recurso más susceptible de alcanzar mejores resultados con una buena optimización.

Vamos a comenzar con la optimización de los conceptos más sencillos de explicar. Luego veremos algunas herramientas que podemos usar para conseguir todas estas optimizaciones.

Optimizar el CSS

El código CSS sirve para definir la capa de presentación y aunque no lo hemos tocado prácticamente nada en este Manual de HTML, es difícil que hagamos cualquier página web sin incluir algo de CSS.

La optimización de nuestro CSS podrá basarse en estos factores principales:

- **Eliminando comentarios:** Si tienes comentarios en el código CSS, deberías eliminarlos de los archivos que se van a distribuir en el código definitivo. Aportan peso innecesario.
- **Compactando el código:** el código compactado ocupa menos espacio, porque se le quitan saltos de línea innecesarios, espacios en blanco, etc.
- **Reduciendo conexiones al servidor:** si tienes varios archivos CSS en tu sitio web, generalmente optimizarás el tiempo de carga si unes todo el código en un mismo fichero.

Nota: Las conexiones al servidor, para descargar todos los archivos externos al propio código HTML, como imágenes, CSS, Javascript, llevan un tiempo. El navegador no es capaz de descargar todo a la vez y si tenemos muchos archivos distintos generalmente tardará más. Esto con el próximo HTTP2 cambiará, puesto que ese nuevo protocolo ha optimizado la posibilidad de realizar más descargas concurrentes sin penalizar el tiempo usado en ellas.

Nota: Como segunda nota, también debes de mantener un buen nivel de calidad de tu código. Por ejemplo, debes intentar reutilizar las declaraciones de estilos, no repetir las reglas a no ser que sea necesario, usar selectores adecuados y en general dejar las cosas sencillas al navegador. Pero esta parte tampoco la vamos a tocar, siendo material de estudio en artículos de [Arquitectura de CSS](#). Te recomendamos la [clase en vivo de arquitectura de CSS](#).

El código CSS compactado luce más o menos como la siguiente imagen:

Otra cosa que se puede hacer con el CSS es dividir los estilos en dos archivos separados. En uno se colocaría el código fundamental para el CSS de cabecera y layout, necesario para la renderización de la parte de la página que se ve al abrirla en el navegador. Luego, se puede usar un segundo archivo CSS en el que se colocan todos los estilos que afectan a los elementos que habrá visibles al hacer scroll hacia abajo. El primer archivo CSS se colocará en el HEAD, para que esté disponible lo más rápido posible y el segundo archivo CSS se enlazará antes de cerrar el BODY, ya que no es necesario tan rápidamente y se puede deferir la carga hasta más tarde.

El código Javascript tiene muchas posibilidades de optimización. Algunas son similares a las comentadas para el código CSS y otras son específicas.

- Esta próxima imagen es una pequeña parte del código de jQuery, una popular librería Javascript, una vez minimizada.

```
if (document.readyState === "complete") {  
    documentElement; return !t&&"HTML"!==t.nodeName}, p=oe.setDocument=function  
(e){var t,i,a=e?e.ownerDocument||e:w; return a!=d&&9==a.nodeType&&  
a.documentElement?(d=a,h=d.documentElement,g=!o(d),w!=d&&(i=d.defaultView)&  
&i.top==i&&(i.addEventListener?i.addEventListener("unload",re,!1)  
:i.attachEvent&&i.attachEvent("onunload",re)),n.attributes=ue(function(e)
```

Defer vs Async en la etiqueta SCRIPT

Los atributos "defer" o "async" sirven para decirle al navegador cuándo procesar el código Javascript. Es solamente posible usarlo en etiquetas SCRIPT que enlazan con archivos js externos.

- **Defer:** indica que el script Javascript será ejecutado cuando la página ha terminado de montarse.
- **Async:** indica que el script Javascript será ejecutado al mismo tiempo que la página se está montando.

Si no tenemos ni defer ni async, entonces ocurre que la ejecución del Javascript se realiza inmediatamente, y una vez finaliza tal ejecución continúa el proceso de montaje de la página y su renderización en la ventana del navegador.

Nota: Obviamente, un código Javascript optimizado también será aquel que tenga menor complejidad algorítmica y que use más capacidades nativas del navegador y menos librerías externas, pero esta parte queda fuera de nuestro alcance en este momento de nuestro aprendizaje.

Optimización de las imágenes

Por fin llevamos a las imágenes, que son realmente importantes porque tenemos mucho margen para optimizar y realmente es bastante sencillo de hacer en el momento en el que estamos. Los principales argumentos para la optimización son los siguientes:

- **Quitar las imágenes:** Parece un poco absurdo este punto, pero resulta obvio que la mejor optimización de una imagen sería hacer que desaparezca de nuestra web. Muchas veces no será posible, pero en realidad y de manera tradicional las páginas web usaban muchas imágenes que realmente hoy no serían necesarias. Por ejemplo imágenes para conseguir un degradado, que se puede definir mediante CSS, imágenes para conseguir una transparencia con canal Alpha (como la del PNG de 24 bit) para una caja, que también se puede conseguir con CSS. O imágenes que se usan para que se pueda representar una fuente específica y no disponible en los ordenadores del usuario, que hoy se puede conseguir con la carga de fuentes web. Otras imágenes son las de los iconos, que también se pueden conseguir con SVG o iconos basados en fuentes web.
- **Usar imágenes vectoriales en vez de mapas de bits:** esto es muy interesante porque hoy la compatibilidad de las imágenes vectoriales en SVG es prácticamente total. (A partir de IE9 está disponible). Los archivos vectoriales tienen la característica que se pueden ampliar y reducir sin perder calidad. Sin embargo, los mapas de bits si se quieren hacer más grandes ocuparán sensiblemente mayor espacio en bytes. Esto es especialmente importante para los dispositivos que tienen mayores densidades de píxeles, pues los archivos vectoriales permiten aprovechar esa mayor densidad de puntos, quedando los diseños más nítidos, sin que eso afecte al peso de los archivos.
- **Escoger el formato idóneo para los mapas de bits:** ya sabes que existen varios formatos gráficos

para la web <https://desarrolloweb.com/articulos/19.php> porque hablamos de ello anteriormente en este manual. Se trata de escoger el que mejor venga en cada caso y el que nos permita ajustar mejor la calidad de las imágenes, siempre con compromiso con su peso.

- **Usar imágenes "responsive":** <https://desarrolloweb.com/articulos/imagenes-responsive-etiqueta-picture.html> de modo que podamos proporcionar a los navegadores diferentes alternativas de imágenes, para que usen la que más les interese en cada caso.

Herramientas para la optimización web

Ahora te preguntará ¿cómo consigues todas estas optimizaciones?. Bueno, la verdad es que aquí las posibilidades son enormes, ya que existen multitud de programas, herramientas, automatizadores de tareas, etc.

Sin entrar en explicar cada una de las herramientas, pues sería extraordinariamente extenso para este artículo, podría clasificarlas en los siguientes apartados.

Programa de diseño

Básicamente, para la parte de las imágenes podrás usar cualquier programa de diseño que tengas. Photoshop, Gimp, Affinity Photo, Sketch, Pixlr, etc. Todos tienen herramientas para que, al guardar los archivos, se pueda ajustar su calidad.

Puedes aprender más sobre este tema por ejemplo en el [Manual de Photoshop](#). Con [Gimp puedes aprender a optimizar imágenes con este vídeo](#).

Programas de automatización de tareas con interfaz gráfica

Existen muchos programas que permiten automatización de tareas de optimización web, con interfaz gráfica y relativamente sencillos de usar.

Estos programas incluyen asuntos como el minimizado del código Javascript y CSS, así como la optimización de las imágenes usadas en un proyecto.

Alternativas conocidas serían Prepros, CodeKit, Koala, Compass. Koala es la única que es de código abierto y gratuita.

Automatización de tareas profesional

Los profesionales del desarrollo frontend más puristas prefieren las herramientas como Gulp o Grunt, o incluso scripts npm, para realizar este tipo de tareas de optimización.

Estas herramientas son lanzadores de tareas, que permiten automatizar todos los procesos de optimización de un sitio web, adaptando al 100% las características de cada proyecto. Tienen la ventaja de ser gratuitas y permitir un sin fin de posibilidades a los desarrolladores, así como la desventaja de requerir algo más de estudio para dominarlas.

Este tipo de herramientas las puedes estudiar por ejemplo en el [Curso de automatización y optimización](#)

[frontend de EscuelaIT](#).

WPO

Ya para acabar queremos mencionar que existe toda una disciplina en torno de la optimización web llamada WPO. Es una disciplina importante y prueba de ello es que hay profesionales que solamente se dedican al WPO.

WPO son las siglas de "Web Performante Optimization", optimización del rendimiento de la web e incluye tanto la parte del sitio web como la optimización del servidor. Es algo de lo que ya hemos hablado en varios momentos en DesarrolloWeb.com, así que te recomendamos asistir a esta clase, en la que se habla de WPO y que es la primera de las clases del [Taller de WPO de EscuelaIT](#).

Para ver este vídeo es necesario visitar el artículo original en:
<http://desarrolloweb.com/articulos/introduccion-optimizacion-webs.html>

Este artículo es obra de *Miguel Angel Alvarez*

Fue publicado por primera vez en 17/04/2018

Disponible online en <http://desarrolloweb.com/articulos/introduccion-optimizacion-webs.html>