



# Manual de HTML 4.01

Breve referencia para la asignatura de Lenguajes de  
Marcado y Sistemas de Información del Ciclo Formativo de  
grado Superior de Desarrollo de Aplicaciones  
Multiplataforma

## Contenido

<b>Manual de HTML 4.01 .....</b>	<b>1</b>
<b>1. Prólogo.....</b>	<b>4</b>
<b>2. Introducción a HTML.....</b>	<b>6</b>
<b>3. Sintaxis de HTML .....</b>	<b>8</b>
3.1 Anatomía de una etiqueta HTML .....	8
3.2 Partes de un documento HTML .....	9
3.3 Saltos de línea en HTML .....	10
3.4 Doctype .....	11
3.5 Juego de caracteres.....	11
<b>4. Elementos de texto .....</b>	<b>15</b>
4.1 Párrafos .....	15
4.2 Encabezados.....	19
<b>5. Formateando el texto .....</b>	<b>22</b>
5.1 Negrita.....	22
5.2 Itálica .....	23
5.3 Subrayado.....	23
5.4 Subíndices y superíndices.....	23
5.5 Anidar etiquetas.....	24
<b>6. Los colores y HTML .....</b>	<b>25</b>
6.1 Atributos de color en etiquetas HTML .....	26
6.2 Combinar otros colores.....	26
6.3 Colores seguros .....	27
<b>7. Atributos para páginas .....</b>	<b>29</b>
7.1 Atributos para fondos .....	29
7.2 Color del texto.....	30
7.3 Márgenes.....	31
7.4 Por qué todos estos estilos deberían definirse en CSS .....	32
<b>8. Listas en HTML.....</b>	<b>33</b>
8.1 Listas desordenadas .....	33
8.2 Listas ordenadas.....	34
8.3 Listas de definición .....	36
8.4 Anidando listas .....	37
<b>9. Enlaces en HTML.....</b>	<b>38</b>
9.1 Sintaxis de un enlace.....	38
9.2 El aspecto de los enlaces.....	39

9.3 Tipos de enlaces .....	39
9.4 Enlaces internos .....	40
9.5 Enlaces locales.....	41
9.6 Enlaces externos.....	43
9.7 Enlaces a direcciones de correo .....	44
9.8 Enlaces con archivos.....	45
<b>10.- Imágenes en HTML.....</b>	<b>47</b>
10.1 Atributos de las imágenes .....	48
10.2 Tipos de archivos.....	51
<b>11. Tablas en HTML.....</b>	<b>52</b>
11.1 Estructura y elementos de las tablas .....	52
11.2 Atributos de tablas, filas y celdas.....	53
11.3 Tablas anidadas .....	57
11.4 Maquetación con tablas .....	60
<b>12. Formularios .....</b>	<b>65</b>
12.1 Qué se puede hacer con un formulario .....	65
12.2 Cómo hacer un formulario en HTML.....	66
12.3 Elementos de formulario.....	67
12.4 Envío y borrado de formularios.....	74
12.5 Ejemplo completo de formulario .....	76

# 1. Prólogo

El presente manual de HTML se basa en gran medida en el trabajo desarrollado por Rubén Álvarez para desarrolloweb.com y ha sido adaptado para conocer los elementos más importantes de HTML 4.01. Esto implica que algunos elementos del lenguaje han quedado fuera y que otros, que en la actualidad se usan con poca frecuencia o que incluso no son ya recomendados gracias a CSS se expliquen con cierta profundidad. En cualquier caso este manual pretende enseñar a trabajar con el lenguaje de marcado que sirve para construir páginas web a nivel introductorio y con la idea en mente de pasar más adelante a otras tecnologías con el mencionado CSS.

HTML es el primer paso que debería completar cualquier persona que quiera dedicarse al desarrollo web en general y es 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.

En este manual encontraremos información acerca de los siguientes aspectos del lenguaje:

## **1.- Prólogo**

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.

## **2.- Introducción a HTML**

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

## **3.- 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.

## **4.- Formato de párrafo en HTML**

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

## **5.- Formateando el texto**

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

## **6.- 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.

## **7.- 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.

## **8.- Listas en HTML**

Vemos lo que son las listas y señalamos los tres tipos existentes.

## **9.- Enlaces en HTML**

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

## **10.- Imágenes en HTML**

Vemos cómo colocar una imagen en una página web y algunos atributos básicos para asignarle estilos a las imágenes en HTML. También abordamos aquí los distintos formatos gráficos utilizados en las páginas web, GIF, JPG y PNG.

## **11.- Tablas en HTML**

Vemos lo que son las tablas, para qué sirven y en qué casos podemos utilizarlas. Vemos la tabla más simple posible. También conocemos los atributos principales que podemos asignar a las tablas y cómo agrupar filas de una tabla, o columnas con el fin de diseñar estructuras más complejas e irregulares.

## **12.- Formularios HTML**

En este tema se estudian los elementos relacionados con la creación y manejo de formularios.

## 2. Introducción a HTML

A través de los próximos 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.

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.

### *Qué aprenderás en este manual*

Si le pones un poco de ganas y sigues este manual hasta el final, obtendrás 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.

### *Revisión de 2017*

Estamos revisando el manual en 2017. 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 a día de hoy tengan una información fiel a las costumbres y buenas prácticas a la hora de usar este lenguaje. En esta revisión estamos se han eliminado elementos cuya utilidad se ha demostrado en desuso y se ha detallado otros que, por su utilidad práctica o didáctica, han demostrado un valor que merece la pena destacar. Esperamos que este esfuerzo sea de provecho todavía para muchas personas a lo largo del mundo.

## 3. Sintaxis de 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 persona, puesto que el lenguaje es muy entendible por los seres humanos.

### 3.1 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:** en este manual, aunque en ocasiones hagamos referencia a determinadas etiquetas en mayúsculas, lo recomendable al programar es escribirlas siempre en minúsculas y acompañados de sus símbolos de mayor y menor.

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 está en negrita</b>
```

Veremos que las palabras "Esto está 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

## 3.2 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 capítulo, dedicado a la página HTML básica.

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.

### 3.3 Saltos 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.

### Primera página web

Vamos a desarrollar a continuación un sencillo ejemplo con los elementos mencionados en este capítulo. 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 apartado anterior 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.

### 3.4 Doctype

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 de alguna manera precursor del HTML. Su funcionalidad principal es la de definir “oficialmente” el tipo de documento que estamos generando, no solo de cara a los navegadores sino también de cara a buscadores y otras herramientas web que día a día exploran y valoran nuestro trabajo e incluso para que nuestros usuarios vean que nuestro trabajo es riguroso y de calidad. Para HTML existen algunas variantes dependiendo de a qué versión nos enfrentemos o cómo deseamos que sea nuestro código validado en caso de que tenga que ser evaluado. Una de las declaraciones Doctype más extendidas en HTML 4.01 es la transicional y es como sigue:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

Esta versión en concreto resulta razonablemente tolerante con posibles errores, lo cual nos conviene a la hora de comenzar con este lenguaje.

Si lo que deseamos es desarrollar nuestro código en formato XHTML, que es prácticamente igual al de HTML 4.01 exceptuando el uso de ciertas etiquetas de estilo que son abolidas, el doctype que debe ser empleado sería el siguiente:

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

Con la llegada de HTML5 la declaración se simplificó para quedar simplemente como esto:

```
<!DOCTYPE html>
```

Aunque existen más variantes de la declaración doctype, queda fuera de este apartado entrar a discutir las y explorarlas, por lo que, en adelante nos centraremos en el uso de las mencionadas únicamente.

### 3.5 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.

Ejemplo para HTML 4.01:

```
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
```

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 uno de los motivos que nos inclina a recomendar el uso **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">
```

### *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 carácter como el guión "-" o el guión bajo "\_" te puede ayudar a separar las palabras. Por ejemplo quienes\_somos.html*

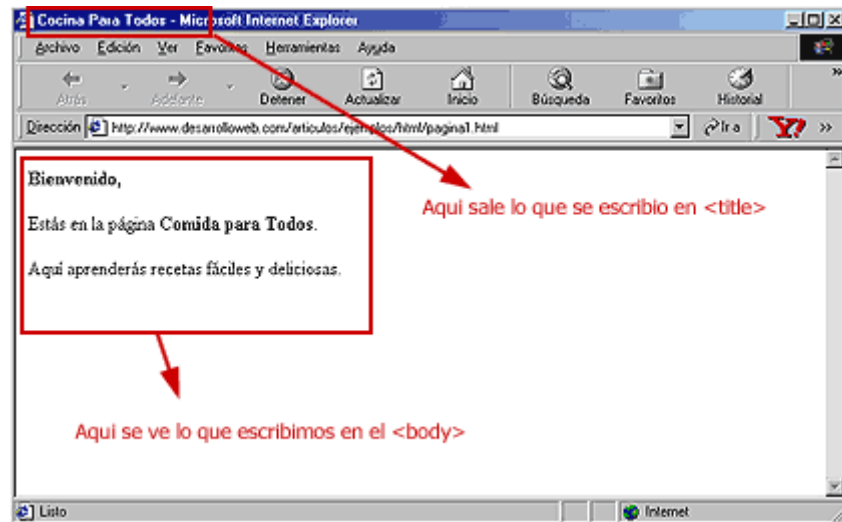
Con el documento HTML creado, podemos ver el resultado obtenido a partir de un navegador. Es conveniente, llegado a este punto, hacer hincapié en el hecho de que no todos los navegadores son idénticos a la hora de interpretar un documento. Desgraciadamente, los resultados de nuestro código pueden cambiar 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.

## 4. Elementos de texto

*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.

### 4.1 Párrafos

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.

**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.

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 se puede observar 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 sí 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...

## 4.2 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 este formateo 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>
```



## 5. Formateando el texto

*Vemos como colocar negritas, itálicas, subrayados, subíndices y superíndices.*

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 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.

### 5.1 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 las 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.*

## 5.2 Itálica

También en este caso existen dos posibilidades, una corta: **I** y su cierre (de italic) y otra un poco más larga: **EM** y su cierre (de emphasis). En este manual, y en la mayoría de las páginas que veréis por ahí os encontraréis con la primera forma, sin duda más sencilla de escribir y recordar.

He aquí un ejemplo de texto en itálica:

```
<i>Texto en itálica</i>
```

Que da el siguiente efecto:

*Texto en itálica*

## 5.3 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.

## 5.4 Subíndices y superíndices

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

```
<sup> y </sup> para los superíndices
```

```
<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  $^{13}\text{CC}_3\text{H}_4\text{CINOS}$  es un heterociclo alergeno enriquecido

## 5.5 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.



## 6. 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 paleta grande, 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.

## 6.1 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 se puede observar, al atributo **color** le damos un valor RGB en formato hexadecimal. El carácter # 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">
```

## 6.2 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 es echar mano de un programa de diseño gráfico con selectores de color que nos suelen dar los valores RGB para que los podamos usar en cualquier sitio. Algunos editores vienen con "color pickers" integrados para facilitar esta tarea. La mayoría de los editores pueden instalar de manera adicional plugins para implementar selectores de color, ya que es una demanda muy habitual de los desarrolladores.

Naranja	#FF8000
Verde turquesa	#339966
Azul oscuro	#000080

## 6.3 Colores seguros

Debemos estar preparados para recibir visitas desde todo tipo de dispositivos y a todos los 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.

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

## 7. 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 capítulo 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 debería definirse a través de CSS. Sin embargo resulta una buena práctica conocer estos conceptos y saber usarlos adecuadamente en HTML antes de introducirse en CSS.

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.

### 7.1 Atributos para fondos

**bgcolor:** especificamos un color de fondo para la página. El color de fondo que podemos asignar con bgcolor es un color plano, es decir el mismo para toda la superficie del navegador y lo emplearemos de forma similar a como vimos en el capítulo anterior mediante el RGB.

**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 cómo 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.



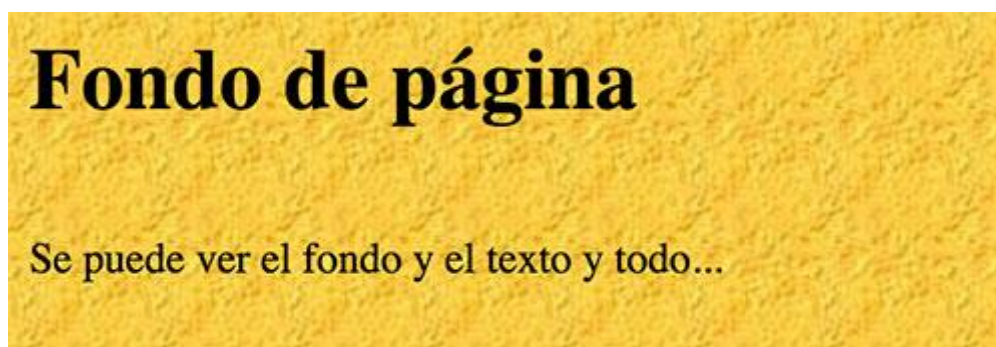
La imagen se llama fondo.jpg. Puedes guardarla en tu disco duro para practicar tú también con ella, mediante un clic con el botón derecho. Para trabajar con esta imagen vamos a colocarla en la misma carpeta donde está el HTML donde vamos a trabajar. Más adelante también hablaremos sobre cómo acceder 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.

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.

## 7.2 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)



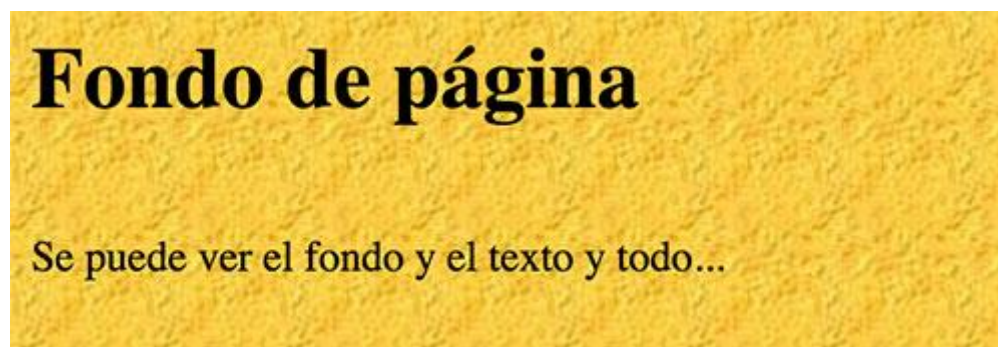
**vlink:** el color de los enlaces visitados. La "v" viene justamente de la palabra "visitado". Es el color que tendrán los enlaces que ya hemos visitado. Por defecto su color es morado. Este color debería ser un poco menos vivo que el color de los enlaces normales.

**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="#ffff33" alink="#ffffcc" vlink="ffff00">
```

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



## 7.3 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)

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>
    <p>Gracias por visitarme!</p>
  </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.

## 7.4 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.



## 8. Listas en HTML

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 cuando apliquemos estilos CSS a las listas.

Aquí 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.

Podemos distinguir tres tipos de listas HTML:

- Listas desordenadas
- Listas ordenadas
- Listas de definición

### 8.1 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`

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

## 8.2 Listas ordenadas

Las listas ordenadas sirven también para presentar información, en diversos elementos o items, con la particularidad que éstos estarán precedidos 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 será 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

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.

## 8.3 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). La 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>Señórula montada en una escóbula
  <dt>Oreja
  <dd>Sesenta minutejos
</dl>
```

El efecto producido:

Diccionario de la Real Academia

**Brujula**  
Señórula montada en una escóbula

**Oreja**  
Sesenta minutejos

El código:

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

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

El resultado:

## Primer nivel de desplazamiento

## Segundo nivel de desplazamiento

## Tercer nivel de desplazamiento

### 8.4 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</li>
  <ol>
    <li>Buenos Aires</li>
    <li>Bariloche</li>
  </ol>
  <li>Uruguay</li>
  <ol>
    <li>Montevideo</li>
    <li>Punta del Este</li>
  </ol>
</ul>
```

#### Ciudades del mundo

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

## 9. Enlaces en HTML

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 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 cómo 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.

### 9.1 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, se presentaría de la siguiente 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>
```

## 9.2 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, a fecha de hoy resulta infrecuente encontrarnos con enlaces que mantengan esa estética, pues su estilo puede ser alterado vía CSS y la mayoría de los sitios web ya contemplan variaciones al respecto.

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 mediante los atributos **alink** y **vlink** que vimos anteriormente.

## 9.3 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 descargar ficheros.

## 9.4 Enlaces internos

Aunque el término "Enlaces internos" puede ser un poco ambiguo, aquí nos referimos a los enlaces que apuntan a un lugar diferente dentro de la misma página. 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, como por ejemplo [Wikipedia](#). 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 son necesarios 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 cómo 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.

## 9.5 Enlaces locales

Como hemos dicho, un sitio web está constituido de páginas interconectadas, 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.

Los llamados "Enlaces locales" son 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, 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 está 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 cómo 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 práctica 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 está 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>
```

## 9.6 Enlaces externos

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 todos 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://**.

## 9.7 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>
```

**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.

## 9.8 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>
```

# 10.- Imágenes en HTML

*En este capítulo veremos cómo colocar imágenes 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.

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 cómo 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. Más adelante estudiaremos dichos formatos.

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.

## 10.1 Atributos de las imágenes

### *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:

```

```

Principalmente 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 discapacidades. Navegadores para invidentes, por ejemplo, no muestran las imágenes y en su lugar los atributos alt son leídos ofreciendo información adicional. 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 para agilizar el proceso de navegación, 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 en la actualidad muchos sitios se diseñan de manera que las imágenes se adapten al tamaño de la pantalla donde se va a visualizar.

En cualquier caso 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 y en "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 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 reservara 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 pixel 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 píxeles se redimensiona para que ocupe solo 400 píxeles), todos los puntos de la imagen servirán para que se vean con mayor nitidez en las pantallas grandes.

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.

### 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 cómo se produce en algunos casos una deformación de las imágenes, debido a por un cambio no proporcional en las dimensiones.

## 10.2 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:

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

# 11. Tablas en HTML

**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.*

En el pasado, especialmente en webs de primera y segunda generación, las tablas resultaban un valioso recurso para maquetar las páginas web, sin embargo con la llegada de CSS esto propició un cambio muy positivo planteando nuevas herramientas para este proceso. 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 más allá de ámbito de estudio.

## 11.1 Estructura y elementos de las tablas

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 filas de arriba a abajo y luego por columnas (o más bien celdas) de izquierda a derecha dentro de esas filas. Cada una de estas filas se define con la etiqueta **TR** y su correspondiente cierre.

Asimismo, cada una de las celdas definidas dentro de estas filas vendrá definida por la etiqueta **TD**, de manera que entre esta etiqueta y su correspondiente cierre incluiremos los contenidos.

### 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 que será más importante al hacer uso de HTML 5.

## 11.2 Atributos de 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 válido 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.

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 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 píxeles o puntos de pantalla) o de una forma relativa, es decir por porcentajes referidos al tamaño total de la tabla.

A modo de ejemplo:

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

Dará una anchura de 80 píxeles 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 aumentara 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 cual 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.

### *Combinar celdas (rowspan y colspan)*

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", por eso tiene fusionada la celda de abajo.	Celda Normal Otra celda normal
--	--

El resto de los atributos presentados presentan una utilidad y uso bastante obvios:

**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 píxeles del borde principal.

**bordercolor:** Define el color del borde.

**cellpadding:** Define, en píxeles, el espacio entre los bordes de la celda y el contenido de la misma.

**cellspacing:** Define el espacio entre los bordes (en píxeles).

**height:** Define la altura de la tabla en píxeles o porcentaje.

**width:** Define la anchura de la tabla en píxeles o porcentaje.

Ojo, 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.

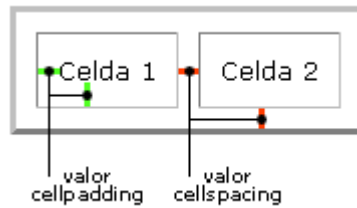
Vamos a poner tres ejemplos de alineado de tablas, centradas, alineadas a la derecha y a la izquierda.

<b>Ejemplo de tabla centrada</b>	Esta tabla está centrada (align="center"). Solo tiene una celda. Este sería un texto cualquiera colocado al lado de una tabla centrada
<b>Ejemplo de tabla alineada a la derecha</b>	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.
<b>Ejemplo de tabla alineada a la izquierda</b>	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 parecernos un poco confuso su uso pero un poco de práctica 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).

## 11.3 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 cellpadding="10" cellspacing="10" border="3">
<tr>
  <td align="center">
    Celda de la tabla principal
  </td>
  <td align="center">
    <table cellpadding="2" cellspacing="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>
</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>
<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:	<table><tr><td>Venezuela</td></tr><tr><td>Colombia</td></tr><tr><td>Ecuador</td></tr><tr><td>Perú</td></tr></table>	Venezuela	Colombia	Ecuador	Perú
Venezuela					
Colombia					
Ecuador					
Perú					
Parte de abajo de América del Sur. Países como:	<table><tr><td>Argentina</td></tr><tr><td>Chile</td></tr><tr><td>Uruguay</td></tr><tr><td>Paraguay</td></tr></table>	Argentina	Chile	Uruguay	Paraguay
Argentina					
Chile					
Uruguay					
Paraguay					
Bosque tropical, clima de sabana, clima marítimo con inviernos secos.	Climas marítimos con veranos secos, con inviernos secos, climas frios, clima de estepa, clima desértico.				

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

```

<table cellpadding="4" cellspacing="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" cellpadding="1" cellspacing="1" border="1">
            <tr>
                <td bgcolor="#cccccc" align="center">Venezuela</td>
            </tr>
            <tr>
                <td bgcolor="#cccccc" align="center">Colombia</td>
            </tr>
            <tr>
                <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" cellpadding="1" cellspacing="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>
        Bosque tropical, clima de sabana, clima marítimo con
        inviernos secos.
    </td>
    <td>
        Climas marítimos con veranos secos, con inviernos secos,
        climas frios, clima de estepa, clima desértico.
    </td>
</tr>
</table>

```

```

        <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 fríos, clima de estepa, clima desértico.</td>
</tr>
</table>

```

## 11.4 Maquetación con tablas

En HTML (antes de la popularización del lenguaje CSS) se utilizaban las tablas para maquetar páginas, aparte de mostrar información tabulada como hemos visto en este artículo. Con 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.

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 y la tendencia a hacer uso de las tablas para esta función ha quedado en desuso.

Sin embargo resulta una buena práctica hacer uso de esta capacidad del pasado para iniciarse en el diseño y el desarrollo web, sobre todo cuando se está empezado con el HTML y no nos hemos iniciado todavía en las bondades de CSS para hacerlo de manera más correcta. Resulta, también, una buena práctica para iniciarse en el procedimiento de maquetación, dado que después, aún con CSS e incluso HTML 5, seguiremos adoptando en cierta medida técnicas similares para la estructuración de contenidos. Por ello incluimos en este capítulo un pequeño apartado en referencia a esta metodología, donde además se presentan modelos de maquetación clásicos en diseños de web sencillas.

### Ejemplo de maquetación web sencilla

<b>LOGO</b>	<b>ENCABEZADO</b>
Menú (lo habitual es ubicar aquí una tabla de una fila y varias celdas para construir el menú de opciones)	
Contenidos (aquí se puede insertar una tabla de estructura mas compleja para dividir los contenidos)	
Pié de pagina (puede tener otra tabla con opciones)	

La imagen anterior representa un ejemplo de maquetación sencilla recreado con tablas. Para que resulte más visual y evidente se han mantenido los bordes, cosa que en un diseño real evitaríamos pues no ayuda a mejorar la estética. Su código fuente sería como sigue:

```
<table bgcolor="#EEEEEE" border ="1" width="80%" align="center">
  <tr> <!-- fila 1 - zona header /-->
    <td width="200" height="150">
      <h1 align="center">LOGO</h1>
    </td>
    <td><h1 align="center">ENCABEZADO</h1></td>
  </tr>

  <tr> <!-- fila 2 - zona de menú /-->
    <td colspan="2" height="40">
      <p align="center"> Menú (lo habitual es ubicar aquí una tabla
        de una fila y varias celdas para construir el menú de
        opciones)</p>
    </td>
  </tr>


  <tr> <!-- fila 3 - zona de contenidos /-->
    <td colspan="2" height="300">
      <p align="center"> Contenidos (aquí se puede insertar una
        tabla de estructura más compleja para dividir los contenidos)
      </p>
    </td>
  </tr>

  <tr> <!-- fila 4 - zona pie de página /-->
    <td colspan="2" height="40">
      <p align="center"> Pié de página (puede tener otra tabla con
        opciones) </p>
    </td>
  </tr>
</table>
```

Como se puede apreciar en este código, el proceso se la desarrollado en base a una única tabla con una anchura del 80% y centrada en nuestra página, la cual se distribuye en cuatro filas, una por sección (encabezado, menú, contenidos y pié) permitiendo albergar en su interior una o más celdas. Dado que en el encabezado se ha reservado espacio para el logotipo y el nombre de la web, lo cual ocupa dos celdas, el resto de las filas contendrán celdas con el atributo `colspan="2"` para mantener el equilibrio.

Las filas 2 y 3, orientadas a albergar un menú y contenidos respectivamente, podrían contener en su interior elementos más complejos, incluidos otras tablas para favorecer espacios y divisiones múltiples independientes del esqueleto principal del ejemplo, tal como se presenta en el siguiente ejemplo.

### Ejemplo de maquetación web sencilla (versión extendida)

		<b>ENCABEZADO</b>		
opción 1	opción 2	opción 3	opción 4	opción 5
<b>Título de sección</b> <p>Es un hecho establecido hace demasiado tiempo que un lector se distraerá con el contenido del texto de un sitio mientras que mira su diseño. El punto de usar Lorem Ipsum es que tiene una distribución más o menos normal de las letras, al contrario de usar textos como por ejemplo "Contenido aquí, contenido aquí".</p> <p>Muchos paquetes de autoedición y editores de páginas web usan el Lorem Ipsum como su texto por defecto, y al hacer una búsqueda de "Lorem Ipsum" va a dar por resultado muchos sitios web que usan este texto si se encuentran en estado de desarrollo. Muchas versiones han evolucionado a través de los años, algunas veces por accidente, otras veces a propósito.</p>			<ul style="list-style-type: none"><li>• Opción A</li><li>• Opción B</li><li>• Opción C</li><li>• Opción D</li></ul>	
Pié de pagina (puede tener otra tabla con opciones)				

Aquí se incluyen algunos elementos adicionales sobre el ejemplo anterior, definidos con un color de fondo diferente para distinguirlos. En él, como se sugería en la explicación anterior, se han incluido dos tablas más, una para el menú y otra para los contenidos. El código, que no difiere demasiado del ejemplo anterior, quedaría como sigue:

```
<table bgcolor="#EEEEEE" border="1" width="80%" align="center">

<tr> <!-- fila 1 - zona header /-->
  <td width="150" height="150">
    
  </td>
  <td><h1 align="center">ENCABEZADO</h1></td>
</tr>
```

```

<tr> <!-- fila 2 - zona de menús /-->
  <td colspan="2" height="40">
    <table width="98%" bgcolor="FFEEEE" border="1" align="center">
      <tr align="center">
        <td>opción 1</td>
        <td>opción 2</td>
        <td>opción 3</td>
        <td>opción 4</td>
        <td>opción 5</td>
      </tr>
    </table>

  </td>
</tr>

<tr> <!-- fila 3 - zona de contenidos /-->
  <td colspan="2" height="300">
    <table width="98%" height="300" border="1" bgcolor="EEFFEE">
      <tr>
        <td width="80%">
          <h2>Título de sección</h2>
          <p>Es un hecho establecido hace demasiado tiempo que un
            lector se distraerá con el contenido del texto de un
            sitio mientras que mira su diseño. El punto de usar
            Lorem Ipsum es que tiene una distribución más o menos
            normal de las letras, al contrario de usar textos
            como por ejemplo "Contenido aquí, contenido aquí".
          </p>
          <p>Muchos paquetes de autoedición y editores de páginas
            web usan el Lorem Ipsum como su texto por defecto, y
            al hacer una búsqueda de "Lorem Ipsum" va a dar por
            resultado muchos sitios web que usan este texto si se
            encuentran en estado de desarrollo. Muchas versiones
            han evolucionado a través de los años, algunas veces
            por accidente, otras veces a propósito. </p>
        </td>
        <td>
          <ul>
            <li>Opción A</li>
            <li>Opción B</li>
            <li>Opción C</li>
            <li>Opción D</li>
          </ul>
        </td>
      </tr>
    </table>
  </td>
</tr>

<tr> <!-- fila 4 - zona pie de página /-->
  <td colspan="2" height="40">
    <p align="center"> Pié de página (puede tener otra tabla con
opciones) </p>
  </td>
</tr>

</table>

```

Este será el modelo de diseño que, en buena medida, podremos luego maquetar estructuras similares con herramientas como las capas <div> y el lenguaje CSS, aunque de manera más sencilla y adaptable además de con un menor coste de procesamiento.



## 12. Formularios

Los formularios y sus elementos son elementos imprescindibles para el desarrollo web, dado que ofrecen la posibilidad de intercambiar información con nuestro visitante. Desde luego, este nuevo aspecto resulta primordial para gran cantidad de acciones que se pueden llevar a cabo mediante la 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. Además esa solución, la de los enlaces con "mailto", requiere que el visitante tenga instalado en su ordenador alguna aplicación de correo electrónico, como por ejemplo Outlook o Firebird. 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.

### 12.1 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. Para poder realizar un envío real desde el formulario se necesita de algún tipo de mecanismo ubicado en un servidor web con capacidad para procesar esa información o para envío de correo (acceso a un servidor de correo).

En este caso 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 y utilizar cualquiera de las docenas de soluciones prediseñadas por otros desarrolladores y ubicar el programa en nuestro servidor.

Así pues HTML permite 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.

## 12.2 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 la etiqueta de apertura 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 cuyo caso se deberá especificar el nombre del script junto con su ruta

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="scripts/FormToEmail.php" ...>
```

La forma en la que se expresa la localización del archivo que contiene el programa es la misma que la vista para los enlaces. En este caso se está haciendo a un script de PHP bastante popular llamado FormToEmail.php que, adecuadamente configurado, es capaz de procesar la información de un formulario y enviarla por correo desde un servidor. En el ejemplo este script se encuentra ubicado dentro de la carpeta scripts.

**method:** Este atributo se encarga de especificar la forma en la que el formulario es enviado. Los dos valores posibles que puede tomar este atributo son **post** y **get**.

Estos valores se diferencian en la forma de realizar el envío de la información del formulario:

- Con get el envío se realiza dividido en variables y concatenado a la URL
- Con post el envío resulta invisible al usuario. A efectos prácticos y, salvo que se indique lo contrario, se usará 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.

**Name:** este atributo no es imprescindible pero sí puede resultar de interés si en nuestra página hacemos uso de javascript y necesitamos acceder a los valores de algún elemento del formulario, de manera que podremos referirnos a él por su nombre.

## 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 name="login" action="mailto:direccion@correo.com" method="post"
enctype="text/plain">

    <!-- elementos del formulario -->

</form>
```

Entre esta etiqueta y su cierre colocaremos el resto de etiquetas que darán forma a nuestro formulario, las cuales veremos a continuación.

## 12.3 Elementos de formulario

El lenguaje HTML nos propone una gran diversidad de alternativas a la hora de crear nuestros formularios. 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 presente capítulo 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 cómo 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, como por ejemplo un comentario, 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.

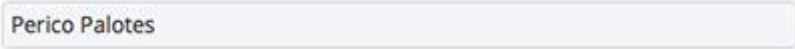
**maxlength:** indica el tamaño máximo del texto, en número de caracteres, que puede ser escrito en el campo. En caso que el campo de texto tenga definido el atributo maxlength, el navegador no permitirá escribir más caracteres en ese campo que los que hayamos indicado.

**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:



Perico Palotes

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 podremos dibujar estos elementos pero no tendrán funcionalidad alguna a menos que lo vinculemos con otras tecnologías como JavaScript. 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:

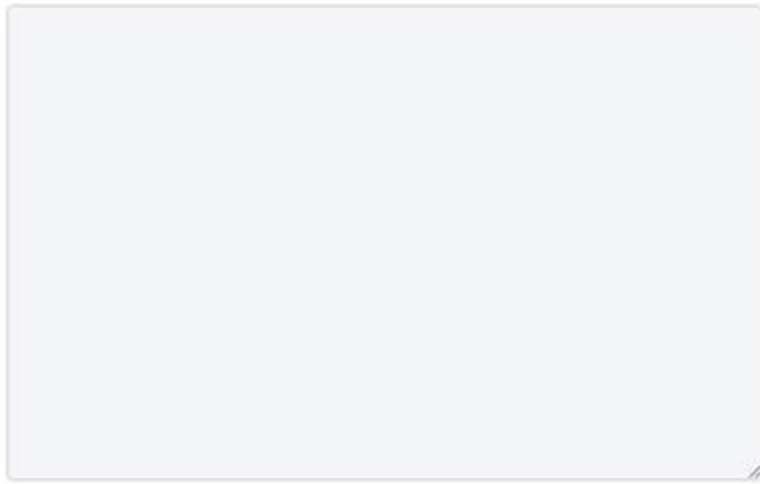
**rows:** define el número de líneas del campo de texto.

**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:



Seguramente hayamos percibido que los textos son una manera muy práctica 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 difícilmente adaptables a programas 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 práctica 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.

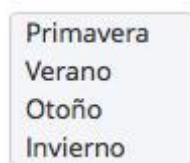
Veamos cual es el efecto producido por estos dos atributos cambiando la línea:

```
<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.

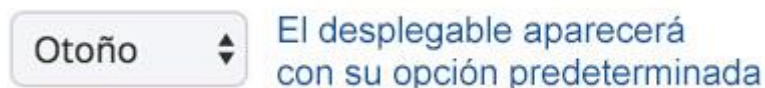
Así, si cambiamos la línea del código anterior:

```
<option>Otoño</option>
```

por:

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

El resultado será:



#### **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 necesitemos.

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).

## 12.4 Envío y borrado de formularios

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 apartado 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:

Enviar

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 rellenos 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 sí pueden ser detectados solicitando el código fuente, no son frecuentemente utilizados por páginas construidas en HTML, son usados sobre todo 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 sí 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:

Texto escrito en el botón

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.

## 12.5 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.

Enviar formulario Borrar todo

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>
```