

Introducción al HTML y al CSS

Marc Llibre Giralt

PID_00168095

Índice

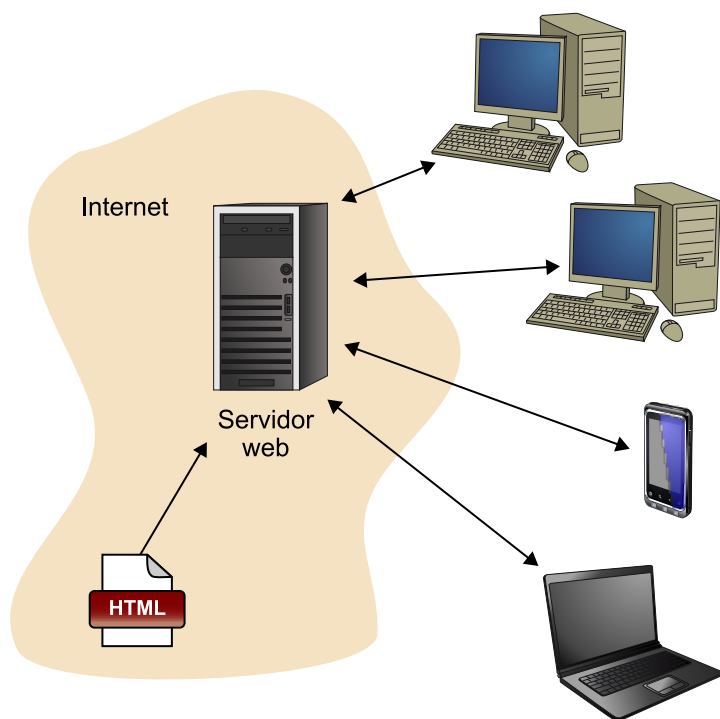
1. Cómo funciona Internet.....	5
2. HTML.....	7
2.1. ¿Qué es HTML?	7
2.2. World Wide Web	8
2.3. Estructura de un documento HTML	9
2.4. Elementos, etiquetas y atributos	12
2.5. Apariencia de una página	16
2.6. Guía de elementos básicos HTML	18
2.7. Caracteres especiales	20
2.8. Enlaces HTML	22
2.9. URL	23
2.9.1. Enlaces relativos y absolutos	24
2.9.2. Otro tipo de enlaces habituales	27
2.10. Listas	28
2.11. Imágenes	29
2.12. Tablas	31
2.13. Maquetación de páginas más complejas	34
3. CSS.....	37
3.1. Selectores	40
3.2. Unidades de medida	45
3.3. Colores	46
3.4. Box model	47
3.4.1. Propiedades anchura y altura	48
3.4.2. Propiedad margin (margen)	48
3.4.3. Propiedad padding (relleno)	51
3.4.4. Propiedad border (bordes)	51
3.4.5. Tamaño final de un elemento	53
3.4.6. Fondos	54
3.5. Tipografía	57
3.6. Texto	60
3.7. Enlaces	62
3.8. Listas	65
3.9. Posicionamiento de los elementos HTML	68
3.9.1. Posicionamiento normal	69
3.9.2. Posicionamiento float	71
3.9.3. Posicionamiento absoluto	74
3.10. Layout	77

1. Cómo funciona Internet

Antes de empezar de lleno en el lenguaje de programación, es necesario tener claro cómo funciona Internet.

Internet está formado por millones de servidores web que almacenan una infinidad de páginas web. Para crear una página web, primero tenemos que crear archivos escritos en HTML en nuestro ordenador o portátil y después ponerlos en un servidor web. A este último paso también se lo denomina subir los archivos al servidor vía FTP, que es el protocolo que sirve para este fin. Veremos más adelante cómo se realiza este paso.

Si ya tenemos nuestras páginas subidas a un servidor, cualquier ordenador con acceso a Internet podrá visualizarlas a través de los navegadores como Internet Explorer, Firefox, Google Chrome, Safari, Opera y otros.



Los códigos HTML y CSS de nuestras páginas web indican a los navegadores cómo se debe mostrar la página. Si lo hemos hecho cumpliendo los estándares web, veremos nuestras páginas correctamente a través de cualquiera de ellos e incluso a través de PDA y dispositivos móviles.

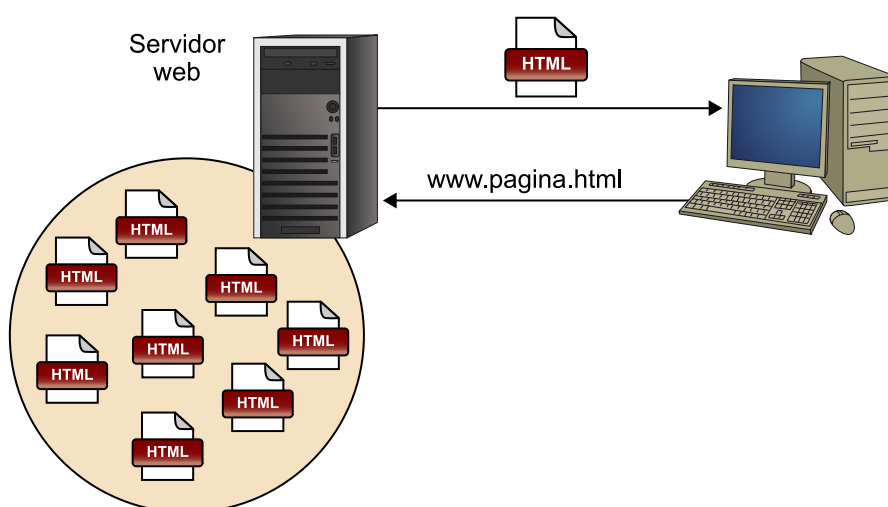
¿Qué hace un servidor web? ¿Y el navegador web?

Los servidores web están funcionando las 24 horas del día en Internet, esperando que algún navegador solicite cualquiera de las páginas web que almacena. Cuando el servidor recibe una solicitud de una página, la busca y la entrega al navegador para que este haga su trabajo.

El servidor web no es más que un ordenador especial conectado permanentemente a Internet y que almacena archivos HTML, imágenes, archivos CSS, sonidos, PDF, y cualquier otro tipo de archivo.

Esto es lo que sucede cuando escribimos en la barra de direcciones de nuestro navegador web `http://www.dominio.com` y pulsamos Enter:

- 1) El navegador hace la solicitud de la página `http://www.dominio.com` al servidor web que la almacena.
- 2) El servidor que contiene esta página recibe la petición, busca en su disco duro la página, y si la encuentra, la entrega al navegador juntamente con los archivos que la forman como imágenes, archivos css, etc.
- 3) El navegador recibe la página e interpreta el código HTML y CSS para mostrarla tal y como la vemos nosotros.



2. HTML

2.1. ¿Qué es HTML?

Como hemos visto, HTML es el código con el que se hacen las páginas web y que los navegadores son capaces de interpretar para que nosotros podamos visualizarlas correctamente.

Un fichero HTML contiene etiquetas delimitadoras que constituyen instrucciones que el navegador interpreta para mostrar la página por pantalla.

Aunque existen algunas excepciones, en general las etiquetas se indican por pares y se forman de la siguiente manera:

- Etiqueta de apertura: carácter <, seguido del **nombre de la etiqueta** (sin espacios en blanco) y terminando con el carácter >
- Etiqueta de cierre: carácter <, seguido del carácter /, seguido del **nombre de la etiqueta** (sin espacios en blanco) y terminado con el carácter >

De esta forma, la estructura de las etiquetas HTML es:

```
<nombre_etiqueta> ... </nombre_etiqueta>
```

HTML es un lenguaje de etiquetas (también llamado lenguaje de marcado) y las páginas web habituales están formadas por cientos o miles de pares de etiquetas.

La principal ventaja de los lenguajes de etiquetas como el HTML es que son muy sencillos de leer y escribir por parte de las personas y de los sistemas electrónicos.

Los ficheros pueden tener extensión htm o html.

Se puede editar un fichero HTML con cualquier editor de texto siempre y cuando se tenga en cuenta que deberemos guardar el archivo con una de las dos extensiones anteriores. Para estas prácticas usaremos el Bloc de Notas para los usuarios de Windows, y el TextEdit para los usuarios de Mac.

2.2. World Wide Web

El lenguaje HTML es un estándar reconocido en todo el mundo, cuyas normas define un organismo sin ánimo de lucro llamado World Wide Web Consortium (<http://www.w3.org>), más conocido como W3C. Como se trata de un estándar reconocido por las empresas y relacionadas con el mundo de Internet, una misma página HTML se visualiza de forma muy similar en cualquier navegador de cualquier sistema operativo.

Algunas de estas recomendaciones que indica el W3C para HTML abarcan desde la obligatoriedad de cerrar etiquetas, al uso de minúsculas para programar HTML.

La finalidad de estas recomendaciones es reducir la permisividad del lenguaje HTML y convertirlo en un lenguaje más estricto, más estandarizado para todos y menos caótico.

Un ejemplo del aspecto permisivo de HTML es que los nombres de etiquetas y atributos pueden ir tanto en minúsculas como en mayúsculas (se puede escribir `<head>` o `<HEAD>`), hay etiquetas que necesita un contenido, como la etiqueta `<a>` que indica un enlace y debe ir acompañada siempre de su correspondiente ``, y etiquetas que no, como `
`, que indica un salto de línea.

Además, hay navegadores aún más permisivos, de modo que si hay etiquetas sin su correspondiente cierre, intentan deducir dónde debería ir ese cierre y muestran la página más o menos de forma adecuada.

Existen dos variantes principales de la última versión del lenguaje HTML definido por el W3C: Transitional y Strict. En la variante Transitional, se pueden utilizar todas las etiquetas y atributos, pero en la variante Strict, estamos limitados a aquellos que no tengan nada que ver con la apariencia de la página.

Esta apariencia se debe definir en una hoja de estilos CSS, de modo que en la variante Strict no podemos utilizar la etiqueta `` (que define características del texto como color o tamaño) o `<center>` (alineación de la página en el centro), ni atributos como `bgcolor` (color de fondo de la página).

Nosotros usaremos la variante Transitional, dándonos un poco de permisividad en la declaración de las etiquetas HTML.

Por lo tanto, deberemos poner en nuestro archivo HTML, en el inicio del documento incluso antes de la etiqueta `<HTML>`, lo siguiente:

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

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

A pesar de eso, debemos tener en cuenta la tendencia actual, que consiste en dejar en manos de las hojas de estilo CSS gran parte o la totalidad de los aspectos que se refieren al formato en el que se presenta la web, como colores, fuentes y tamaños.

Así pues, procuraremos estructurar las páginas de esta forma en la medida de lo posible. Separando claramente el contenido (HTML) del formato con que se presenta ese contenido (CSS).

En muchos casos, el color, tamaño de la fuente, formato de los bloques de contenido,... vendrán determinados por las hojas de estilos CSS y el contenido (sin formato) por el lenguaje HTML.

Hay que tener en cuenta que esta forma de estructurar la creación de un sitio web facilita mucho el trabajo, nos evitará tener que estar buscando código para cambiar el aspecto de nuestras páginas web, podremos centralizar el formato de la web en hojas de estilo CSS.

2.3. Estructura de un documento HTML

Las páginas HTML se dividen en dos partes: la cabecera y el cuerpo. La cabecera incluye información sobre la propia página, como por ejemplo su título y la descripción. El cuerpo de la página incluye todos sus contenidos, como párrafos, titulares, tablas e imágenes.

El cuerpo (llamado *body* en inglés) contiene todo lo que el usuario ve en su pantalla, y la cabecera (llamada *head* en inglés) contiene todo lo que el usuario no ve en su pantalla a excepción del título de la página que se visualiza en la parte superior del navegador.

Este es el código HTML de una página sencilla:

```
<html>

<head>
<title>El lenguaje HTML</title>
</head>

<body>

<h1>HTML: la estructura y el contenido de una web</h1>



<p>El lenguaje HTML es <strong>tan sencillo</strong> que prácticamente se entiende sin estudiar
```

```
el significado de sus <em>etiquetas principales</em>.</p>

<h2>CSS: la apariencia de la web</h2>

<p>Los colores, el tipo de letra, el tamaño, etc. se definen a través de estilos CSS,
separándolos completamente de la estructura HTML de una página.</p>

</body>

</html>
```

Vamos a realizar ahora este primer ejemplo. Estos son los pasos que tenemos que seguir:

- 1) Abrimos un editor de archivos de texto como el Notepad (Win) o el TextEdit (Mac), y creamos un archivo nuevo.
- 2) Escribimos el código HTML del ejemplo. Hay que tener cuidado de no cometer errores tipográficos, y de cerrar todas las etiquetas HTML.
- 3) Guardamos el archivo con el nombre que queramos con la extensión .html, y en una carpeta de nuestro ordenador. Es importante que la imagen html.jpg se encuentre en la misma carpeta que el documento html.

Para que esto funcione correctamente, debemos utilizar un editor de texto sin formato como los que he mencionado. No podemos usar procesadores de texto como Word o Open Office.

Una vez creado y guardado el archivo HTML ya podemos abrirlo con nuestro navegador web y visualizaremos la página web.

El resultado debería ser como la siguiente imagen:



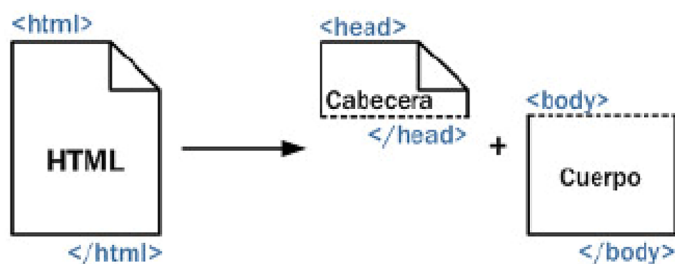
Si ya estamos visualizando nuestra primera página HTML en el navegador, en el menú superior pulsamos Ver > Código fuente. De hecho, podemos hacer esto en cualquier página de Internet. Si lo hacemos en alguna de las páginas que más visitamos veréis la cantidad de etiquetas HTML que puede llegar a tener una página compleja.

Si analizamos el código de nuestro primer ejercicio, es importante conocer las tres etiquetas principales de un documento HTML (`<html>`, `<head>`, `<body>`):

`<html>`: indica el comienzo y el final de un documento HTML. Ninguna etiqueta o contenido puede colocarse antes o después de la etiqueta `<html>`. En el interior de la etiqueta `<html>` se definen la cabecera y el cuerpo del documento HTML y todo lo que se coloque fuera de la etiqueta `<html>` se ignora.

`<head>`: delimita la parte de la cabecera del documento. La cabecera contiene información sobre el propio documento HTML, como por ejemplo su título y el idioma de la página, entre muchos otros datos. Los contenidos indicados en la cabecera no son visibles para el usuario, con la excepción de la etiqueta `<title>`, que se utiliza para indicar el título del documento y los navegadores lo muestran en la parte superior izquierda de la ventana del navegador.

`<body>`: delimita el cuerpo del documento HTML. El cuerpo contiene todos los contenidos que se muestran al usuario (párrafos de texto, titulares, imágenes, tablas, ...). En general, el `<body>` de un documento contiene cientos de etiquetas HTML, mientras que el `<head>` solamente contiene algunas pocas.



2.4. Elementos, etiquetas y atributos

Etiquetas HTML

Como hemos visto anteriormente, las etiquetas HTML que escribimos en nuestra página se encierran en los caracteres `<` y `>`.

Para hacerle entender al navegador la estructura de la página web, se utilizan pares de etiquetas alrededor del contenido.

Generalmente las etiquetas se especifican en pares, como `` y ``. La primera etiqueta del par es la etiqueta inicial que abre el código, y la segunda, la etiqueta final, que lo cierra. El texto entre las etiquetas inicial y final es el contenido del elemento y está sujeto al formato que éste le da.

Elemento HTML = etiqueta apertura + contenido + etiqueta cierre

Ejemplo: Elemento párrafo: `<p>Esto es un párrafo</p>`

Las etiquetas no distinguen entre mayúsculas y minúsculas, aunque los estándares recomiendan utilizar minúsculas.

Atributos de las etiquetas

Los atributos de las etiquetas HTML proveen información adicional acerca de las etiquetas HTML de la página. Por ejemplo, la etiqueta `<p>` puede tener un atributo para diferenciar visualmente a través de estilos un párrafo de otro:

`<p class="normal">Esto es un párrafo</p>`

El atributo `class` sirve para aplicar un estilo diferente mediante la definición de una clase en CSS.

Otros ejemplos de atributos de elementos HTML:

`<style type="text/css">` : El atributo `type` especifica qué tipo de lenguaje de estilos usaremos. En este caso, usaremos CSS.

`` : El atributo `href` nos dice el destino del vínculo que definimos con la etiqueta HTML `<a>`

`` : El atributo `src` especifica el nombre del archivo de imagen a mostrar en la etiqueta HTML ``

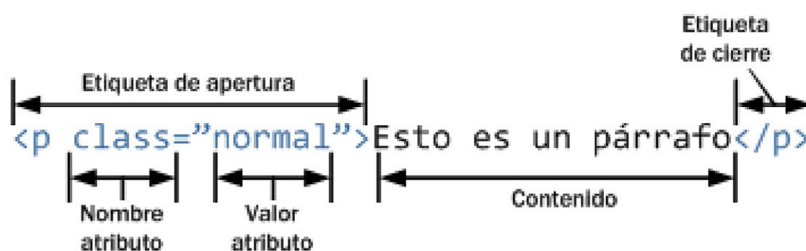
Elementos HTML

Además de etiquetas y atributos, HTML define el término *elemento* para referirse a las partes que componen los documentos HTML.

Aunque en ocasiones se habla de forma indistinta de "elementos" y "etiquetas", en realidad un elemento HTML es mucho más que una etiqueta, ya que está formado por:

- Una etiqueta de apertura
- Cero o más atributos
- Texto encerrado por la etiqueta
- Una etiqueta de cierre.

El texto encerrado por la etiqueta es opcional, ya que algunas etiquetas de HTML no pueden encerrar ningún texto. El siguiente esquema muestra un elemento HTML, formado por una etiqueta `<p>`, atributos y contenidos de texto:



La estructura mostrada en el esquema anterior es un elemento HTML, ya que comienza con una etiqueta de apertura (`<p>`), contiene cero o más atributos (`class="normal"`), dispone de un contenido de texto (Esto es un párrafo) y finaliza con una etiqueta de cierre (`</p>`).

Por otra parte, el lenguaje HTML clasifica a todos los elementos en dos grupos: elementos en línea (*inline elements* en inglés) y elementos de bloque (*block elements* en inglés).

La principal diferencia entre los dos tipos de elementos es la forma en la que ocupan el espacio disponible en la página. Los elementos de bloque empiezan una nueva línea y ocupan todo el espacio disponible hasta el final de la línea, aunque sus contenidos no lleguen hasta el final de la línea. Por su parte, los elementos en línea sólo ocupan el espacio necesario para mostrar sus contenidos.

Veamos un ejemplo escribiendo en un archivo nuevo el siguiente código:

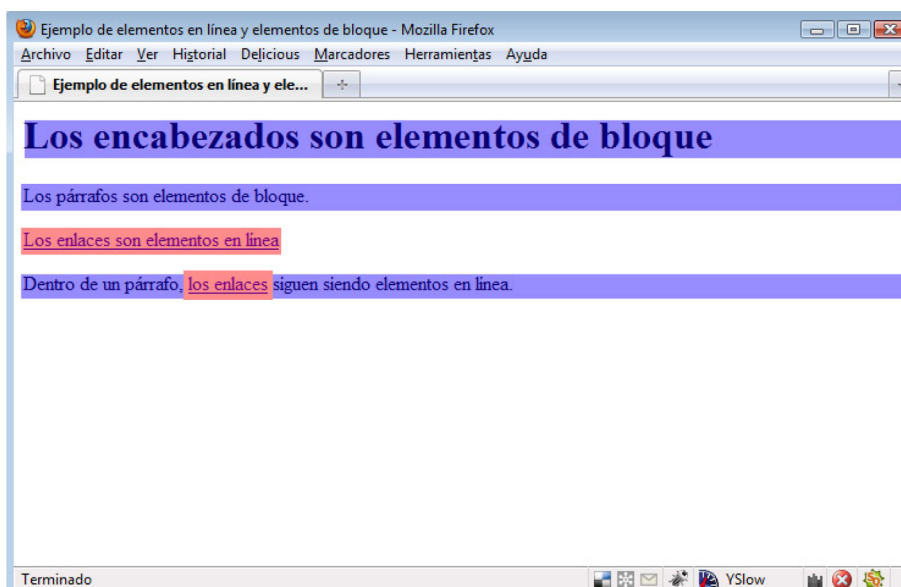
```
<html>

<head>
<title>Ejemplo de elementos en línea y elementos de bloque</title>
</head>

<body>
<h1>Los encabezados son elementos de bloque</h1>
<p>Los párrafos son elementos de bloque.</p>
<a href="http://www.google.com">Los enlaces son elementos en línea</a>
<p>Dentro de un párrafo, <a href="http://www.google.com">los enlaces</a> siguen siendo elementos
en línea.</p>
</body>

</html>
```

En la siguiente imagen se puede ver cómo se visualizaría el código HTML anterior. He coloreado de color azul los elementos de bloque, y de color rojo los elementos en línea. De esta forma es fácil hacerse una idea del espacio que ocupan los distintos elementos.



Los encabezados (h1, h2, h3, h4, h5, h6) son elementos de bloque, y el navegador reserva todo el espacio hasta el final de línea, haciendo un salto de línea al final de este.

El primer párrafo contiene un texto corto que sólo ocupa la mitad de la anchura de la ventana del navegador. De todas formas, el navegador también reserva para este

elemento HTML todo el espacio hasta el final de esa línea, por lo que resulta evidente que los elementos párrafo <p> también son elementos de bloque.

El primer enlace del ejemplo anterior también tiene un texto corto que ocupa solamente la mitad de la anchura de la ventana del navegador. En este caso, el navegador sólo reserva para el enlace el sitio necesario para mostrar sus contenidos. Si se añade otro enlace en esa misma línea, se mostraría a continuación del primer enlace, sin salto de línea. Por lo tanto, los elementos <a> son elementos en línea.

Por último, el segundo párrafo sigue ocupando todo el espacio disponible hasta el final de línea (por ser un elemento de bloque) y el enlace que se encuentra dentro del párrafo sólo ocupa el sitio necesario para mostrar sus contenidos (por ser un elemento en línea).

En general, los elementos de bloque pueden contener en su interior elementos en línea y otros elementos de bloque. Los elementos en línea sólo pueden contener texto u otros elementos en línea.

En otras palabras, un elemento de bloque no puede aparecer dentro de un elemento en línea. En cambio, un elemento en línea puede aparecer dentro de un elemento de bloque y dentro de otro elemento en línea.

Los elementos en línea definidos por HTML son: a, abbr, acronym, b, basefont, bdo, big, br, cite, code, dfn, em, font, i, img, input, kbd, label, q, s, samp, select, small, span, strike, strong, sub, sup, textarea, tt, u, var.

Los elementos de bloque definidos por HTML son: address, blockquote, center, dir, div, dl, fieldset, form, h1, h2, h3, h4, h5, h6, hr, isindex, menu, noframes, noscript, ol, p, pre, table, ul.

Los siguientes elementos también se considera que son de bloque: dd, dt, frame-set, li, tbody, td, tfoot, th, thead, tr.

Los siguientes elementos pueden ser en línea y de bloque según las circunstancias: button, del, iframe, ins, map, object, script.

De todos estos elementos solamente veremos los más importantes.

Comentarios en HTML

La etiqueta de comentario se utiliza para insertar un comentario en el código fuente HTML que será ignorado por el navegador. Se utilizan para documentar el código y facilitar su mantenimiento luego.

```
<!-- Esto es un comentario -->
```

Si probamos de añadir esta línea de código en un documento HTML, veremos cómo al visualizarlo en un navegador el comentario no aparecerá.

2.5. Apariencia de una página

Antes de entrar los diferentes elementos y etiquetas HTML, es muy importante remarcar bien y entender la diferencia entre el código HTML de una página y el código CSS que define la apariencia y estilo de la misma.

Como hemos visto, el lenguaje HTML nos da una forma para describir la estructura de contenidos de la página. Cuando el navegador muestra nuestro archivo HTML, utiliza su propio estilo por defecto para presentarlo y mostrarlo en la pantalla. Obviamente, esta apariencia que nos da el navegador por defecto no será de nuestro agrado y lo queremos personalizar y aplicar un diseño nuestro.

Aquí es cuando entran en escena los CSS. CSS nos da una forma para describir cómo queremos presentar y mostrar nuestro contenido. CSS es la abreviación de Cascading Style Sheet. Más adelante veremos en detalle los estilos CSS pero vamos a ver en un ejemplo cómo podemos cambiar el aspecto de nuestro ejemplo anterior.

El siguiente código lo tenemos que añadir dentro de las etiquetas `<head>` `</head>` (que recordamos que nos sirven para definir información sobre nuestra página), y normalmente se sitúa antes de cerrar la etiqueta HEAD.

```
<head>
<title>El lenguaje HTML</title>
<style type="text/css">
body {
    background-color: #000000;
    color: #FFFFFF;
    margin-left: 20%;
    margin-right: 20%;
    border: 1px dotted gray;
    padding: 10px 10px 10px 10px;
    font-family: sans-serif;
}</style>
```

```
</head>
```

Guardamos los cambios, y abrimos el documento con nuestro navegador para ver el resultado final.

Como podemos ver en el ejemplo anterior, la etiqueta HTML que da inicio a la definición de estilos es `STYLE`, con la etiqueta de apertura `<style>` y la de cierre `</style>`. Si nos fijamos un poco más, vemos como tiene un atributo para especificar el CSS: `type="text/css"`.

Dentro de las etiquetas `<style>` `</style>` se definen los estilos de los elementos HTML que queramos para darle una apariencia. Aquí es donde se pone la sintaxis CSS para definir los estilos, que es algo diferente al lenguaje HTML.

En el ejemplo, estamos dando unas propiedades visuales a todo el elemento BODY, por eso `body { }`. Leyendo la sintaxis CSS del ejemplo, y sin entrar en detalle aún,

podemos deducir que se está dando un color de fondo, unos márgenes, se define un borde, y una familia de tipografía.

Como todo este código y sintaxis CSS está en la cabecera `<HEAD>` `</HEAD>`, no se visualiza en sí directamente, sino que se está aplicando en los elementos que se definen. Recordamos que lo que se muestra de una página web es lo que se define dentro de las etiquetas `<BODY>` `</BODY>`.

Si lo hemos hecho correctamente, en el navegador debería salir una página como esta:



Más adelante entraremos en profundidad con los CSS, y veremos cómo darle la apariencia que nosotros queramos a nuestra página web.

Veremos también que los CSS se pueden definir en la misma página HTML, o bien se puede definir todo en un archivo externo .css para que todas las páginas de nuestro sitio web puedan usar los mismos estilos.

2.6. Guía de elementos básicos HTML

A continuación veremos un listado de los elementos HTML más relevantes. Ya sabemos que estas etiquetas deben ponerse dentro de las etiquetas BODY `<body>` `</body>` para después ver el resultado con el navegador.

Haced las pruebas que queráis con cada elemento con cualquiera de los ejemplos vistos anteriormente, o bien con un archivo nuevo con toda la estructura de un archivo HTML. Si añadís cualquiera de los siguientes elementos HTML con sus etiquetas de inicio y de cierre, y su contenido, debéis guardar los cambios, y abrir o actualizar la página con el navegador si ya la tenéis abierta.

<code><p></code> <code></p></code>	Párrafo. Los párrafos creados con HTML son elementos de bloque, por lo que siempre ocupan toda la anchura de la ventana del navegador. Además, no tienen atributos específicos, pero sí que se les pueden asignar los atributos comunes de HTML básicos, de internacionalización y de eventos.
<code><h1></code> <code></h1></code> <code><h2></code> <code></h2></code> <code><h3></code> <code></h3></code> <code><h4></code> <code></h4></code> <code><h5></code> <code></h5></code> <code><h6></code> <code></h6></code>	Encabezados. Define los títulos de las secciones de mayor importancia de la página. Al igual que la etiqueta <code><p></code> , las etiquetas de título de sección son elementos de bloque. Los navegadores asignan de forma automáticamente el tamaño del título de cada sección en función de su importancia. Así, los títulos de sección <code><h1></code> se muestran con el tamaño de letra más grande, ya que son el nivel jerárquico superior, mientras que los títulos de sección <code><h6></code> se visualizan con un tamaño de letra muy pequeño, adecuado para el nivel jerárquico de menor importancia. Evidentemente, el aspecto que los navegadores aplican por defecto a los títulos de sección se pueden modificar utilizando las hojas de estilos de CSS.
<code>
</code> <code>
</code>	Salto de línea. Para hacer un salto de línea en un punto en concreto sin querer terminar un párrafo. Esta etiqueta crea una nueva línea en un punto y fuerza a que el texto que sigue se muestre en la línea inferior. La etiqueta <code>
</code> es una de las pocas etiquetas especiales de HTML. La particularidad de <code>
</code> es que es una etiqueta vacía, es decir, no encierra ningún texto. En el HTML Transitional se acepta escribir esta etiqueta de estas dos formas <code>
</code> o <code>
</code> , aunque los estándares recomiendan usar la segunda.
<code></code> <code></code>	Énfasis. Realza la importancia del texto que encierra. La etiqueta <code></code> marca un texto indicando que su importancia es mayor que la del resto del texto. Por defecto, los navegadores muestran los elementos <code></code> en cursiva para hacer evidente su importancia. Mediante CSS podremos cambiar el aspecto de esta etiqueta a nuestro gusto. Antes para hacer una cursiva se usaban las etiquetas <code><i></code> <code></i></code> . Aunque se permite su uso, ahora se aconseja usar <code></code> <code></code> , ya que la cursiva la haremos en CSS. Es un elemento en línea.

<code> </code>	Énfasis más acentuado. Realza con la máxima importancia el texto que encierra. La etiqueta <code></code> indica que un determinado texto es de la mayor importancia dentro de la página. Por defecto, los navegadores muestran los elementos <code></code> en negrita, para indicar que son los más importantes. Antes se usaba para hacer una negrita las etiquetas <code> </code> . Aunque se permite, ahora se aconseja usar <code> </code> , ya que la cursiva la haremos en CSS. Es un elemento en línea.
<code><ins> </ins></code>	Inserción. Se emplea para marcar una modificación en los contenidos originales consistente en la inserción de un nuevo contenido. El texto aparecerá subrayado. No se usa demasiado. Es un elemento en línea.
<code> </code>	Borrado. Se emplea para marcar una modificación en los contenidos originales consistente en el borrado de cierto contenido. El texto aparecerá tachado. No se usa demasiado. Es un elemento en línea.
<code><blockquote> </blockquote></code>	Citas. Se emplea para indicar que el texto que encierra es una cita textual de otro texto externo. Para indicar de forma clara que el texto es una cita externa, los navegadores muestran por defecto el texto del elemento <code><blockquote></code> con un gran margen en la parte izquierda. Es un elemento de bloque.
<code><q> </q></code>	Comillas. Se emplea para indicar al navegador que es un texto con comillas. No es necesario añadir las comillas. El navegador nos las mostrará. Con CSS podremos cambiar el aspecto de las comillas a nuestro gusto. Es un elemento en línea.
<code><abbr> </abbr></code>	Abreviaturas. Se emplea para marcar las abreviaturas del texto y proporcionar el significado de esas abreviaturas. Es un elemento en línea. Atributo específico: title = "texto" - Indica el significado completo de la abreviatura Ejemplo: <code><abbr title="Barcelona">Barna</abbr></code> . El texto que define el atributo title aparecerá al mantener el ratón quieto sobre el texto Barna.
<code><acronym> </acronym></code>	Acrónimos o siglas. Se emplea para marcar las siglas o acrónimos del texto y proporcionar el significado de esas siglas. Es un elemento en línea. Atributo específico: title = "texto" - Indica el significado completo del acrónimo o sigla Ejemplo: <code><acronym title="Cascade Style Sheet">CSS</acronym></code> . El texto que define el atributo title aparecerá al mantener el ratón quieto sobre el texto CSS.
<code><address> </address></code>	Dirección. Define un elemento de dirección.
<code><sup> </sup></code>	Superíndice. Elemento en línea.
<code><sub> </sub></code>	Subíndice. Elemento en línea.
<code><pre> </pre></code>	Texto preformateado. Los espacios en blanco en el código son ignorados por el navegador. En ocasiones, es necesario mostrar los espacios en blanco de un texto que no se puede modificar. Muestra el texto que encierra tal y como está escrito (respetando los espacios en blanco).

2.7. Caracteres especiales

Una consideración importante directamente relacionada con el texto de las páginas HTML es la codificación de los caracteres y la inserción de caracteres especiales. Algunos de los caracteres que se utilizan habitualmente en los textos no se pueden incluir directamente en las páginas web:

- Los caracteres que utiliza HTML para definir sus etiquetas (<, > y ") no se pueden utilizar libremente.
- Los caracteres propios de los idiomas que no son el inglés (ñ, á, ç, ò, ï, etc.) pueden ser problemáticos dependiendo de la codificación de caracteres utilizada.

La solución a la primera limitación consiste en sustituir los caracteres reservados de HTML por unas expresiones llamadas entidades HTML y que representan a cada carácter:

Entidad	Carácter
<	<
>	>
&	&
"	"
 	(espacio en blanco)
'	'

Por otra parte, los caracteres propios de los idiomas diferentes al inglés también pueden ser problemáticos. El motivo es que desde que se crea una página web hasta que llega al navegador del usuario, intervienen numerosos procesos que pueden afectar a la correcta visualización de los acentos:

- El diseñador crea la página web con su editor HTML (por ejemplo, Dreamweaver).
- Si se trata de una aplicación dinámica, el programador recorta la página HTML del diseñador y la mezcla con el resto del código de la aplicación (por ejemplo, PHP).
- El servidor web almacena las páginas HTML estáticas o el código de la aplicación web y sirve las páginas solicitadas por los usuarios.
- El usuario solicita y visualiza las páginas web a través de su navegador.

Si en todos los procesos anteriores se utiliza la misma codificación de caracteres, los caracteres propios de los idiomas se pueden escribir directamente:

```
<p>Este párrafo contiene caracteres acentuados y se almacena en formato UTF-8</p>
```

La palabra *párrafo* del ejemplo anterior incluye la letra á. Si el editor HTML del diseñador utiliza la codificación UTF-8, el entorno de desarrollo del programador también utiliza UTF-8, el servidor web sirve las páginas con esa codificación y el navegador del usuario es capaz de visualizar las páginas con formato UTF-8, el texto anterior se verá correctamente en el navegador del usuario.

Sin embargo, muchas veces no es posible que todos los procesos involucrados utilicen la misma codificación de caracteres. Por limitaciones técnicas o por decisiones de los diseñadores y programadores, los textos pueden pasar de codificación UTF-8 a codificación ISO-8859 en cualquier momento. Si se produce este cambio sin realizar una conversión correcta, el navegador del usuario mostrará caracteres extraños en todos los acentos y en todas las letras como la ñ.

Para especificar en qué codificación estamos trabajando, hay que utilizar el siguiente código dentro de la cabecera HEAD (<head> </head>):

codificación ISO-8859

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

codificación UTF-8

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

La solución más sencilla para asegurar que todos estos caracteres potencialmente problemáticos se van a visualizar correctamente en el navegador del usuario consiste en sustituir cada carácter problemático por su entidad HTML:

Entidad	Carácter
ñ	ñ
Ñ	Ñ
á	á
é	é
í	í
ó	ó
ú	ú

Entidad	Carácter
Á	Á
É	É
Í	Í
Ó	Ó
Ú	Ú
€	€

Así, el párrafo de texto del ejemplo anterior se podría escribir de la siguiente manera:

```
<p>Este párrafo contiene caracteres acentuados y se almacena en formato UTF-8</p>
```

Si se utilizan las entidades HTML en lugar de los caracteres problemáticos, es indiferente pasar de una codificación de caracteres a otra diferente. En la Wikipedia se puede consultar la lista completa de las 252 entidades HTML definidas (http://en.wikipedia.org/wiki/List_of_XML_and_HTML_character_entity_references).

2.8. Enlaces HTML

HTML ofrece muchas de las posibilidades de publicación convencionales para la creación de textos enriquecidos y documentos estructurados, pero lo que lo separa de la mayoría de los otros lenguajes para el formato de documentos son sus características para hipertexto y para documentos interactivos.

La incorporación del hipertexto fue una de las claves del éxito del lenguaje HTML, ya que permitió crear documentos interactivos que proporcionan información adicional cuando se solicita. El elemento principal del hipertexto es el "hiperenlace", también llamado "enlace web" o simplemente "enlace".

Los enlaces se utilizan para establecer relaciones entre dos recursos. Aunque la mayoría de enlaces relacionan páginas web, también es posible enlazar otros recursos como imágenes, documentos y archivos.

Un hiperenlace, hipervínculo, o vínculo, no es más que un enlace (en inglés *link*), que al ser pulsado nos lleva a una página o archivo.

Aquellos elementos (texto, imágenes, etc.) sobre los que se desee insertar un enlace han de encontrarse entre las etiquetas de ancla `<a>` y ``.

A través del atributo href, se especifica la página que se visualizará cuando el usuario haga clic en el enlace.

Ejemplo

Esto es lo que se vería en el navegador

Visita Google

Código HTML

```
<a href="http://www.google.com">Visita Google</a>
```

Probamos este ejemplo escribiendo este pequeño código dentro de las etiquetas HTML y veremos cómo funciona el enlace.

<a> 	Enlace. Se emplea para enlazar todo tipo de recursos. Es un elemento en línea. Atributo específico: href = "url" - Indica la URL del recurso que se quiere enlazar target = "_blank" - Indica que la página con la que se quiere enlazar se debe abrir en una ventana nueva del navegador
----------	---

2.9. URL

El acrónimo URL (del inglés *uniform resource locator*) hace referencia al identificador único de cada recurso disponible en Internet. Las URL son esenciales para crear los enlaces, pero también se utilizan en otros elementos HTML, como las imágenes y los formularios.

Las URL permiten que cada página HTML publicada en Internet tenga un nombre único que permita diferenciarla de las demás. De esta forma es posible crear enlaces que apunten de forma inequívoca a una determinada página.

Si se accede a la página principal de Google, la dirección que muestra el navegador es: <http://www.google.com>.

La cadena de texto <http://www.google.com> es la URL completa de la página principal de Google.

La URL de las páginas es imprescindible para crear los enlaces, ya que permite distinguir una página de otra.

El segundo objetivo de las URL es el de permitir la localización eficiente de cada recurso de Internet. Para ello es necesario comprender las diferentes partes que forman las URL. Una URL sencilla siempre está formada por las mismas tres partes. Si por ejemplo se considera la siguiente URL:

<http://www.mipagina.es/carpeta1/archivo.html>

Las partes que componen la URL anterior son:

- Protocolo (http://): el mecanismo que debe utilizar el navegador para acceder a ese recurso. Todas las páginas web utilizan http://. Las páginas web **seguras** (por ejemplo, las de los bancos y las de los servicios de email) utilizan https:// (se añade una letra s).
- Servidor (www.mipagina.es): simplificando mucho su explicación, se trata del ordenador en el que se encuentra guardada la página a la que se quiere acceder. Los navegadores son capaces de obtener la dirección de cada servidor a partir de su nombre.
- Ruta (/carpeta1/archivo.html): **camino** que se debe seguir, una vez que se ha llegado al servidor, para localizar el recurso específico al que se quiere acceder.

Por tanto, las URL no sólo identifican de forma única cada recurso de Internet, sino que también proporcionan a los navegadores la información necesaria para poder llegar hasta ese recurso.

2.9.1. Enlaces relativos y absolutos

El modo de escribir el enlace mediante el atributo href de la etiqueta <a> es mediante URL relativas o absolutas.

Las URL absolutas incluyen todas las partes de la URL (protocolo, servidor y ruta), mientras que las URL relativas prescinden de algunas partes de la URL para hacerlas más cortas.

Las URL relativas se construyen a partir de las URL absolutas y prescinden de la parte del protocolo, del nombre del servidor e incluso de parte o toda la ruta del recurso enlazado.

Aunque las URL relativas pueden ser difíciles de entender para los que comienzan con HTML, son tan útiles que todos los sitios web las utilizan.

Una URL relativa es una versión abreviada de una URL absoluta. Su objetivo es eliminar todas las partes de la URL absoluta que se pueden adivinar a partir de la información de contexto de la página web. En otras palabras, las URL relativas aprovechan la "inteligencia" de los navegadores para crear URL incompletas que los navegadores pueden completar deduciendo la información que falta.

Veamos varios ejemplos:

1) Página origen y destino están en el mismo directorio

Tenemos una página que quiere enlazar con otra.

Si quisiera enlazar de la primera página a la segunda podría hacerlo usando tanto la URL absoluta como la URL relativa.

```
http://www.mipagina.com/carpeta1/archivo1.html  
http://www.mipagina.com/carpeta1/archivo2.html
```

Si uso URL absoluta, el enlace sería:

http://www.mipagina.com/carpeta1/archivo2.html

Si uso URL relativas, el enlace sería:

archivo2.html

Se elimina toda la parte de la URL que se puede deducir, de la página actual. Como la página actual está dentro de la carpeta1, la ruta relativa empieza desde esta misma carpeta.

2) La página destino del enlace se encuentra en una carpeta superior

El archivo1.html se encuentra dentro de la carpeta llamada carpeta2, que al mismo tiempo está dentro de la carpeta llamada carpeta1. En cambio, el archivo2.html se encuentra en la carpeta llamada carpeta1. Por lo tanto, los dos recursos (archivo1.html y archivo2.html) ya no se encuentran dentro de la misma carpeta, sino que el archivo2.html se encuentra en un nivel superior.

```
Origen: http://www.mipagina.com/carpeta1/carpeta2/archivo1.html  
Destino: http://www.mipagina.com/carpeta1/archivo2.html
```

La URL absoluta del enlace sería:

http://www.mipagina.com/carpeta1/carpeta2/archivo2.html

La URL relativa del enlace sería:

../archivo2.html

Cuando el navegador encuentra la URL relativa ../archivo2.html, sabe que para encontrar el recurso enlazado (archivo2.html) tiene que subir un nivel desde el lugar en el que se encuentra esa URL relativa. La página que incluye esa URL se encuentra en el directorio carpeta1/carpeta2/, por lo que subir un nivel equivale entrar en el directorio carpeta1/.

De la misma forma, si quisiéramos subir dos niveles, deberíamos escribir ../dos veces seguidas.

Origen: <http://www.mipagina.com/carpeta1/carpeta2/archivo1.html>

Destino: <http://www.mipagina.com/archivo2.html>

La URL absoluta del enlace sería:

<http://www.mipagina.com/archivo2.html>

La URL relativa del enlace sería:

<../../archivo2.html>

Y si la página [archivo2.html](#) está en un nivel superior y además en otra carpeta, el enlace sería de la siguiente forma:

Origen: <http://www.mipagina.com/carpeta1/carpeta2/archivo1.html>

Destino: <http://www.mipagina.com/carpeta3/archivo2.html>

La URL absoluta del enlace sería:

<http://www.mipagina.com/carpeta3/archivo2.html>

La URL relativa del enlace sería:

<../../carpeta3/archivo2.html>

3) La página destino del enlace se encuentra en una carpeta inferior

Ejemplo muy parecido pero más sencillo aún:

Origen: <http://www.mipagina.com/carpeta1/carpeta2/archivo1.html>

Destino: <http://www.mipagina.com/carpeta1/carpeta2/carpeta3/archivo2.html>

La URL absoluta del enlace sería:

<http://www.mipagina.com/carpeta1/carpeta2/carpeta3/archivo2.html>

La URL relativa del enlace sería:

<carpeta3/archivo2.html>

4) La página destino del enlace se encuentra en direcciones muy diferentes respecto a la página origen

Cuando el origen y el destino de un enlace se encuentran muy alejados (pero en el mismo servidor), las URL relativas se pueden complicar en exceso. Aunque es posible utilizar `../` para subir por la jerarquía de directorios y se puede entrar en cualquier directorio indicando su nombre, las URL relativas que se obtienen son demasiado largas y complicadas.

En estos casos, lo más sencillo es indicar la ruta completa hasta el recurso enlazado comenzando desde la raíz del servidor web. Por lo tanto, estas URL relativas sólo omiten el protocolo y el nombre del servidor.

Origen: <http://www.mipagina.com/carpeta1/carpeta2/carpeta2/archivo1.html>
Destino: <http://www.mipagina.com/carpeta4/carpeta5/carpeta6/archivo2.html>

La URL absoluta del enlace sería:

<http://www.mipagina.com/carpeta4/carpeta5/carpeta6/archivo2.html>

La URL relativa del enlace sería:

</carpeta4/carpeta5/carpeta6/archivo2.html>

Cuando la URL relativa comienza por /, el navegador considera que es la ruta completa comenzando desde la raíz del servidor, por lo que sólo le añade el protocolo y el nombre del servidor origen.

2.9.2. Otro tipo de enlaces habituales

Ya hemos visto cómo se enlaza un recurso web con otro, y recordamos que un recurso web puede ser tanto un archivo html, como imágenes (archivos .jpg, .gif, ...) archivos pdf, archivo .doc, etc.

Ahora lo que veremos es otro tipo de enlaces muy habituales en la creación de páginas web:

1) Enlace a la home del sitio web

Podemos usar ruta relativa o absoluta.

URL absoluta: `Inicio`

URL relativa: `Inicio`

Lo que veremos en el navegador en ambos casos es solamente la palabra Inicio

Los dos funcionan igual de bien. Al pulsar el enlace anterior desde cualquier página web, se vuelve directamente a la página de inicio, home o página principal del sitio web.

2) Enlace a un correo electrónico

`Solicita más información`

Al hacer clic sobre el enlace anterior, se abre automáticamente el programa de correo electrónico del ordenador del usuario y se establece la dirección de envío al valor indicado después de mailto:

La sintaxis es la misma que la de un enlace normal, salvo que se cambia el prefijo

http:// por mailto:

3) Enlace con un archivo CSS

```
<link rel="stylesheet" type="text/css" href="/css/comun.css">
```

Recordamos que los archivos CSS nos darán información de la apariencia, colores, etc. de la página HTML.

Se usa un elemento HTML diferente al `<a>` ``. El elemento que se usa para este tipo de enlace es `<link>` y se debe poner dentro de la cabecera HEAD (entre las etiquetas `<head>` `</head>`).

2.10. Listas

Las listas sirven para agrupar diferentes palabras o frases mediante un listado. Uno de los usos más habituales de las lista es la creación de un menú. Más adelante veremos que mediante CSS podemos dar a una lista y a los enlaces que la forman una apariencia de menú.

El lenguaje HTML define tres tipos diferentes de listas para agrupar los elementos:

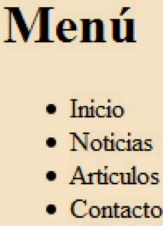
- Listas no ordenadas
- Listas ordenadas
- Listas de definición

1) Listas no ordenadas

Una lista no ordenada es un conjunto de elementos relacionados entre sí pero para los que no se indica un orden o secuencia determinados. Normalmente, cada elemento del listado se marca con alguna viñeta, que se puede definir en CSS. La etiqueta `` encierra todos los elementos de la lista y la etiqueta `` cada uno de sus elementos individualmente.

<code></code> <code></code>	Lista no ordenada. Se emplea para definir listas no ordenadas. Es un elemento de bloque.
<code></code> <code></code>	Elemento de una lista. Se emplea para definir los elementos de las listas (ordenadas y no ordenadas). Es un elemento de bloque.

Ejemplo

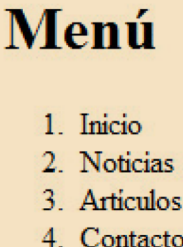
Código HTML	Vista navegador
<pre><h1>Menú</h1> Inicio Noticias Artículos Contacto </pre>	

2) Listas ordenadas

Son listas de palabras o frases marcadas con números o letras. La lista ordenada se define con la etiqueta ``. Los elementos de la lista se definen mediante la etiqueta ``, la misma que se utiliza en las listas no ordenadas.

<code> </code>	Lista ordenada. Se emplea para definir listas ordenadas. Es un elemento de bloque.
-------------------------------------	---

Ejemplo:

Código HTML	Vista navegador
<pre><h1>Menú</h1> Inicio Noticias Artículos Contacto </pre>	

3) Listas de definición

Las listas de definición apenas se utilizan en la mayoría de páginas HTML. Su funcionamiento es similar al de un diccionario, ya que cada elemento de la lista está formado por términos y definiciones. La etiqueta `<dl>` crea la lista de definición y las etiquetas `<dt>` y `<dd>` definen respectivamente el término y la descripción de cada elemento de la lista.

2.11. Imágenes

Para insertar una imagen en una página web es necesario almacenarla como un archivo independiente del documento HTML.

Las imágenes son un recurso más de todo el sitio web, son archivos independientes de los archivos HTML que forman y construyen las páginas. Por lo tanto, deberemos tener muy claro la estructura de archivos de todo nuestro sitio web, y en este caso, dónde tenemos las imágenes guardadas para decir al navegador las rutas hacia ellas.

Los tipos de archivos más admitidos en entorno web son JPEG y GIF. Los archivos JPEG suelen utilizarse para representar fotografías, mientras que los de tipo GIF, para iconos e imágenes que no requieran de un número elevado de colores.

Los nombres de los archivos que contienen imágenes, igual que los de los documentos HTML, son sensibles a mayúsculas/minúsculas y por lo tanto, .gif no es lo mismo que .GIF, con lo que al referirnos al archivo debemos escribir su nombre y extensión respetando mayúsculas y minúsculas.

Así pues, es aconsejable guardar los archivos con extensión en minúsculas, y para los nombres de los archivos es aconsejable no usar caracteres especiales, acentos ni espacios en blanco. Aunque es probable que funcione en local (en vuestro ordenador), pueden dejar de funcionar una vez subamos todos los archivos a un servidor.

	Imagen. Se emplea para enlazar todo tipo de recursos. Es un elemento en línea. Atributo específico: src = "url" - Indica la URL de la imagen alt = "texto" - Descripción corta de la imagen height = "píxeles" - Altura de la imagen width = "píxeles" - Anchura de la imagen
---------	---

Código HTML

```

```

Con el atributo src indicamos la ruta donde está guardada la carpeta.

Los atributos width y height sirven para indicar el tamaño de la imagen. Es importante no abusar de estos dos atributos para disminuir el tamaño de imágenes grandes, ya que el tamaño en Kb que ocupan sigue siendo el mismo. El caso ideal es que las imágenes ya estén guardadas con el tamaño en que se presentarán en la web.

También hay que procurar mantener las proporciones, ya que si no se creará el antiestético efecto de imágenes deformadas (alargadas o achatadas, por ejemplo).

Especificar la anchura y la altura no es imprescindible. Si no lo hacemos, el navegador lo calculará por nosotros. Sin embargo, la ventaja de hacerlo es que el navegador, al llegar al punto donde debe insertar la imagen, reserva ya el espacio necesario que le especifica los atributos `height` y `width`, y sigue cargando el resto de la página. Si no ponemos estos atributos, el navegador tardará un poquito más en saber las medidas de la imagen.

El atributo `alt` nos permite incluir una descripción en texto de la imagen. Será utilizada por los navegadores cuando no puedan mostrar la imagen: navegadores solo texto, como el `lynx`, o navegadores que tengan deshabilitada la opción de abrir gráficos e imágenes... o navegadores que en ese momento concreto no puedan acceder a la imagen. Es también útil para los lectores de personas con discapacidades visuales.

2.12. Tablas

Desde sus primeras versiones, HTML incluyó el soporte para crear tablas de datos en las páginas web. Además de ser sencillo, el modelo definido por HTML es muy flexible y bastante completo.

Las tablas en HTML utilizan los mismos conceptos de filas, columnas, cabeceras y títulos que los que se utilizan en cualquier otro entorno de publicación de documentos.

Las tablas de HTML pueden contener elementos simples, agrupaciones de filas y de columnas, cabeceras y pies de tabla, subdivisiones, cabeceras múltiples y otros elementos complejos.

A pesar de que las tablas HTML son fáciles de comprender y utilizar, son uno de los elementos más polémicos de HTML. El problema de las tablas es que no siempre se utilizan adecuadamente. Aunque parezca obvio, las tablas se deben utilizar para mostrar información tabular.

Hasta hace unos años, las tablas también se utilizaban para definir la estructura de las páginas web. La cabecera de la página era una fila de una gran tabla, el pie de página era otra fila de esta tabla y la zona de contenidos estaba formada por varias columnas dentro de esa gran tabla.

Aunque algunos malos diseñadores siguen utilizando hoy en día las tablas para definir la estructura completa de las páginas web, se trata de una técnica obsoleta y nada recomendable.

El motivo es que se complica en exceso el código HTML y su mantenimiento es muy complejo. La solución correcta para definir la estructura de las páginas consiste en la utilización de hojas de estilos CSS.

Ejemplo:

Código HTML	Vista navegador												
<pre><table> <tr> <td>Curso</td> <td>Horas</td> <td>Horario</td> </tr> <tr> <td>CSS</td> <td>20</td> <td>16:00 - 20:00</td> </tr> <tr> <td>HTML</td> <td>20</td> <td>16:00 - 20:00</td> </tr> <tr> <td>Dreamweaver</td> <td>60</td> <td>16:00 - 20:00</td> </tr> </table></pre>	<table><tr><th>Curso</th><th>Horas</th><th>Horario</th></tr><tr><td>CSS</td><td>20</td><td>16:00 - 20:00</td></tr><tr><td>HTML</td><td>20</td><td>16:00 - 20:00</td></tr><tr><td>Dreamweaver</td><td>60</td><td>16:00 - 20:00</td></tr></table>	Curso	Horas	Horario	CSS	20	16:00 - 20:00	HTML	20	16:00 - 20:00	Dreamweaver	60	16:00 - 20:00
Curso	Horas	Horario											
CSS	20	16:00 - 20:00											
HTML	20	16:00 - 20:00											
Dreamweaver	60	16:00 - 20:00											

La etiqueta `<table>` encierra todas las filas y columnas de la tabla. Las etiquetas `<tr>` (del inglés *table row*) definen cada fila de la tabla y encierran todas las columnas. Por último, la etiqueta `<td>` (del inglés *table data cell*) define cada una de las columnas de las filas, aunque realmente HTML no define columnas sino celdas de datos.

Al definir una tabla, se debe pensar en primer lugar en las filas que la forman y a continuación en las columnas. El motivo es que HTML procesa primero las filas y por eso las etiquetas `<tr>` aparecen antes que las etiquetas `<td>`.

<code><table> </table></code>	Tabla. Se emplea para definir tablas de datos. Elemento de bloque.
<code><tr> </tr></code>	Fila de tabla. Se emplea para definir cada fila de las tablas de datos. Es un elemento de bloque.
<code><td> </td></code>	Celda de tabla. Se emplea para definir cada una de las celdas que forman las filas de una tabla, es decir, las columnas de la tabla. Es un elemento tipo bloque. Atributos específicos más importantes <code>scope = "col, row, colgroup, rowgroup"</code> - Indica las celdas para las que esta celda será su cabecera. Ej: <code>scope="col"</code> indica que esta celda es la cabecera de todas las demás celdas que están en la misma columna. <code>colspan = "numero"</code> - Número de columnas que ocupa esta celda. <code>rowspan = "numero"</code> - Número de filas que ocupa esta celda.

Normalmente, algunas de las celdas de la tabla se utilizan como cabecera de las demás celdas de la fila o de la columna. En este caso, HTML define la etiqueta `<th>` (del inglés *table header cell*) para indicar que una celda es cabecera de otras celdas.

<code><th> </th></code>	Celda cabecera de tabla. Se emplea para definir las celdas que son cabecera de una fila o de una columna de la tabla. Elemento de bloque. Mismos atributos que <code><td></code> .
-------------------------------------	---

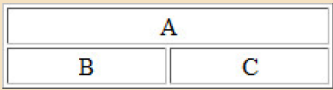

Los atributos de la etiqueta `<th>` son idénticos que los atributos definidos para la etiqueta `<td>`.

En este caso, el atributo más utilizado es `scope`, que permite indicar si la celda es cabecera de la fila o de la columna (`<th scope="row">` y `<th scope="col">` respectivamente).

Por otra parte, HTML define la etiqueta `<caption>` para establecer la leyenda o título de una tabla. La etiqueta debe colocarse inmediatamente después de la etiqueta `<table>` y cada tabla sólo puede incluir una etiqueta `<caption>`.

<code><caption> </caption ></code>	Leyenda o título de tabla. Se emplea para definir la leyenda o título de una tabla. Elemento en línea.
--	---

Ejemplo:

Código HTML	Vista navegador
<pre><table border="1"> <tr> <td width="200" colspan="2" align="center">A</td> </tr> <tr> <td width="100" align="center">B</td> <td width="100" align="center">C</td> </tr> </table></pre>	
<pre><table border="1"> <tr> <td width="100" align="center">A</td> <td rowspan="2" width="100" align="center">B</td> </tr> <tr> <td width="100" align="center">C</td> </tr> </table></pre>	

Es importante remarcar que atributos que se ven en estos ejemplos, como **align** o **border**, están en desuso porque los estándares recomiendan que se use mediante CSS, ya que hacen referencia a temas de visualización.

2.13. Maquetación de páginas más complejas

Hasta ahora hemos estado viendo un gran número de elementos HTML que nos sirven para marcar y estructurar nuestras páginas (tablas, listas, párrafos, imágenes,...). Aunque con estos elementos ya estamos creando páginas web, no podemos hacer maquetaciones complejas.

La mayoría de páginas HTML disponen de estructuras complejas formadas por varias columnas de contenidos y otro tipo de divisiones. Utilizando exclusivamente HTML no es posible crear estas estructuras complejas, ya que es imprescindible emplear las hojas de estilos CSS.

No obstante, los estilos de CSS necesitan la ayuda de HTML/XHTML para crear los diseños más avanzados. En concreto, el código HTML se encarga de agrupar los elementos de la página en diferentes divisiones en función de su finalidad: la zona de la cabecera de la página, la zona de contenidos, una zona lateral para el menú y otras secciones menores, la zona del pie de página, etc.

En la siguiente imagen se muestran algunas de las zonas definidas de la página principal del sitio <http://www.topimatge.com>:



Para agrupar los elementos que forman cada zona o división de la página se utiliza la etiqueta `<div>`:

<code><div> </div></code>	Divisiones. Agrupa elementos de bloque. Elemento de bloque.
---------------------------------------	--


El nombre de la etiqueta `div` tiene su origen en la palabra *división*, ya que esta etiqueta define zonas o divisiones dentro de una página HTML.

Las páginas web complejas que están bien diseñadas utilizan decenas de etiquetas `<div>`. Con mucha diferencia, los atributos más utilizados con esta etiqueta son `id` (para identificar la capa de forma única) y `class` (para aplicar a la capa estilos CSS).

Cuando abordemos el tema CSS veremos en detalle esta etiqueta `div`.

Por último, si observas el código HTML de algunas páginas web complejas, veremos que la mayoría utilizan los mismos nombres para identificar sus divisiones. Los nombres más comunes, y sus equivalentes en inglés, se muestran a continuación:

- contenedor (wrapper) suele encerrar la mayor parte de los contenidos de la página y se emplea para definir las características básicas de la página: su anchura, sus bordes, imágenes laterales, si se centra o no respecto de la ventana del navegador, etc.
- cabecera (header) que incluye todos los elementos invariantes de la parte superior de la página (logotipo, imagen o banner, cuadro de búsqueda superior, etc.).
- contenido (content) engloba el contenido principal del sitio (la zona de noticias, la zona de artículos, la zona de productos, etc. dependiendo del tipo de sitio web).
- menú (menu) se emplea para agrupar todos los elementos del menú lateral de navegación de la página.
- pie (footer) que incluye todos los elementos invariantes de la parte inferior de la página (aviso de copyright, política de privacidad, términos de uso, etc.).
- lateral (sidebar) se emplea para agrupar los elementos de las columnas laterales y secundarias de la página.

Código HTML	Esquema gráfico estructura web
<pre><div id="contenedor"> <div id="cabecera"> ... </div> <div id="contenido"> <div id="menu"> .. </div> ... </div> <div id="pie"> ... </div> </div></pre>	 <p>El diagrama ilustra la estructura visual de los divs descritos en el código HTML. Se muestra un contenedor principal que alberga tres secciones principales: la cabecera, el contenido y el pie. El contenido, a su vez, contiene un submenú. Las relaciones de contenedor se representan mediante rectángulos anidados.</p>

La etiqueta `<div>` es un elemento de bloque utilizado para contener y colocar otros elementos para crear un contenedor. Puede utilizarse también para formar cajas conjuntamente con estilos CSS.

La etiqueta `` es un elemento en línea que se utiliza para representar texto utilizando una hoja de estilos. Normalmente se utiliza para cambiar el estilo de un elemento o un texto dentro de una frase o un bloque.

La etiqueta `<div>` provoca un salto de línea, la etiqueta `` no. Tanto la etiqueta `<div>` como la etiqueta `` se utilizan con CSS.

3. CSS

CSS es un lenguaje de hojas de estilos creado para controlar la presentación de los documentos electrónicos definidos con HTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para la creación de páginas web complejas.

La separación de los contenidos y su presentación presentan numerosas ventajas, ya que obliga a crear documentos HTML bien definidos y con significado completo (también llamados "documentos semánticos"). Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Si el lenguaje HTML se utiliza para marcar los contenidos, es decir, para designar lo que es un párrafo, lo que es un titular o lo que es una lista de elementos, el lenguaje CSS se utiliza para definir el aspecto de todos los contenidos, es decir, el color, tamaño y tipo de letra de los párrafos de texto, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista, etc.

Cómo incluir CSS en un documento HTML

Una de las características principales de CSS es su flexibilidad y las diferentes opciones que ofrece para realizar una misma tarea. De hecho, existen tres opciones para aplicar CSS en un documento HTML.

1) Incluir CSS en el mismo documento HTML

```
<style type="text/css">
  p { color: black; font-family: Verdana; }
</style>
```

Este código lo incluimos dentro del HEAD entre las etiquetas <head> y </head>.

Este método se emplea cuando se define un número pequeño de estilos o cuando se quieren incluir estilos específicos en un determinado documento HTML que completen los estilos que se incluyen por defecto en todos los documentos del sitio web.

El principal inconveniente es que si se quiere hacer una modificación en los estilos definidos, es necesario modificar todas las páginas que incluyen el estilo que se va a modificar.

2) Incluir CSS en un archivo externo

```
<link rel="stylesheet" type="text/css" href="/css/estilos.css" media="screen" />
```

Este código lo incluimos dentro del HEAD entre las etiquetas <head> y </head>.

Un archivo de tipo CSS no es más que un archivo de texto normal y corriente cuya extensión es .css. Aunque generalmente se emplea la etiqueta <link> para enlazar archivos CSS externos, también se puede emplear la etiqueta <style>.

De las tres formas de aplicar CSS a los documentos HTML, esta es la más utilizada (sobre todo mediante la etiqueta <link>). La principal ventaja es que se puede incluir un mismo archivo CSS en multitud de páginas HTML, por lo que se garantiza la aplicación homogénea de los mismos estilos a todos los documentos que forman un sitio web.

La principal razón por la que es el método más utilizado es que es el más sencillo de mantener, ya que un solo cambio en un solo archivo CSS permite variar de forma instantánea los estilos de todos los documentos HTML que enlazan ese archivo.

3) Incluir CSS en los elementos

Esta forma nos permite incluir los CSS directamente en los elementos HTML. Solamente se utiliza en determinadas situaciones en las que se debe incluir un estilo muy específico para un solo elemento concreto.

Se aplica, mediante el atributo style, directamente sobre el elemento en el que queramos algún estilo en concreto. Vemos un ejemplo sobre el elemento párrafo.

```
<p style="color: black; font-family: Verdana;">Un párrafo de texto.</p>
```

Definición de una regla CSS

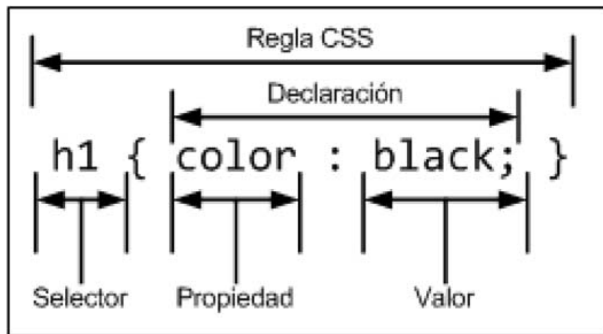
Un archivo CSS se crea simplemente guardando el archivo con la extensión .css.

Al contrario que el archivo HTML, no es necesario una estructura concreta de etiquetas para poder definir nuestras reglas CSS.

A continuación vemos un ejemplo de regla CSS que aplica el color negro a todos los elementos `<h1>` de la página HTML:

```
h1 { color: black; }
```

CSS define una serie de términos que permiten describir cada una de las partes que componen los estilos CSS. El siguiente esquema muestra las partes que forman un estilo CSS muy básico:



Los diferentes términos se definen a continuación:

- **Regla:** cada uno de los estilos que componen una hoja de estilos CSS. Cada regla está compuesta de una parte de "selectores", un símbolo de "llave de apertura" (`{`), otra parte denominada "declaraciones" y por último, un símbolo de "llave de cierre" (`}`).
- **Selector:** indica el elemento o elementos HTML a los que se aplica la regla CSS.
- **Declaración:** la declaración especifica los estilos que se aplicarán a los elementos. Está compuesta por una o más propiedades CSS.
- **Propiedad:** permite modificar el aspecto de una característica del elemento.
- **Valor:** indica el nuevo valor de la característica modificada en el elemento.

Un archivo CSS puede contener infinitas reglas CSS, cada regla puede contener infinitos selectores y cada declaración puede estar formada por un número infinito de pares propiedad/valor.

Agrupación de reglas

Una de las características más importantes de los estilos definidos con CSS es que se pueden agrupar las diferentes reglas que se aplican a un mismo selector.

Por lo tanto, las siguientes definiciones de reglas son exactamente iguales:

```
h1 {color: red;}  
h1 {font-size: 2em;}  
h1 {font-family: Verdana;}
```

Agrupamos las diferentes propiedades dentro del selector h1.

```
h1 {  
    color: red;  
    font-size: 2em;  
    font-family: Verdana;  
}
```

Y finalmente, podemos escribirlo todo en la misma línea.

```
h1 { color: red; font-size: 2em; font-family: Verdana; }
```

Comentarios en archivos CSS

Es importante que vayamos poniendo comentarios dentro de nuestros archivos CSS para que facilite la posterior lectura del archivo, para una mejor comprensión de la función y objetivo de la definición de una regla.

Así pues, es muy normal ir poniendo comentarios a modo de títulos a algunas reglas que todas ellas entre sí tienen la función de, por ejemplo, dar aspecto al menú.

Los comentarios en CSS se escriben:

```
/* Esto es un comentario */
```

3.1. Selectores

Para crear diseños web profesionales, es imprescindible conocer y dominar los selectores de CSS. Como se vio en el capítulo anterior, una regla de CSS está formada por una parte llamada "selector" y otra parte llamada "declaración".

La declaración se utiliza para decir "qué hay que hacer" y el selector es lo que dice "a quién hay que hacérselo". La declaración de una regla sencilla puede indicar que el color de la letra debe ser rojo, y el selector de esa regla sencilla puede indicar que los elementos a los que se aplica ese estilo son todos los párrafos de la página.

A un mismo elemento HTML se le pueden definir infinitas reglas CSS y cada regla puede tener un número infinito de selectores sobre los que se aplica.

Aunque CSS 2.1 define una docena de tipos de selectores, la mayoría de las páginas web se pueden definir utilizando solamente los 5 selectores básicos. Además, Internet Explorer, el navegador web que más utilizan los usuarios, no soporta los selectores avanzados, por lo que es casi obligatorio utilizar solamente los selectores básicos.

Selectores básicos

1) Selector universal

Se utiliza para seleccionar todos los elementos de la página. El siguiente ejemplo elimina el margen y el relleno de todos los elementos HTML (por ahora no es importante fijarse en la declaración de la regla CSS, solamente en los selectores):

```
* {  
  margin: 0;  
  padding: 0;  
}
```

El selector universal se indica mediante un asterisco (*). No se utiliza habitualmente, porque es difícil que un mismo estilo se pueda aplicar a todos los elementos de una página.

2) Selector de tipo o etiqueta

Selecciona todos los elementos de la página cuya etiqueta HTML coincide con el valor del selector. El siguiente ejemplo selecciona todos los párrafos de la página:

```
p {  
  ...  
}
```

Para utilizar este selector, solamente es necesario indicar el nombre de la etiqueta HTML correspondiente a los elementos que se quieren seleccionar. El siguiente ejemplo aplica diferentes estilos a los titulares y a los párrafos de una página HTML:

```
h1 {  
  color: red;  
}  
  
h1 {  
  color: blue;  
}
```

```
p {  
  color: black;  
}
```

CSS permite aplicar directamente los mismos estilos a varios selectores de forma simultánea.

Para ello, se indican todos los selectores diferentes separados por una coma (.). Así nos podemos ahorrar el hecho de volver a escribir la misma regla CSS a otro elemento HTML.

La siguiente regla CSS muestra cómo aplicamos un mismo estilo a tres selectores diferentes:

```
h1, h2, h3 {  
  color: #8A8E27;  
  font-weight: normal;  
  font-family: Arial, Helvetica, sans-serif;  
}
```

En las hojas de estilo complejas, es habitual agrupar las propiedades comunes de varios elementos en una única regla CSS y posteriormente definir las propiedades específicas de esos mismos elementos.

El siguiente ejemplo establece en primer lugar las propiedades comunes de los títulos de sección y a continuación, establece el tamaño de cada uno de ellos:

```
h1, h2, h3 {  
  color: #8A8E27;  
  font-weight: normal;  
  font-family: Arial, Helvetica, sans-serif;  
}  
h1 { font-size: 2em; }  
h2 { font-size: 1.5em; }  
h3 { font-size: 1.2em; }
```

3) Selector descendente

Permite seleccionar los elementos que se encuentran dentro de otros elementos. El siguiente ejemplo se emplea para visualizar en negrita el texto de cualquier elemento `` contenido dentro de un elemento `<p>`:

```
p span { font-weight: bold; }
```

Si el código HTML es el siguiente:

```
<p>
```

```
...  
<span>texto1</span>  
...  
<a href="">...<span>texto2</span></a>  
...  
</p>
```

Aplicando la regla CSS anterior, tanto texto1 como texto2 se verán en negrita. La razón es que con el selector descendente, un elemento no tiene que ser "hijo directo" de otro, sino que la única condición es que esté dentro de ese elemento, sin importar lo profundo que se encuentre.

Al resto de elementos `` de la página que no están dentro de un elemento `<p>`, no se les aplica la regla CSS anterior.

4) Selector de clase

Los selectores de clase son los selectores más utilizados junto con los selectores de ID (que se verán a continuación). Utilizando este selector, se pueden seleccionar todos los elementos de la página cuyo atributo `class` coincida con el selector.

Este tipo de selector es imprescindible para poder seleccionar elementos concretos de la página. ¿Cómo es posible seleccionar tres párrafos concretos de una página HTML en la que cada párrafo se encuentra en una zona diferente?

En este tipo de situaciones, lo que se hace es asignar un atributo `class` específico a los elementos que se quieren seleccionar y en la hoja de estilos se utiliza el selector de clase.

El selector está formado por un signo de punto (.) y el nombre del atributo `class` que se quiere seleccionar. Por tanto, en el siguiente ejemplo, solamente el segundo párrafo se mostrará de color rojo:

```
.especial { color: red; }
```

```
<p>Primer párrafo</p>  
<p class="especial">Segundo párrafo</p>  
<p>Tercer párrafo</p>
```

El selector `.especial` se traduce como "cualquier elemento cuyo atributo `class` sea igual a `especial`", por lo que solamente el segundo párrafo cumple la condición.

Combinando este selector con los anteriores, se puede restringir el alcance del selector. El siguiente ejemplo aplica la regla CSS solamente a los elementos de tipo `<p>` que tengan un atributo `class` igual al indicado:

```
p.aviso {  
padding: 0.5em;  
border: 1px solid #98be10;  
background: #f6feda;  
}
```

5) Selector de ID

Los selectores también pueden seleccionar los elementos HTML en función del valor de su atributo `id`. La explicación es la misma que para el atributo de clase. La sintaxis utilizada también es la misma, salvo que en este caso se utiliza el símbolo de la almohadilla (`#`) en lugar del símbolo del punto (`.`).

```
#especial { color: red; }
```

```
<p>Primer párrafo</p>  
<p id="especial">Segundo párrafo</p>  
<p>Tercer párrafo</p>
```

En el ejemplo anterior, solamente el segundo párrafo (cuyo atributo `id` es igual a `especial`) será seleccionado por el selector `#especial`.

La principal diferencia entre este selector y el selector de clase tiene que ver con HTML y no con CSS. Como se sabe, en una misma página, el valor del atributo `id` debe ser único, de forma que dos elementos diferentes no pueden tener el mismo valor de `id`. Sin embargo, el atributo `class` no es obligatorio que sea único, de forma que muchos elementos HTML diferentes pueden compartir el mismo valor para su atributo `class`.

Por tanto, cuando se quiere aplicar un estilo concreto a un solo elemento específico, se utiliza el selector de `id`. Si se quiere aplicar un estilo concreto a varios elementos diferentes, se utiliza el selector de clase.

Al igual que los selectores de clase, en este caso también se puede restringir el alcance del selector mediante la combinación con otros selectores. El siguiente ejemplo aplica la regla CSS solamente a los elementos de tipo `<p>` que tengan un atributo `id` igual al indicado:

```
p#aviso {  
padding: 0.5em;  
border: 1px solid #98be10;  
background: #f6feda;
```



```
}
```

3.2. Unidades de medida

Las unidades de medida en CSS se emplean para definir la altura y la anchura de las cajas que definen los elementos y para establecer el tamaño del texto mostrado.

CSS divide las unidades de medida en dos grupos: absolutas y relativas. Las medidas relativas definen su longitud en relación con otra medida. Las unidades absolutas establecen de forma completa el valor de una medida.

Unidades relativas:

Son más flexibles que las unidades absolutas porque se adaptan más fácilmente a los diferentes medios.

- em, relativa respecto del tamaño de letra del sistema del usuario
- ex, relativa respecto del tamaño de la letra x del sistema del usuario
- px, (píxel) relativa respecto de la pantalla del usuario

En el siguiente ejemplo, se indica que el tamaño normal del texto de la página debe ser el 90% del tamaño por defecto:

```
body { font-size: 0.9em; }
```

1.0em es el tamaño por defecto en el navegador del usuario. Y con 0.9em estamos reduciendo el tamaño de la letra en todo el elemento BODY un 10%. Esta es la forma más común de definir los tamaños del texto.

El funcionamiento de la unidad ex es idéntico a em, salvo que en este caso, la referencia es la altura de la letra x minúscula.

Las medidas indicadas en píxel son relativas a la resolución del dispositivo en el que se visualiza el documento HTML.

Los tamaños relativos también se pueden definir con porcentajes.

```
body { font-size: 90% }
```

Esto es exactamente igual que el ejemplo anterior.

En general, se recomienda el uso de unidades relativas siempre que sea posible, ya que mejora la accesibilidad de la página y permite que los documentos se adapten fácilmente a cualquier medio y dispositivo.

Habitualmente se utilizan píxeles y porcentajes para definir el layout del documento y em y porcentajes, para el tamaño de los textos.

3.3. Colores

Los colores en CSS se pueden indicar de 5 formas diferentes: palabras clave, colores del sistema, RGB hexadecimal, RGB numérico y RGB porcentual. Sólo veremos mediante palabras clave y mediante RGB hexadecimal, que son los más habituales.

Palabras clave

HTML define 17 palabras clave para referirse a los colores básicos:

aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, orange, purple, red, silver, teal, white, yellow

Ejemplo:

```
h1 { color: yellow; }
```

Además de la lista básica, los navegadores modernos soportan muchos otros nombres de colores. La lista completa se puede ver en <http://en.wikipedia.org/wiki/websage>.

RGB hexadecimal

Se trata del formato más utilizado (casi exclusivamente) y se basa en indicar los colores mediante sus valores RGB expresados en sistema hexadecimal. El resultado se indica mediante el símbolo # seguido de 2 caracteres por cada uno de los componentes RGB.

Ejemplo de conversión de los componentes RGB de un color a formato hexadecimal:

Componentes RGB originales: R = 71, G = 98, B = 176

Conversión a formato hexadecimal: R = 47, G = 62, B = B0

Para expresar el color escogido en formato RGB hexadecimal, se juntan los componentes RGB y se les añade el símbolo #, de la siguiente forma: #4762B0

```
p { color: #4762B0; }
```

3.4. Box model

La box model es la característica más importante del lenguaje de hojas de estilos CSS, ya que condiciona el diseño de todas las páginas web y de todos los documentos HTML. Todos los elementos que forman un documento HTML se representan mediante cajas rectangulares, cuyas propiedades define CSS y cuya representación visual controla también CSS. Cuando decimos todos los elementos, nos referimos a cualquier elemento HTML que hemos visto, como `<h1>`, `<h2>`, `<h3>`, ... `<p>`, ``... Cualquiera de ellos se define como una caja, con un **anchura y altura**, y después un **padding**, un **border** y un **margin**.

El diseño de cualquier página web está compuesto por cajas rectangulares. CSS permite definir la altura y anchura de cada caja, el margen que se dejará entre cada caja y el espacio de relleno interior que mostrará cada caja.

Además, CSS permite controlar la forma en la que se visualizan las cajas: se pueden ocultar, desplazar respecto de su posición original, fijarlas en una posición concreta dentro del documento, etc.



Los elementos que componen cada caja y su orden de visualización desde el punto de vista del usuario son los siguientes:

- **Contenido (content):** se trata del componente principal del elemento y puede estar formado por las palabras de un párrafo, una imagen, el texto de una lista de elementos, titulares, etc.
- **Relleno (padding):** está formado por el espacio libre (opcional) entre el contenido y el borde que lo encierra (opcionalmente).
- **Borde (border):** línea que encierra completamente el contenido y su relleno.

- **Imagen de fondo (background image):** imagen que se muestra por debajo del contenido. Si se define un color y una imagen de fondo, la imagen tiene más prioridad y es la que se visualiza.
- **Color de fondo (background color):** color que rellena el espacio ocupado por el contenido y su posible relleno. Si se define un color y una imagen de fondo, el color tiene menos prioridad y por tanto se visualiza la imagen.
- **Margen (margin):** espacio libre entre la caja y las posibles cajas adyacentes.

El relleno y el margen son transparentes, por lo que en el espacio ocupado por el relleno se muestra el color o imagen de fondo (si están definidos) y en el espacio ocupado por el margen se muestra el color o imagen de fondo de su elemento padre (si están definidos). Si ningún elemento padre tiene definido un color o imagen de fondo, se mostraría el color o imagen de fondo de la propia página (si están definidos).

Vamos a ver a continuación las propiedades de cada una de estas partes de la *box model*.

3.4.1. Propiedades anchura y altura

Width	Anchura. Establece la anchura de un elemento, lo que sería el contenido propiamente (content).
Height	Altura. Establece la altura de un elemento, lo que sería el contenido propiamente (content).

Ejemplo

Código CSS	Código HTML
<pre>#cabecera { height: 60px; width: 200px; }</pre>	<pre><div id="cabecera"> ... </div></pre>

3.4.2. Propiedad margin (margen)

Margin-top Margin-right Margin-bottom Margin-left	Margen superior, margen derecho, margen inferior, Margen izquierdo. Establece cada uno de los márgenes horizontales y verticales de un elemento.
--	---

Además de las cuatro propiedades que controlan los márgenes del elemento, CSS define una propiedad que permite definir los cuatro márgenes de forma directa empleando una única propiedad. Este tipo de propiedades resumidas se denominan propiedades de tipo *shorthand* y CSS define varias propiedades de este tipo, como se verá más adelante.

La propiedad que permite definir de forma simultánea los cuatro márgenes se denomina **margin**.

La propiedad **margin** admite entre 1 y 4 valores, con el siguiente significado:

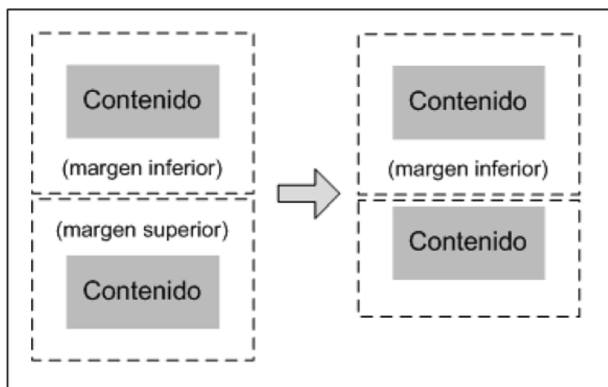
- Si solo se indica un valor, todos los márgenes tienen ese valor.
- Si se indican 2 valores, el primero se asigna al margen superior e inferior y el segundo se asigna a los márgenes izquierdo y derecho.
- Si se indican 3 valores, el primero se asigna al margen superior, el tercero se asigna al margen inferior y el segundo valor se asigna a los márgenes izquierdo y derecho.
- Si se indican 4 valores, el orden de asignación es: margen superior, margen derecho, margen inferior y margen izquierdo.

Código CSS	Código CSS
<pre>div img { margin-top: .5em; margin-bottom: .5em; margin-left: 1em; margin-right: .5em; }</pre>	<pre>div img { margin: .5em .5em .5em 1em; }</pre>

Estos dos ejemplos son exactamente iguales. Esta regla se aplicaría a cualquier imagen que esté dentro de un div.

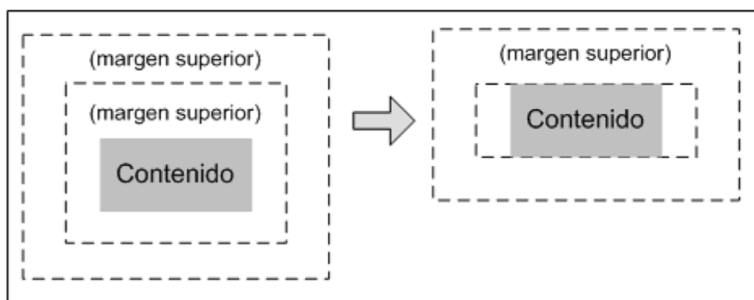
El comportamiento de los márgenes verticales es más complejo de lo que se puede imaginar. Cuando se juntan dos o más márgenes verticales, se fusionan de forma automática y la altura del nuevo margen será igual a la altura del margen más alto de los que se han fusionado.

Fusión automática de los márgenes verticales



De la misma forma, si un elemento está contenido dentro de otro elemento, sus márgenes verticales se fusionan y resultan en un nuevo margen de la misma altura que el mayor margen de los que se han fusionado:

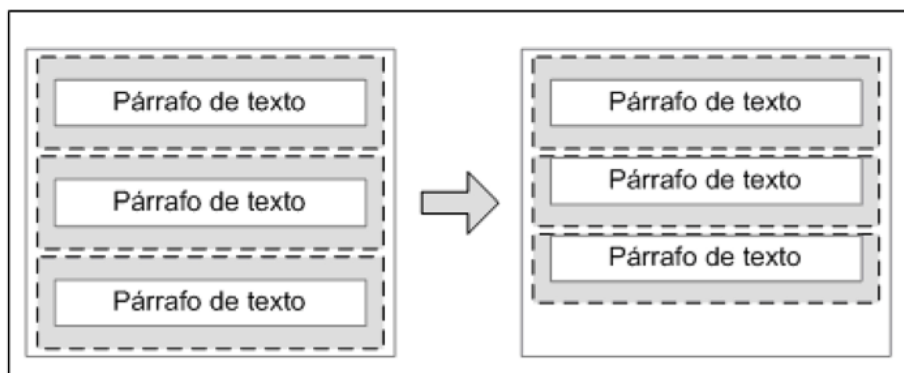
Fusión de los márgenes de los elementos interiores



Aunque en principio puede parecer un comportamiento extraño, la razón por la que se propuso este mecanismo de fusión automática de márgenes verticales es la de dar uniformidad a las páginas más habituales.

En una página con varios párrafos, si no se diera este comportamiento y se estableciera un determinado margen a todos los párrafos, el primer párrafo no mostraría un aspecto homogéneo respecto de los demás.

Motivo por el cual se fusionan automáticamente los márgenes verticales



3.4.3. Propiedad padding (relleno)

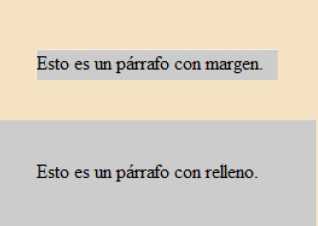
CSS define 4 propiedades para controlar cada uno de los espacios de relleno horizontales y verticales de un elemento.

Padding-top Padding-right Padding-bottom Padding-left	Relleno superior, relleno derecho, relleno inferior, relleno izquierdo. Establece cada uno de los rellenos horizontales y verticales de un elemento.
--	--

Como sucede con la propiedad **margin**, CSS también define una propiedad de tipo shorthand para establecer los 4 rellenos de un elemento de forma directa.

La propiedad que permite definir de forma simultánea los 4 márgenes se denomina **padding**.

Ejemplo: probamos este código para ver las diferencias entre **margin** y **padding**:

Código CSS	Vista navegador
<pre> .margen { margin: 2em; background-color:#CCCCCC; width: 200px; height: 25px; } .relleno { padding: 2em; background-color:#CCCCCC; width: 200px; height: 25px; } </pre>	
Código HTML	
<pre> <p class="margen">Esto es un párrafo con margen.</p> <p class="relleno">Esto es un párrafo con relleno.</p> </pre>	

En este ejemplo ya vemos una nueva propiedad: background-color... Si nos fijamos de nuevo en la imagen inicial de BOX MODEL, nos damos cuenta de que la propiedad background-color abarca hasta la propiedad **padding**, pero no pinta la zona definida por **margin**.

3.4.4. Propiedad border (bordes)

CSS permite definir el aspecto de cada uno de los 4 bordes horizontales y verticales de los elementos. Para cada borde se puede establecer su anchura, su color y su estilo.

Para definir la anchura del borde:

Border-top-width Border-right-width Border-bottom-width Border-left-width	Anchura del borde superior, anchura del borde derecho, anchura del borde inferior, anchura del borde izquierdo. Establece la anchura de cada uno de los cuatro bordes de los elementos.
--	--

El valor de las propiedades que controlan la anchura de los bordes puede estar formado por una medida (absoluta o relativa y en cualquier unidad de medida de las definidas) o por una palabra clave (seleccionada entre las palabras thin, medium y thick).

Normalmente se utilizan medidas expresadas en píxel o en em, ya que las palabras clave definidas no son muy comunes.

La propiedad que permite definir de forma simultánea la anchura de los 4 borders se denomina **border-width**.

Para definir el color del borde:

Border-top-color Border-right-color Border-bottom-color Border-left-color	Color del borde superior, color del borde derecho, color del borde inferior, color del borde izquierdo. Establece el color de cada uno de los cuatro bordes de los elementos.
--	--

La propiedad que permite definir de forma simultánea el color de los 4 borders se denomina **border-color**.

Para definir el estilo del borde:

Border-top-style Border-right-style Border-bottom-style Border-left-style	Estilo del borde superior, estilo del borde derecho, estilo del borde inferior, estilo del borde izquierdo. Establece el estilo de cada uno de los cuatro bordes de los elementos.
--	---

La propiedad que permite definir de forma simultánea el estilo de los 4 borders se denomina **border-style**.

Los estilos de los bordes pueden ser: none, hidden, dotted, dashed, solid, double, groove, ridge, inset, outset.

Los bordes más utilizados en los diseños habituales son solid y dashed, seguidos de double y dotted.

Ejemplo: probamos este código para ver las diferentes propiedades del border:

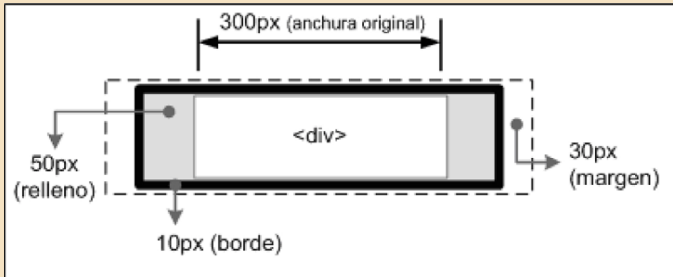
Código CSS	Vista navegador
<pre>div { padding: 2em; background-color: #CCCCCC; width: 200px; height: 25px; border-width: 5px; border-color: #FF0000; border-style: dotted; }</pre>	
Código HTML	
<pre><div>Ejemplo de bordes</div></pre>	

También podemos agrupar las diferentes propiedades CSS del borde en una sola línea. De esta forma podemos definir el mismo borde del ejemplo anterior de la siguiente manera:

```
border: 5px dotted #FF0000;
```

3.4.5. Tamaño final de un elemento

Todas las propiedades que definen el BOX MODEL de un elemento HTML nos dirán finalmente el tamaño de este elemento. Es decir, el margen, el relleno y los bordes establecidos a un elemento determinan la anchura (y altura) final del elemento. En el siguiente ejemplo se muestran los estilos CSS de un elemento:

Código CSS
<pre>div { width: 300px; padding-left: 50px; padding-right: 50px; margin-left: 30px; margin-right: 30px; border: 10px solid black; }</pre>
Esquema gráfico estructura web


De esta forma, la anchura del elemento en pantalla sería igual a la suma de la anchura original, los márgenes, los bordes y los rellenos:

30px + 10px + 50px + 300px + 50px + 10px + 30px = 480 píxeles

Así, la anchura (y altura) establecida con CSS siempre hace referencia a la anchura del contenido. La anchura (y altura) total del elemento debe tener en cuenta además los valores del resto de elementos del box model.

3.4.6. Fondos

El último elemento que forma el box model es el fondo de la caja del elemento. El fondo puede ser un color simple o una imagen. El fondo solamente se visualiza en el área ocupada por el contenido y su relleno, ya que el color de los bordes se controla directamente desde los bordes y las zonas de los márgenes siempre son transparentes.

Para establecer un color o imagen de fondo en la página entera, se debe establecer un fondo al elemento <body>. Si se establece un fondo a la página, como el valor inicial del fondo de los elementos es transparente, todos los elementos visualizan el mismo fondo que la página a menos que esos elementos especifiquen su propio fondo.

CSS define 5 propiedades para establecer el fondo de cada elemento (background-color, background-image, background-repeat, background-attachment, background-position) y otra propiedad de tipo shorthand (background).

Para definir el color del fondo:

Background-color	Color de fondo. Establece un color de fondo para los elementos.
------------------	--

```
body {background-color: #F5F5F5;}
```

Para definir una imagen de fondo:

Background-image	Imagen de fondo. Establece una imagen como fondo para los elementos.
------------------	---

El siguiente ejemplo muestra una imagen como fondo de toda la página:

```
body { background-image: url("imagenes/fondo.gif") }
```

Las imágenes de fondo se indican a través de su URL, que puede ser absoluta o relativa. Suele ser recomendable crear una carpeta de imágenes que se encuentre en el mismo nivel que los archivos CSS y que almacene todas las imágenes utilizadas en el diseño de la página.

Así, las imágenes correspondientes al diseño de la página se mantienen separadas del resto de imágenes del sitio y el código CSS es más sencillo (por utilizar URL relativas) y más fácil de mantener (por no tener que actualizar URL absolutas en caso de que se cambie la estructura del sitio web).

Por otra parte, suele ser habitual indicar un color de fondo siempre que se muestra una imagen de fondo. En caso de que la imagen no se pueda mostrar o contenga errores, el navegador mostrará el color indicado (que debería ser, en lo posible, similar a la imagen) y la página no parecerá que contiene errores.

Con una imagen muy pequeña (y que por tanto, tarda muy poco en ser enviada hasta el ordenador del usuario) se consigue cubrir completamente el fondo de la página, con lo que se consigue un gran ahorro de ancho de banda.

En ocasiones, no es conveniente que la imagen de fondo se repita horizontal y verticalmente.

Para ello, CSS introduce la propiedad `background-repeat`, que permite controlar la forma de repetición de las imágenes de fondo.

Para definir la repetición de la imagen de fondo:

Background-repeat	Repetición de la imagen de fondo. Controla la forma en la que se repiten las imágenes de fondo.
-------------------	--

Los valores de esta propiedad pueden ser: `repeat`, `repeat-x`, `repeat-y`, `no-repeat`.

El valor `repeat` indica que la imagen se debe repetir en todas direcciones y por tanto, es el comportamiento por defecto. El valor `no-repeat` muestra una sola vez la imagen y no se repite en ninguna dirección.

El valor `repeat-x` repite la imagen solo horizontalmente y el valor `repeat-y` repite la imagen solamente de forma vertical.

Para definir la posición de la imagen de fondo:

Además de seleccionar el tipo de repetición de las imágenes de fondo, CSS permite controlar la posición de la imagen dentro del fondo del elemento mediante la propiedad `background-position`.

Background-position	Posición de la imagen de fondo. Controla la posición en la que se muestra la imagen en el fondo del elemento.
---------------------	--

La propiedad `background-position` permite indicar a qué distancia se desplaza la imagen de fondo respecto de su posición original situada en la esquina superior izquierda.

Si se indican 2 porcentajes o 2 medidas, el primero indica el desplazamiento horizontal y el segundo el desplazamiento vertical respecto del origen (situado en la esquina superior izquierda). Si solamente se indica un porcentaje o una medida, se considera que es el desplazamiento horizontal y al desplazamiento vertical se le asigna automáticamente el valor de 50%.

Palabras clave permitidas para este parámetro: top, left, center, right, bottom.

Las palabras clave permitidas son equivalentes a algunos porcentajes significativos: top = 0%, left = 0%, center = 50%, bottom = 100%, right = 100%.

CSS permite mezclar porcentajes y palabras clave, como por ejemplo 50% 2cm, center 2cm, center 10%.

Si se utilizan solamente palabras clave, el orden es indiferente y por tanto, es equivalente indicar top left y left top.

Ejemplo: probamos este código para ver las diferentes propiedades del background-position:

Código CSS	Vista navegador
<pre>#cajal { background-image: url(images/help.png); background-repeat: no-repeat; background-position: bottom left; height: 200px; width: 400px; border: 1px solid #333333; }</pre>	
Código HTML	
<pre><div id="cajal"><h1>bottom left</h1></div></pre>	

Propiedad background

Por último, CSS define una propiedad de tipo shorthand para indicar todas las propiedades de los colores e imágenes de fondo de forma directa. La propiedad se denomina background y es la que generalmente se utiliza para establecer las propiedades del fondo de los elementos.

El orden en el que se indican las diferentes propiedades es indiferente, aunque en general se sigue el formato indicado de color, url de imagen, repetición y posición.

El siguiente ejemplo muestra la ventaja de utilizar la propiedad background:

```
background: #222d2d url(/graphics/colorstrip.gif) repeat-x 0 0;
```

es equivalente a:

```
background-color: #222d2d;  
background-image: url(/graphics/colorstrip.gif);  
background-repeat: repeat-x;  
background-position: 0 0;
```

3.5. Tipografía

CSS define numerosas propiedades para establecer la apariencia del texto y de la letra utilizada en el texto. A pesar de que no dispone de tantas posibilidades como los lenguajes y programas específicos de publicación impresa, CSS permite aplicar estilos complejos y muy variados al texto.

Estas son las propiedades más importantes para definir la tipografía de nuestro texto en CSS:

Font-family	Establece el tipo de letra utilizado para el texto.
Font-size	Establece el tamaño de letra utilizado para el texto.
Font-weight	Establece la anchura de la letra utilizada para el texto.
Font-style	Establece el estilo de la letra utilizada para el texto.

La característica básica para controlar el aspecto del texto es seleccionar el tipo de letra que se va a mostrar. Para ello, CSS define la propiedad ***font-family***.

Los valores permitidos son los siguientes:

- `nombre_familia`: nombre de un tipo de letra, como por ejemplo "Arial", "Verdana", "Garamond", etc.
- `familia_generica`: nombre de un tipo de letra genérico. Los nombres definidos son serif (tipo Times New Roman), sans-serif (tipo Arial), cursive (tipo Comic Sans), fantasy (tipo Impact) y monospace (tipo Courier New).

El tipo de letra indicado debe encontrarse instalado en el sistema operativo del usuario. Por tanto, el valor de `font-family` suele definirse como una lista de tipos de letra alternativos separados por comas. Si el navegador no encuentra un tipo de letra, pasa al siguiente hasta que el tipo de letra indicado se encuentre instalado en el sistema operativo del usuario.

De esta forma, se suele indicar una serie de 2 o 3 tipos de letra seguidos de un nombre de familia genérica que el navegador utilizará en caso de que no encuentre ninguno de los tipos de letra anteriores.

Las series más utilizadas son las siguientes:

font-family: Arial, Helvetica, sans-serif;

font-family: "Times New Roman", Times, serif;

font-family: "Courier New", Courier, monospace;

font-family: Georgia, "Times New Roman", Times, serif;

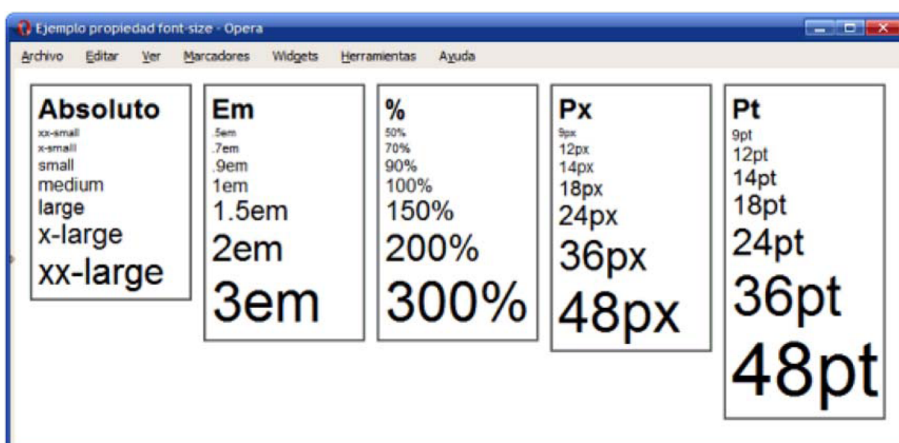
font-family: Verdana, Arial, Helvetica, sans-serif;

Una vez seleccionado el tipo de letra, se puede indicar el tamaño de letra mediante la propiedad **font-size**.

Además de todas las unidades de medida relativas y absolutas y el uso de porcentajes, CSS define una serie de palabras clave que permite establecer el tamaño de forma absoluta o relativa:

- **tamaño_absoluto**: se trata de un valor seleccionado entre las palabras clave definidas: `xx-small` | `x-small` | `small` | `medium` | `large` | `x-large` | `xx-large`.
- **tamaño_relativo**: también consiste en un valor seleccionado entre dos palabras clave definidas (*larger* | *smaller*) y que toman como referencia el tamaño de letra del elemento padre.

La siguiente imagen muestra una comparación entre los tamaños típicos del texto y las unidades que más se utilizan:



CSS recomienda indicar el tamaño del texto en la unidad `em` o en porcentaje (%). Además, es habitual indicar el tamaño del texto en puntos (`pt`) cuando el documento está específicamente diseñado para ser impreso.

Por defecto los navegadores asignan los siguientes tamaños a los títulos de sección: h1 =xx-large, h2 = x-large, h3 = large, h4 = medium, h5 = small, h6 = xx-small.

Una vez indicado el tipo y el tamaño de letra, es habitual modificar su anchura (texto en negrita) y su estilo (cursiva, mayúsculas pequeñas, etc.). La propiedad que controla la anchura de la letra es **font-weight**.

Valores de **font-weight**: normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | inherit

Los valores que normalmente se utilizan son normal (el valor por defecto) y bold para los textos en negrita. El valor normal equivale al valor numérico 400 y el valor bold al valor numérico 700.

Además de la anchura de la letra, CSS permite variar el estilo de la letra mediante la propiedad **font-style**.

Valores **font-style**: normal | italic | oblique | inherit.

Normalmente se emplea la propiedad **font-style** para mostrar un texto en cursiva mediante el valor italic.

CSS proporciona una propiedad tipo shorthand y denominada font que permite indicar de forma directa todas las propiedades de la tipografía de un texto.

El orden en el que se deben indicar las propiedades del texto es el siguiente:

- En primer lugar y de forma opcional se indican el font-style y font-weight en cualquier orden.
- A continuación, se indica el valor de font-size seguido opcionalmente por el valor de line-height.
- Por último, se indica el tipo de letra a utilizar.

Ejemplos de uso de la propiedad font:

```
font: bold 1em "Trebuchet MS", Arial, Sans-Serif;  
font: 11px verdana, sans-serif;  
font: bold 14px georgia, times, serif;
```

3.6. Texto

Además de las propiedades relativas a la tipografía del texto, CSS define numerosas propiedades que determinan la apariencia del texto en su conjunto. Estas propiedades adicionales permiten controlar la alineación del texto, el interlineado, la separación entre palabras, etc.

Estas son las propiedades más importantes para definir el texto en su conjunto CSS:

Text-align	Establece la alineación del contenido del elemento.
Line-height	Permite establecer la altura de línea de los elementos.
Text-decoration	Establece la decoración del texto (subrayado, tachado, parpadeante, etc.).
Text-transform	Transforma el texto original (lo transforma a mayúsculas, a minúsculas, etc.).
Vertical-align	Determina la alineación vertical de los contenidos de un elemento.

Text-align

La propiedad que define la alineación del texto se denomina text-align.

Valores: left | right | center | justify | inherit

Los valores definidos por CSS permiten alinear el texto según los valores tradicionales: a la izquierda, a la derecha, centrado y justificado.

La propiedad text-align no solo alinea el texto que contiene un elemento, sino todos sus contenidos, por lo que se puede emplear para centrar imágenes, alinear elementos en el lado derecho, etc.

El interlineado de un texto se controla mediante la propiedad line-height, que permite controlar la altura de cada línea de texto.

Line-height

Además de todas las unidades de medida y el uso de porcentajes, la propiedad **line-height** permite indicar un número sin unidades que se interpreta como el múltiplo del tamaño normal de la letra. Por tanto, estas tres reglas CSS son equivalentes:

```
p { line-height: 1.2; font-size: 1em }  
p { line-height: 1.2em; font-size: 1em }  
p { line-height: 120%; font-size: 1em }
```


El interlineado mejora notablemente la legibilidad de un texto siempre que se aplique un valor moderado.

Text-decoration

Además de la decoración que se puede aplicar a la tipografía que utilizan los textos, CSS define estilos y decoraciones adicionales para el texto. La propiedad que decora el texto se denomina **text-decoration**.

Valores: none | underline | overline | line-through | blink | inherit

Se trata de una propiedad que debe utilizarse de forma ocasional, ya que su uso puede interferir en el diseño general del documento. El valor **underline** subraya el texto y puede confundir a los usuarios haciéndoles creer que se trata de un enlace.

El valor **overline** añade una línea en la parte superior del texto, un aspecto que raramente es deseable. El valor **line-through** muestra el texto con una línea tachándolo, por lo que su uso tampoco es muy habitual. Por último, el valor **blink** muestra el texto parpadeante y se recomienda evitar su uso por las molestias que genera a la mayoría de usuarios.

Text-transform

Una de las propiedades de CSS más desconocidas y que puede ser de gran utilidad en algunas circunstancias es la propiedad **text-transform**, que puede variar de forma sustancial el aspecto del texto.

Valores: capitalize | uppercase | lowercase | none | inherit

La propiedad **text-transform** permite mostrar el texto original transformado en un texto completamente en mayúsculas, en minúsculas o con la primera letra de cada palabra en mayúscula.

Vertical-align

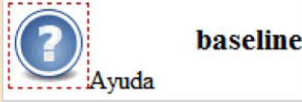
Uno de los principales problemas con el diseño de documentos y páginas mediante CSS se encuentra en la alineación vertical de varios elementos diferentes como imágenes y texto en una misma línea.

CSS define la propiedad **vertical-align** para controlar el mecanismo que alinea verticalmente los diferentes elementos de una misma línea.

Valores: baseline | sub | super | top | text-top | middle | bottom | text-bottom |

<porcentaje> | <medida> | inherit

Ejemplo:

Código CSS	Vista navegador
<pre>img { border:1px dashed #CC0000; padding: 1px; } div#baseline img { vertical-align:baseline; }</pre>	
Código HTML	
<pre><div id="baseline"> <h1>baseline</h1> Ayuda </div></pre>	

3.7. Enlaces

Tamaño, color y decoración

Los estilos más sencillos que se pueden aplicar a los enlaces son los que modifican su tamaño de letra, su color y la decoración del texto del enlace.

La siguiente imagen muestra algunos enlaces con diferentes estilos en una misma página:

Ejemplo:

Código CSS

```
a {  
    margin: 1em 0;  
    display: block;  
}  
.alternativo {color: #CC0000;}  
.simple {text-decoration: none;}  
.importante {  
    font-weight: bold;  
    font-size: 1.3em;  
}  
.raro {text-decoration:overline;}
```

Código HTML

```
<a href="#">Enlace con el estilo por defecto</a>  
<a class="alternativo" href="#">Enlace de color rojo</a>  
<a class="simple" href="#">Enlace sin subrayado</a>  
<a class="importante" href="#">Enlace muy importante</a>  
<a class="raro" href="#">Enlace con un estilo raro</a>
```

Vista navegador**Pseudo-clases**

CSS define cuatro pseudo-clases que permiten aplicar estilos avanzados para los enlaces de los documentos. Cada pseudo-clase permite diferenciar el estilo de un enlace según su tipo o estado: enlace no visitado, enlace visitado, enlace en el que se pasa el puntero del ratón por encima y enlace activo en ese momento.

Cada una de las pseudo-clases definidas se muestra a continuación:

- **:link**, permite aplicar estilos para los enlaces que aun no han sido pinchados.
- **:visited**, aplica estilos a los enlaces que han sido pinchados anteriormente (el navegador del usuario elimina automáticamente el historial de enlaces visitados cada cierto tiempo).
- **:hover**, estilos que muestra el enlace cuando el usuario posiciona el puntero del ratón sobre el enlace.

- **:active**, estilos que se aplican al enlace cuando el usuario está pinchando sobre el enlace (el tiempo durante el que se aplican estos estilos es muy breve).

El orden recomendado para la definición de las pseudo-clases de los enlaces es `:link`, `:visited`, `:hover`, `:active`.

Las pseudo-clases de los enlaces permiten variar el comportamiento por defecto que los navegadores aplican a los enlaces. El siguiente ejemplo oculta el subrayado a los enlaces cuando el usuario pasa el ratón por encima de un enlace:

```
a {  
  }  
a:hover {  
  text-decoration: none;  
}
```

El subrayado creado con la propiedad **text-decoration** no permite controlar el aspecto del subrayado. El color y la anchura del subrayado se definen de forma automática por el navegador.

La única posibilidad de crear decoraciones personalizadas para los enlaces es emplear la propiedad **border**. El siguiente ejemplo muestra algunos enlaces con el subrayado personalizado:

Ejemplo:

Código CSS

```
a {  
  margin: 1em 0;  
  float: left;  
  clear: left;  
  text-decoration: none;  
}  
.simple {text-decoration: underline;}  
.color { border-bottom: medium solid #CC0000;}  
.ancho {border-bottom: thick solid;}  
.separado {border-bottom: 1px solid; padding-bottom: .6em;}  
.discontinuo {border-bottom: thin dashed;}
```

Código HTML

```
<a class="simple" href="#">Enlace con el estilo por defecto</a>  
<a class="color" href="#">Enlace un subrayado de otro color</a>  
<a class="ancho" href="#">Enlace con un subrayado muy ancho</a>  
<a class="separado" href="#">Enlace con un subrayado muy separado</a>  
<a class="discontinuo" href="#">Enlace con un subrayado discontinuo</a>
```

La propiedad **background** (y **background-image**) permite personalizar los enlaces según su tipo y mostrar una imagen al lado de cada enlace. La técnica consiste en mostrar una imagen de fondo sin repetición y añadir el **padding** necesario al texto del enlace para que no se solape con la imagen de fondo.

Ejemplo:

Código CSS
<pre>a { margin: 1em 0; float: left; clear: left; } .pdf { padding: 0 0 0 22px; background: #FFF url(images/pdf.png) no-repeat left center; }</pre>
Código HTML
<pre>Enlace a un documento PDF</pre>
Vista navegador


Imágenes

Eliminar el borde de las imágenes con enlaces

Cuando una imagen forma parte de un enlace, los navegadores por defecto muestran las imágenes con un borde azul ancho. Por tanto, la regla CSS más habitual para modificar los estilos CSS por defecto suele ser la de eliminar los bordes de las imágenes con enlaces:

```
img { border: none; }
```

3.8. Listas

CSS define varias propiedades para controlar el tipo de viñeta que muestran las listas, la posición de la viñeta, etc.

Estas son las propiedades más importantes para definir el texto en su conjunto CSS:

List-style-type	Permite establecer el tipo de viñeta mostrada para una lista.
List-style-position	Permite establecer la posición de la viñeta de cada elemento de una lista.
List-style-image	Permite reemplazar las viñetas automáticas por una imagen personalizada.

List-style-type

La propiedad básica es la que controla el tipo de viñeta que se muestra y que se denomina **list-style-type**.

Valores: `disc` | `circle` | `square` | `decimal` | `decimal-leading-zero` | `lower-roman` | `upper-roman` | `lower-greek` | `lower-latin` | `upper-latin` | `armenian` | `georgian` | `lower-alpha` | `upper-alpha` | `none` | `inherit`

En primer lugar, el valor **none** permite mostrar una lista en la que sus elementos no contienen viñetas, números o letras. Se trata de un valor muy utilizado, ya que es imprescindible para los menús de navegación creados con listas, como se verá más adelante.

El resto de valores de la propiedad `list-style-type` se dividen en 3 tipos: gráficos, numéricos y alfabéticos.

Los valores gráficos son **disc**, **circle** y **square** y muestran como viñeta un círculo relleno, un círculo vacío y un cuadrado relleno respectivamente.

Los valores numéricos están formados por **decimal**, **decimal-leading-zero**, **lower-roman**, **upper-roman**, **armenian** y **georgian**.

Por último, los valores alfanuméricos se controlan mediante **lower-latin**, **lower-alpha**, **upper-latin**, **upper-alpha** y **lower-greek**.

List-style-position

La colocación de las viñetas se controla mediante la propiedad **list-style-position**.

Valores: `inside` | `outside` | `inherit`

La diferencia entre los valores **outside** y **inside** se hace evidente cuando los elementos contienen mucho texto, como en la siguiente imagen:



List-style-image

Cuando se requiere personalizar el aspecto de las viñetas, se debe emplear la propiedad **list-style-image**, que permite mostrar una imagen personalizada en vez de una viñeta automática.

Las imágenes personalizadas se establecen mediante la URL de la imagen. Si no se encuentra la imagen o no se puede cargar, se muestra la viñeta automática correspondiente (salvo que explícitamente se haya eliminado mediante la propiedad **list-style-type**).

La siguiente imagen muestra el uso de la propiedad **list-style-image** mediante tres ejemplos sencillos de listas con viñetas personalizadas:

Código CSS	Vista navegador
<pre> ul { margin:0; padding-left: 1.5em; line-height: 1.5em; } ul li { padding-left: .2em; } ul.ok { list-style-image: url(imagenes/ok.png); } </pre>	
Código HTML	
<pre> <div> <ul class="ok"> Elemento Elemento Elemento Elemento </div> </pre>	

Como es habitual, CSS define una propiedad de tipo shorthand, que permite establecer todas las propiedades de una lista de forma directa. La propiedad se denomina **list-style**.

El orden en el que se indican las propiedades es indiferente, ya que las propiedades no tienen ningún valor común.

3.9. Posicionamiento de los elementos HTML

CSS define 3 mecanismos para posicionar cada caja formada por los elementos HTML:

- Normal: posicionamiento por defecto de los elementos en línea y los de bloque. Además, incluye el posicionamiento relativo.
- Float: posicionamiento que consiste en posicionar el elemento según el esquema normal y una vez colocado, desplazarlo todo lo posible hacia la izquierda o hacia la derecha.
- Absoluto: posicionamiento que consiste en extraer por completo el elemento de su posicionamiento normal y colocarlo en la posición indicada respecto de su elemento padre.

La propiedad de CSS que define el posicionamiento del elemento se denomina **position**.

Los valores de la propiedad **position** pueden ser:

El significado de cada uno de los posibles valores de **position** es el siguiente:

- **static**: es el posicionamiento que se utiliza por defecto y todos los elementos inicialmente se muestran de esta forma. No se tienen en cuenta los valores de las propiedades **top**, **right**, **bottom** y **left**.
- **relative**: la nueva posición del elemento se calcula a partir de la posición que tendría si no se utilizara la propiedad **position**. Las posiciones de los siguientes elementos no se ven afectadas por el desplazamiento de este elemento.
- **absolute**: la nueva posición del elemento se determina mediante las propiedades **top**, **right**, **bottom** y **left**. Los valores de esas propiedades indican la posición del elemento respecto de la posición de su elemento padre. La posición del siguiente elemento se calcula como si no existiera el elemento que se desplaza, ya que este último se desentiende por completo del posicionamiento normal.

- **fixed:** la nueva posición del elemento se calcula de forma idéntica al posicionamiento absoluto. La diferencia reside en que en el caso de **fixed**, el elemento no se mueve cuando se desplaza la ventana del navegador. En los medios visuales (como la pantalla) el elemento se muestra en una posición fija independiente del movimiento de la ventana del navegador. En los medios impresos el elemento se muestra en todas las páginas.

Las 4 propiedades relacionadas con la propiedad **position** son las que determinan el desplazamiento de los elementos respecto de sus posiciones originales. CSS define para ello las propiedades **top**, **right**, **bottom** y **left**.

En el caso del posicionamiento relativo, cada una de estas propiedades indica el desplazamiento del elemento desde su borde superior/derecho/inferior/izquierdo.

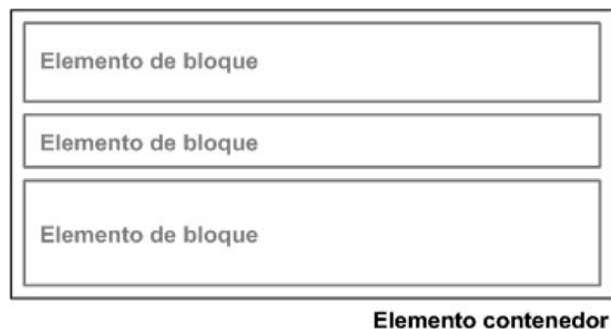
Si el posicionamiento es absoluto, las propiedades indican el desplazamiento del elemento respecto del borde superior/derecho/inferior/izquierdo de su elemento padre.

En cualquiera de los 2 casos, si el desplazamiento se indica en forma de porcentaje, se refiere al porcentaje sobre la anchura (propiedades **right** y **left**) o altura del elemento (propiedades **top** y **bottom**).

3.9.1. Posicionamiento normal

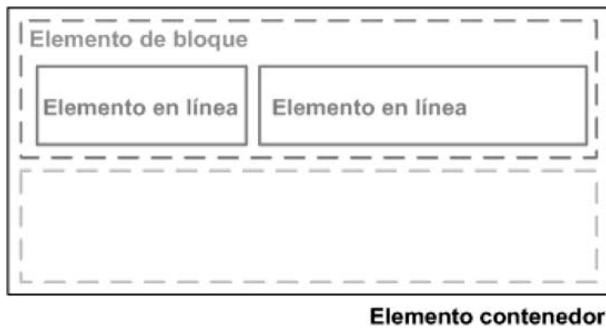
Los elementos de bloque forman lo que CSS denomina "contextos de formato de bloque". En este tipo de contextos, las cajas se muestran una debajo de otra comenzando desde el principio del elemento padre. La distancia de las cajas se controla mediante los márgenes verticales.

Posicionamiento normal de elementos de bloque



Los elementos en línea forman los "contextos de formato en línea". En este tipo de contextos, las cajas se muestran una detrás de otra de forma horizontal comenzando desde la posición más a la izquierda del elemento padre. La distancia entre las cajas se controla mediante los márgenes horizontales.

Posicionamiento normal de elementos de línea



Si las cajas en línea ocupan más espacio del disponible en su propia línea, el resto de cajas se muestran en las líneas siguientes. Si las cajas en línea ocupan un espacio menor que su propia línea, se puede controlar la distribución de las cajas mediante la propiedad **text-align** para centrarlas, alinearlas a la derecha o justificarlas.

El posicionamiento relativo también se considera parte del posicionamiento normal. En este caso, el desplazamiento de una caja no afecta al resto, que se muestran en la misma posición que si la caja desplazada no se hubiera movido de su sitio original.

Diferencias visuales entre el posicionamiento normal y el posicionamiento relativo



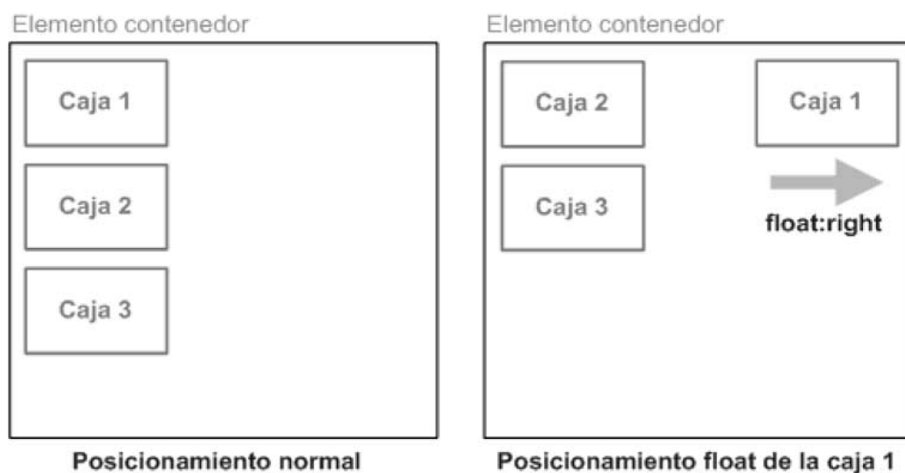
Si imaginamos que las cajas anteriores son imágenes (elementos en línea), el código CSS que se debería aplicar a la imagen Caja 2 del esquema anterior sería:

```
img.caja2 {  
  position: relative;  
  top: 4em;  
  left: 4em;  
}
```

El resto de imágenes no varían su posición y no ocupan el hueco dejado por la primera imagen, porque el posicionamiento relativo no influye en el resto de elementos de la página. La única incidencia que tiene el posicionamiento relativo sobre el resto de elementos es que el elemento desplazado puede solaparse con otros elementos de la página.

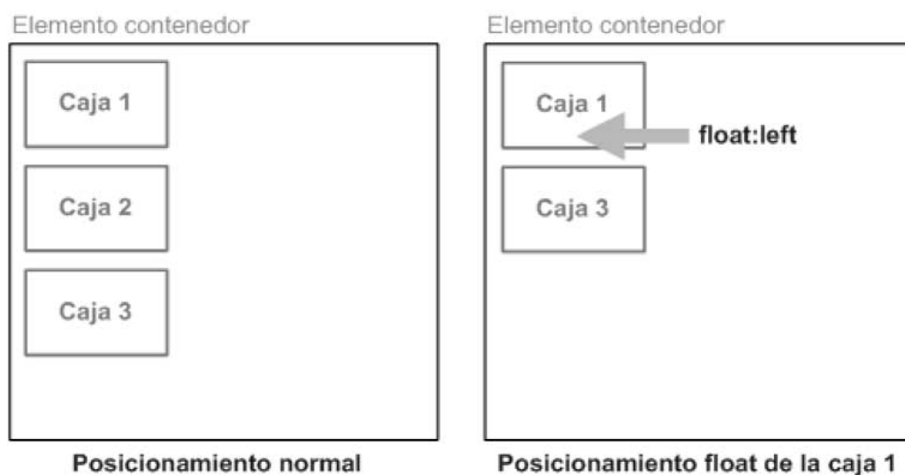
3.9.2. Posicionamiento float

Una caja posicionada mediante la propiedad **float**, se desplaza hasta la zona más a la izquierda o más a la derecha de la línea en la que se debería mostrar si no se desplazara.



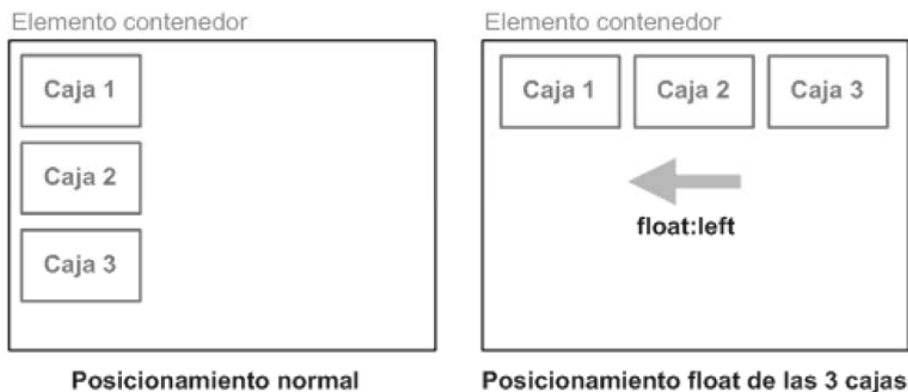
Una caja desplazada mediante float no pertenece al posicionamiento normal de un documento, por lo que los elementos de bloque anteriores y posteriores se visualizan como si la caja desplazada no existiera.

Si en el anterior ejemplo la "Caja 1" se posiciona mediante un `float: left` el resultado sería el que se muestra en la imagen:

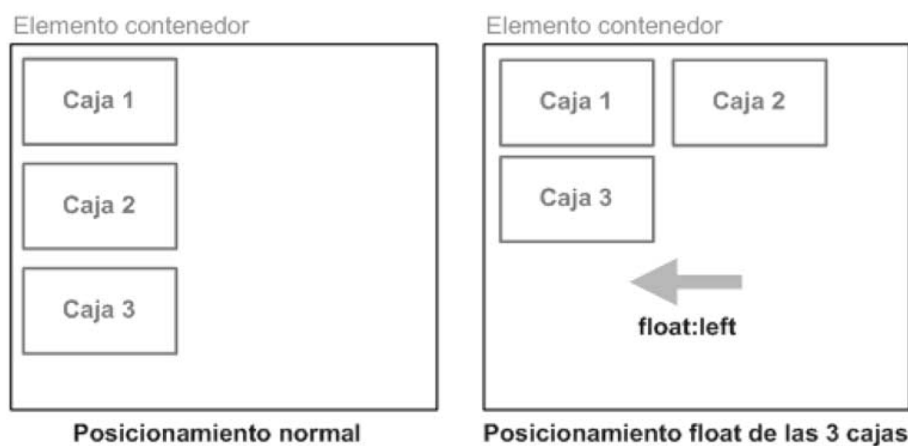


En las imágenes anteriores, el resto de cajas ocupan la posición libre dejada por la "Caja 1" original. Como la "Caja 1" está desplazada a la izquierda, se coloca por encima de la nueva posición que ocupa la "Caja 2", a la que cubre por completo.

Si existen otras cajas desplazadas hacia la izquierda o derecha, la nueva caja desplazada se coloca al lado de las demás cajas. El siguiente ejemplo desplaza las tres cajas:



Si no existe sitio en la línea actual, la caja pasa a la siguiente línea hasta que encuentra el sitio necesario para mostrarse a la izquierda o derecha de la línea.



Una caja posicionada mediante **float** influye en la disposición de todas las demás cajas. CSS permite definir si el resto de los elementos fluyen alrededor de la caja desplazada o no lo hacen.

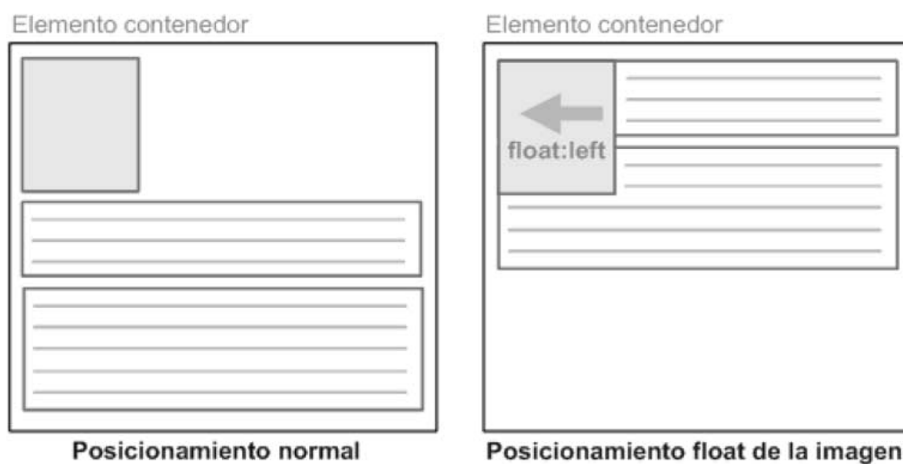
Los elementos en línea que se encuentran al lado de las cajas desplazadas mediante **float** adaptan su anchura al espacio libre dejado por la caja desplazada. Si en la línea donde se encuentra la caja desplazada no existe sitio necesario para los contenidos de los elementos en línea, estos se visualizan en la línea inmediatamente inferior.

Los posibles valores de la propiedad determinan el posicionamiento del elemento y el comportamiento de los elementos adyacentes. Si se indica un valor **left**, el elemento se desplaza hasta el punto más a la izquierda posible en esa misma línea (si no existe sitio en esa línea, el elemento baja una línea y se muestra lo más a la izquierda posible en esa nueva línea).

El resto de elementos adyacentes se adaptan y fluyen alrededor del elemento desplazado.

El valor **right** tiene un funcionamiento idéntico, salvo que en este caso el elemento se desplaza hacia la derecha. El valor **none** permite eliminar el posicionamiento y que el elemento se muestre en su posición original.

Los elementos que se encuentran alrededor de un elemento que ha sido posicionado mediante **float**, adaptan sus contenidos para que fluyan alrededor del elemento posicionado:

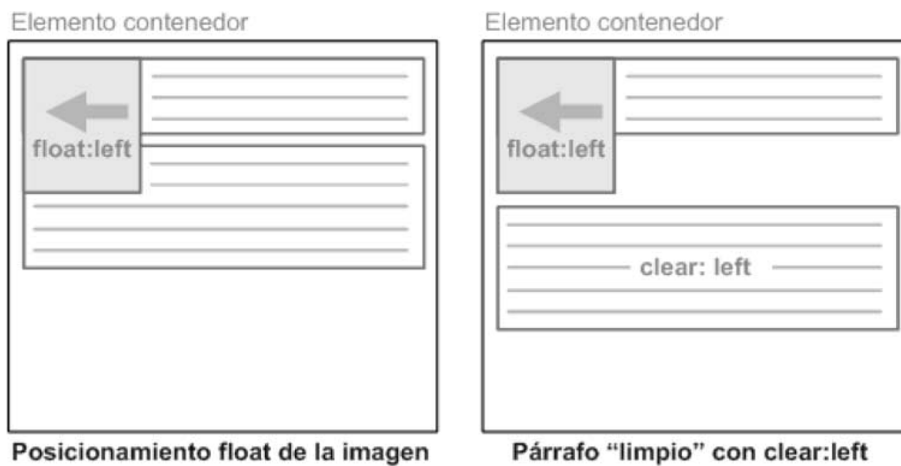


La regla CSS que se aplica en la imagen del ejemplo anterior es:

```
img {float: left;}
```

La principal motivación de la creación del posicionamiento mediante **float** fue precisamente la posibilidad de colocar imágenes alrededor de las cuales fluye el texto.

La flexibilidad de CSS permite controlar la forma en la que los contenidos fluyen alrededor de los contenidos posicionados mediante **float**. De hecho, en muchas ocasiones es admisible que el texto del primer párrafo fluya alrededor de una imagen, pero el resto de los párrafos deberían mostrarse en su totalidad y no fluyendo alrededor de la imagen:



La propiedad `clear` permite contrarrestar el comportamiento por defecto de los `float` y permite forzar a un elemento a mostrarse debajo de cualquier elemento posicionado con `float`.

La regla CSS que se aplica al segundo párrafo del ejemplo anterior sería la siguiente:

```
<p style="clear: left;">...</p>
```

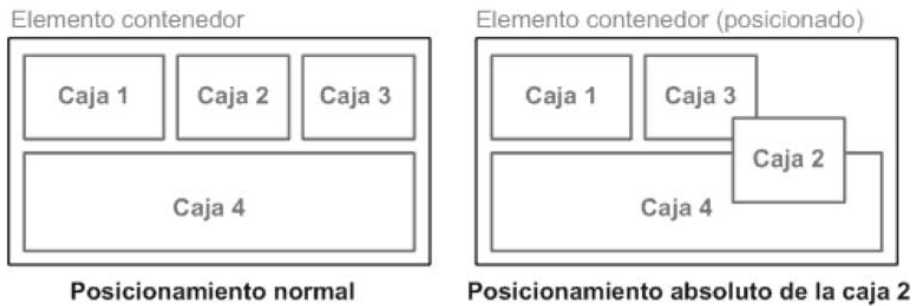
Si se indica el valor **left**, el elemento se desplaza de forma descendente hasta que pueda colocarse en una línea en la que no haya ningún elemento en el lado izquierdo. La especificación oficial de CSS explica este comportamiento como un desplazamiento descendente hasta que el borde superior del elemento esté por debajo del borde inferior de cualquier elemento desplazado hacia la izquierda.

Si se indica el valor **right**, el comportamiento es el mismo, salvo que en este caso se tienen en cuenta los elementos desplazados hacia la derecha.

El valor **both** despeja los lados izquierdo y derecho del elemento, ya que desplaza el elemento de forma descendente hasta que el borde superior se encuentre por debajo de cualquier borde inferior de los elementos desplazados hacia la izquierda y hacia la derecha.

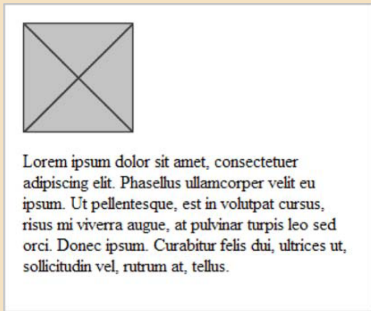
3.9.3. Posicionamiento absoluto

El posicionamiento absoluto implica que el elemento desplazado sale por completo del flujo normal del documento y por tanto, la posición del resto de elementos se determina como si no existiera el elemento desplazado.



La referencia del posicionamiento absoluto de un elemento es el primer elemento padre que esté posicionado. Si ningún elemento padre del elemento posicionado tiene establecido un posicionamiento, la referencia se toma respecto del documento HTML completo.

El siguiente ejemplo muestra las diferencias entre un elemento padre posicionado y otro que no lo está:

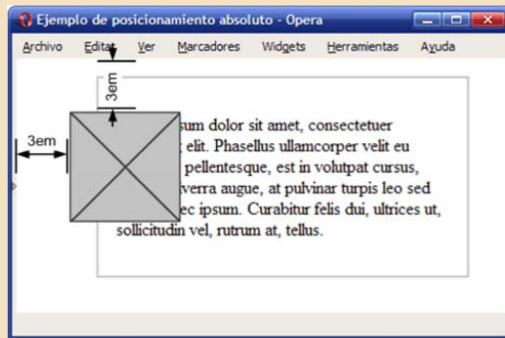
Código CSS	Vista navegador
<pre>div { border: 2px solid #CCC; padding: 1em; margin: 1em 0 1em 4em; width: 300px; }</pre>	 <p> Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus ullamcorper velit eu ipsum. Ut pellentesque, est in volutpat cursus, risus mi viverra augue, at pulvinar turpis leo sed orci. Donec ipsum. Curabitur felis dui, ultrices ut, sollicitudin vel, rutrum at, tellus. </p>
Código HTML	
<pre><div> <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus ullamcorper velit eu ipsum. Ut pellentesque, est in volutpat cursus, risus mi viverra augue, at pulvinar turpis leo sed orci. Donec ipsum. Curabitur felis dui, ultrices ut, sollicitudin vel, rutrum at, tellus. </p> </div></pre>	

Si hacemos el siguiente cambio:

Código CSS

```
div img {  
    position: absolute;  
    top: 3em;  
    left: 3em;  
}
```

Vista navegador



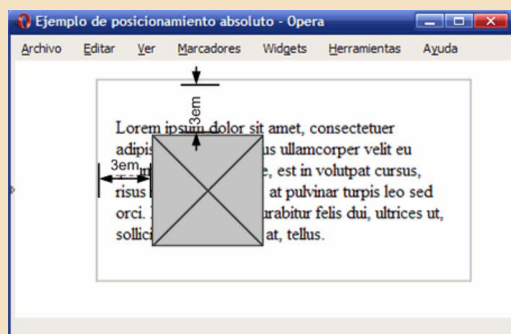
La imagen posicionada absolutamente no toma como origen la esquina superior izquierda del <div> en el que se encuentra, sino que su referencia es la esquina superior izquierda de la página.

Sin embargo, si el elemento contenedor de la imagen (es decir, el elemento <div>) se posiciona de forma relativa, la imagen posicionada de forma absoluta varía su posición:

Código CSS

```
div {
  border: 2px solid #CCC;
  padding: 1em;
  margin: 1em 0 1em 4em;
  width: 300px;
  position: relative;
}
div img {
  position: absolute;
  top: 3em;
  left: 3em;
}
```

Vista navegador



3.10. Layout

Centrar una página completa

A medida que aumenta el tamaño y la resolución de las pantallas de ordenador, se hace más difícil diseñar páginas que se adapten completamente al tamaño de la ventana. Con resoluciones superiores a 1024 x 768, el texto de las páginas se muestra con unas líneas demasiado largas como para leerlas con comodidad. Por ese motivo se suele optar por diseños con una anchura fija limitada a un valor aceptable para mantener la legibilidad del texto.

Por otra parte, las páginas se alinean por defecto a la izquierda de la ventana, dando una sensación de vacío en los monitores con más resolución y las páginas muy estrechas.

Una solución para evitar los grandes espacios en blanco, es crear páginas con una anchura fija adecuada y centrar la página respecto de la ventana del navegador. CSS permite centrar las páginas de forma muy sencilla. Las siguientes imágenes muestran el aspecto de una página centrada a medida que aumenta la anchura de la ventana del navegador.

La solución se basa en agrupar todos los contenidos de la página en un <div> llamado contenedor (en inglés se denomina wrapper) y asignarle a ese <div> unos márgenes laterales automáticos:

Código CSS	Código HTML
<pre>#contenedor { width: 300px; margin: 0 auto; }</pre>	<pre><body> <div id="contenedor"> <h1>Lorem ipsum dolor sit amet</h1> ... </div> </body></pre>

El resultado es una página centrada sea cual sea la resolución de pantalla del ordenador del visitante.

Modificando ligeramente el código CSS se puede conseguir un diseño dinámico (también llamado líquido) que se adapta a la anchura de la ventana del navegador y permanece siempre centrado:

Código CSS
<pre>#contenedor { width: 70%; margin: 0 auto; }</pre>

El resultado es una anchura de página que se adapta a la anchura del navegador. Por eso se le llama dinámico porque en función del usuario es más amplia la página o no.

Estructura o *layout*

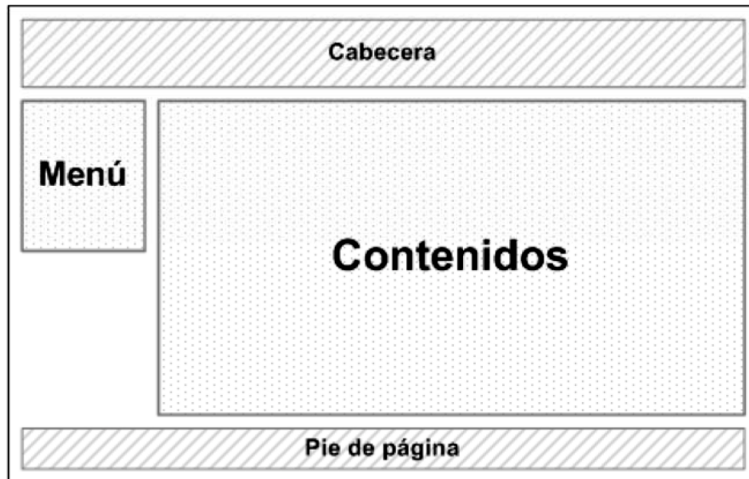
El diseño tradicional de páginas web se basaba en el uso de tablas para definir las estructuras de las páginas y posicionar cada uno de los bloques lógicos que forman las páginas: cabecera, menús, contenidos, columnas laterales, pies de página, etc.

Sin embargo, la estructura basada en tablas es compleja (complica en exceso el código HTML), poco flexible (no permite cambios sencillos en la estructura) y es poco semántica (no es sencillo dotar de significado a la estructura).

Por estos motivos, la estructura basada en tablas ha dado paso a la estructura basada exclusivamente en CSS. Aunque esta nueva alternativa en ocasiones presenta retos importantes, en general es más sencilla y flexible.

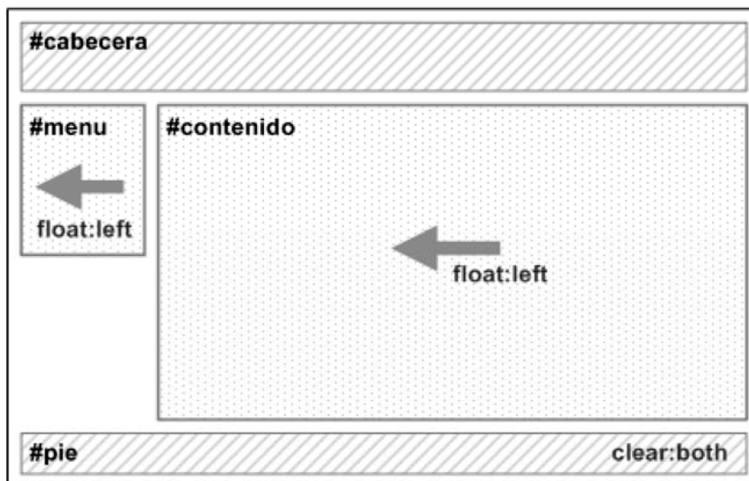
Diseño a 2 columnas con cabecera y pie de página

El objetivo de este diseño es definir una estructura de página con cabecera y pie, un menú lateral de navegación y una zona de contenidos. La anchura de la página se fija en 700px, la anchura del menú es de 150px y la anchura de los contenidos es de 550px:



La solución CSS se basa en el uso de la propiedad float para los elementos posicionados como el menú, y los contenidos y el uso de la propiedad clear en elementos como el pie de página para evitar los problemas ocasionados por los elementos con float.

#contenedor



El código HTML y CSS mínimos para definir la estructura de la página sin aplicar ningún estilo adicional son los siguientes:

Código CSS	Código HTML
<pre>#contenedor { width: 700px; } #cabecera { } #menu { float: left; width: 150px; } #contenido { float: left; width: 550px; } #pie { clear: both; }</pre>	<pre><body> <div id="contenedor"> <div id="cabecera"> </div> <div id="menu"> </div> <div id="contenido"> </div> <div id="pie"> </div> </div> </body></pre>

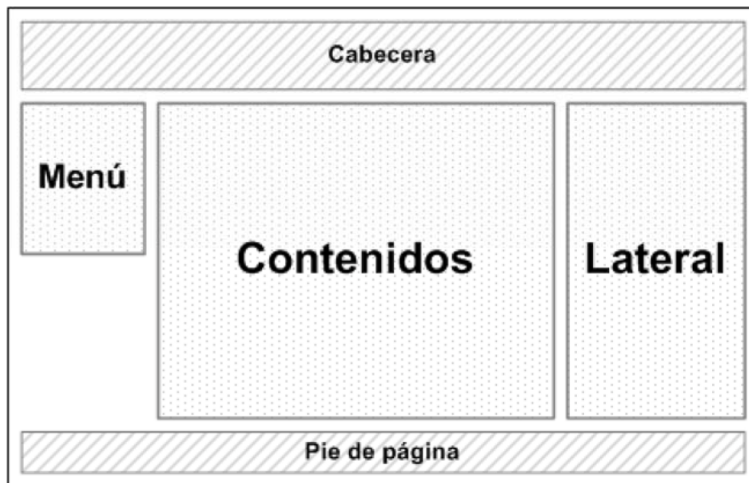
En este ejemplo, el elemento #contenido también se puede alinear a la derecha de la página, manteniendo el elemento #menu alineado a la izquierda.

Para conseguir una página de anchura variable y que se adapte a la ventana del navegador, se deben aplicar las siguientes reglas CSS:

Código CSS	Código HTML
<pre>#contenedor { } #cabecera { } #menu { float: left; width: 15%; } #contenido { float: left; width: 85%; } #pie { clear: both; }</pre>	<pre><body> <div id="contenedor"> <div id="cabecera"> </div> <div id="menu"> </div> <div id="contenido"> </div> <div id="pie"> </div> </div> </body></pre>

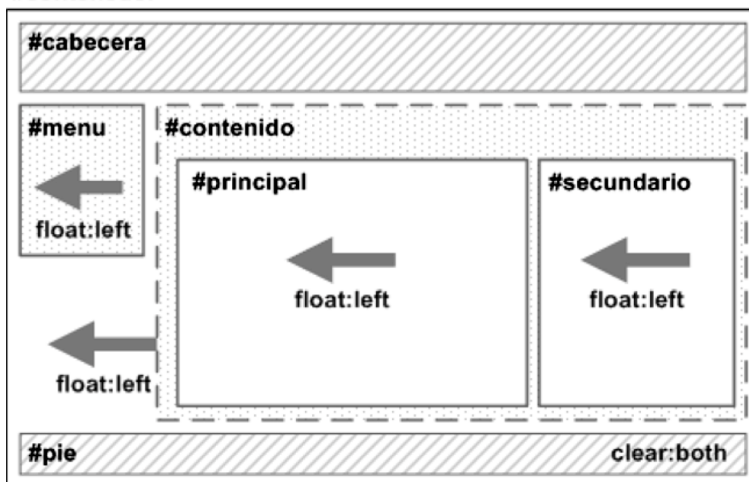
Diseño a 3 columnas con cabecera y pie de página

Además del diseño a 2 columnas, el diseño más utilizado es el de 3 columnas con cabecera y pie de página. En este caso, los contenidos se dividen en 2 zonas diferenciadas: zona principal de contenidos y zona lateral de contenidos auxiliares:



La solución CSS emplea la misma estrategia del diseño a 2 columnas y se basa en utilizar las propiedades `float` y `clear`:

#contenedor



El código HTML y CSS mínimos para definir la estructura de la página sin aplicar ningún estilo adicional son los siguientes:

Código CSS	Código HTML
<pre>#contenedor { } #cabecera { } #menu { float: left; width: 15%; } #contenido { float: left; width: 85%; } #contenido #principal { float: left; width: 80%; } #contenido #secundario { float: left; width: 20%; } #pie { clear: both; }</pre>	<pre><body> <div id="contenedor"> <div id="cabecera"> </div> <div id="menu"> </div> <div id="contenido"> <div id="principal"> </div> <div id="secundario"> </div> <div id="pie"> </div> </div> </body></pre>

El código anterior crea una página con anchura variable que se adapta a la ventana del navegador. Para definir una página con anchura fija, solamente es necesario sustituir las anchuras en porcentajes por anchuras en píxel.