Introducción al HTML

Elementos en un documento HTML

En esta guía, a las instrucciones que forman el lenguaje HTML les llamaremos **elementos**. Otros autores también las denominan "TAGS", o "ETIQUETAS". La notación de los elementos consiste en los símbolos < y > que encierran dentro una instrucción.

Los elementos pueden ser llenos o vacíos.

Elementos llenos

Se forman escribiendo la instrucción correspondiente seguida del texto al que se quiere aplicar la instrucción y se termina repitiendo la instrucción pero con una barra inclinada inmediatamente antes de la misma.
Por ejemplo, el elemento **H1** que sirve para dar el máximo tamaño al texto, se escribirá:

```
<H1> Texto de prueba </H1>.
```

y este sería el resultado:

Texto de prueba

Si olvidas poner </H1> todo el resto de la página tendrá el mismo tamaño grande.

Elementos vacíos

Los elementos vacíos se escriben como los llenos, pero no es necesario poner la instrucción repetida al final con una barra. Esto se debe a que estos elementos no producen un efecto sobre el texto. Generalmente se utilizan para separar bloques de texto, y por tanto no es necesario indicar su fin. Empiezan y terminan en el mismo punto.

Por ejemplo, el elemento **<HR>** que sirve para dibujar una línea horizontal en la pantalla, se escribirá:

```
<HR>
```

Y este sería el resultado:

Elementos con argumento

Algunos elementos se escriben con argumento. Es como pasarle parámetros a la instrucción, y se llaman **atributos** del elemento.

Por ejemplo, el elemento **<A>** que sirve para hacer un link (enlace) con otro documento o con otra página del actual, se escribirá:

```
<A HREF="http://www.miservidor.es/mifichero.htm"> Link de prueba  </A>.
```

Este es un elemento lleno donde al atributo HREF se le asigna el valor que aparece entre comillas: "http://www.miservidor.es/mifichero.htm". El texto al que afecta este elemento es **Link de prueba** y realiza un link con el fichero **mifichero.htm** que está en el servidor **www.miservidor.es**.

Los elementos pueden escribirse tanto en mayúsculas como en minúsculas. Puede ser preferible la primera opción ya que aporta claridad al documento fuente, y eso se agradece a la hora de hacer correcciones, pero CUIDADO, el valor de algunos atributos hay que escribirlos EXACTAMENTE como deban ser. En el ejemplo anterior no funcionaría el link si escribiéramos *www.miservidor.es* en mayúsculas, eso es un nombre de máquina y sería interpretado como OTRA máquina.

Los elementos pueden anidarse unos con otros, teniendo cuidado de poner los cierres en el lugar adecuado. Por ejemplo, el elemento **<H1>** combinado con **<I>** que sirve para generar texto en itálica, se escribirá:

```
<H1><I>Texto de prueba </I></H1>.
```

y este sería el resultado:

Texto de prueba

Los elementos, en HTML, los puedes escribir tanto en mayúsculas como en minúsculas. Funciona igual <P> que <p>. A la hora de revisar el código escrito, resulta más claro si se escribieron en mayúsculas, pero si en el futuro hay que convertir la página a otro estándar, como el XHTML, no se admitirán las mayúsculas. Si no piensas migrar a otras tecnologías, hazlo como más cómodo te resulte.

Estructura de una página HTML

Las páginas HTML están estructuradas en dos partes diferenciadas: la **HEAD** (cabecera) y el **BODY** (cuerpo).

Ejemplo de página

Si escribes:

```
<!Tipo de documento>
<HTML>
<HEAD>
<TITLE> Documento de prueba </TITLE>
</HEAD>

<!-- Esto es un comentario-->

<BODY>

<H1> Esto es una "demo" de documento HTML </H1>

Esto es el más sencillo de los documentos HTML.

</BODY>
</HTML>
```

El **tipo de documento** no es obligatorio a efectos prácticos, es decir que la página se verá igual tanto si lo escribes como si no. Sólo sirve como identificación del tipo de contenido del fichero a los efectos de cumplir las especificaciones de estándar recomendadas por el consorcio W3C, que es el organismo que regula el lenguaje. Por ejemplo:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
        "http://w3.org/TR/REC-html40/loose.dtd">

Indica que cumple el estándar HTML 4.0

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"
        "http://w3.org/TR/REC-html40/strict.dtd">

Significa que cumple el estándar HTML 4.0 y, además, no contiene
elementos desaconsejables.

Y esta es una definición de marcos que cumple el estándar HTML 4.0:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN"
        "http://w3.org/TR/REC-html40/frameset.dtd">
```

En el HTML 4.0 se consideran desaconsejables aquellos elementos que, aún siendo soportados, han sido sustituidos por otros y, por ello, es posible que sean eliminados en el futuro. En cualquier caso, el que un elemento escrito en tus páginas se vea o no, dependerá siempre del navegador utilizado y/o de su versión, no de lo que diga el estándar.

Puedes ver el aspecto real del ejemplo de página pulsando aquí.

Para escribir comentarios en la página (que sólo se ven en el texto fuente, pero no en el visualizador) se utilizará el elemento **<!-- -->** **cuidado**: el que no se vea de forma normal en el navegador, no significa que no pueda verse el código fuente. Nunca escribas comentarios con claves de acceso o datos confidenciales.

Esquema de tipos de un documento HTML

Cuando un visualizador recibe un fichero, antes de mostrarlo necesita saber el tipo de contenido que éste tiene, a fin de procesarlo adecuadamente. No es lo mismo recibir un documento de texto que una fotografía en formato GIF, o un vídeo en formato MPG, o si lo que estamos intentando es transferir un fichero. En cada caso, el visualizador pondrá en juego distintas habilidades, de las muchas que posee.
Si el fichero procede de un servidor **http** remoto, (es decir un servidor web; *http* es el nombre del protocolo que utilizará el servicio invocado), éste se encargará de decirle al visualizador cuanto necesite saber, pero si lo que hay que procesar es un fichero local, la única forma de saberlo es por la **extensión** del fichero.

La extensión de un fichero son las tres o cuatro letras (depende del sistema operativo con el que se trabaje) que hay después del nombre del fichero y que están separadas del mismo por un punto.

Por ejemplo, esta página se llama estruc.**htm**. Las tres letras que hay después del punto (htm) son la **extensión** del fichero. Las otras, las que están delante del punto, son el **nombre** del fichero.

Las extensiones de 4 o más letras son típicas de sistemas operativos como UNIX o LINUX y las de 3 o menos de Windows, que ha mantenido esta costumbre por herencia de los tiempos del DOS, ya que puede trabajar perfectamente con extensiones de más de 3 letras. Por ejemplo, la extensión **.html** funciona en Windows igual que la **.htm**
Estas son algunas de las extensiones stándard más comunes y sus contenidos. La primera de ellas suele utilizarse en sistemas UNIX-LINUX y la segunda en Windows:

`.html` O `.htm`

Documento HTML. Contine texto e instrucciones HTML que serán interpretadas.

.text o .txt

Contiene texto plano. El visualizador presentará todo el fichero como si fuera un único bloque de texto y no interpretará ninguna instrucción HTML que pudiera llevar. Esto lo hará con cualquier fichero que lleve una extensión desconocida o simplemente no lleve ninguna.

.gif

Contiene un fichero de imagen en formato GIF.

.npg

Contiene un fichero de imagen en formato NPG.

.xbm

Contiene un fichero de imagen en formato X-Bitmap (blanco y negro).

.xpm

Contiene un fichero de imagen en formato X-Pixmap (color).

.jpeg o .jpg

Contiene un fichero de imagen en formato codificado jpeg.

.mpeg o .mpg

Contiene un fichero de imagen de video o cine (en movimiento).

.au

Contiene un fichero de audio (sonido) codificado en aiff-encoded.

.mid

Contiene un fichero de audio (sonido) generado con secuenciadores midi típicos de Windows.

.avi

Contiene un fichero con video y sonido típico de Windows.

.mp3

Contiene un fichero con sonido típico de Windows y otros sistemas.

.exe

Contiene un fichero binario ejecutable en DOS o Windows.

.hqx

Contiene un fichero binario ejecutable en Macintosh.

.Z o .zip

Contiene un (o varios) fichero/s comprimido/s de cualquier clase. Para poder utilizarlo hay que indicarle al visualizador una herramienta de compresión-descompresión.

.pdf

Contiene un fichero de imagen de cualquier documento generado con Acrobat Writer.

.doc

Contiene un fichero de documento generado por Microsoft Word

En el entorno web se utilizan los **MIME** types (**M**ultipurpose **I**nternet **M**ail **E**xtension) para definir el tipo de un fichero transferido. El visualizador determina desde el MIME type cómo hay que tratar cada fichero.

Estos son los mime types más habituales junto con las extensiones de fichero a las que suele relacionarse:

```
application/rtf                     .rtf
application/pdf                     .pdf
application/postscript                    .ps
application/msword                  .doc
application/x-msexcel              .xls
application/x-mspowerpoint          .ppt
application/asx                     .asx
application/x-mplayer2              .asx .avi .mp3 .mpg
application/x-shockwave-flash       .swf
application/x-java-applet           .class .clas
application/java                    .clas .class
application/octet-stream            .exe .com
application/x-compressed            .zip

audio/mpeg                          .mp3 .mpg
audio/x-wav                         .wav
audio/x-ms-wma                      .wma
audio/x-midi                        .mid .midi
audio/x-ms-mp3                      .mp3
audio/x-mp3                         .mp3
audio/mpeg3                         .mp3
audio/x-mpeg-3                      .mp3 .mpeg
audio/x-pn-realaudio                .rm

image/jpeg                          .jpg .jpeg
image/gif                           .gif
image/png                           .png

text/html                           .htm .html
text/plain                          .txt .text .htm .html
text/css                            .css
text/x-javascript                   .js
text/richtext                      .rtf
text/rtf                            .rtf

video/mpeg                          .mpg .mpeg
video/avi                           .avi
video/x-ms-wmv                     .wmv
video/x-msvideo                     .wmv
video/quicktime                     .mov
video/x-ms-asf                      .asf
```

Los visualizadores tienen una configuración de los mapas de tipos aceptables, e instrucciones de cómo proceder en cada caso. Algunos tipos son tratados directamente por el propio visualizador (p.ej.: htm, gif, txt) mientras que para otros necesitan de una herramienta auxiliar adecuada, que hay que indicar al visualizador, para que sea llamada en el momento preciso (p.ej.: zip, mpg, avi, pdf).

Ahora que ya sabes qué son las extensiones de fichero, es el momento de aclarar una cuestión importante:

Los documentos HTML los puedes escribir con lo que quieras... Siempre que los salves en modo **Solo Texto**. Es decir, que si utilizas, por ejemplo, Word de Windows o de MAC, por defecto estos programas salvan en formato Word. Y aunque luego los renombres como documentos **.htm** o **.html**, el formato interno sigue siendo Word.

Tampoco vale dejarlos con la extensión **.txt** que les dejan la mayoría de editores al salvarlos en modo Solo Texto. No serían interpretados correctamente. Por lo tanto si tienen extensión **.txt** (y por supuesto **son** .txt), entonces puedes renombrarlos a .htm, o bien en el momento de guardarlos en formato **Solo Texto** no dejar la extensión por defecto .txt y darle ya directamente la extensión **.htm**.

Algunos procesadores de texto ya incluyen entre sus tipos el **.htm**. Si es así, lo que hacen es eliminar la codificación propia, y convertirla a elementos de HTML. Por ejemplo, si tienes un texto en itálica, automáticamente es convertido al elemento <I>, etc.. El problema de los editores avanzados de texto y de la mayoría de programas asistentes para escribir HTML es que generan un código bastante "sucio", difícil de corregir o modificar después. No es que no funcione, pero como el programa no sabe cual es la finalidad del texto escrito, por defecto le pone todo lo que sabe por si acaso, aplicando el dicho de "mejor que sobre..."

Cabecera (HEAD) de un documento HTML

La **HEAD** es la primera de las dos partes en que se estructura un documento HTML.

En la HEAD reside información acerca del documento, y generalmente no se ve cuando se navega por él. En la HEAD se pone el elemento lleno <**TITLE**> que es una breve descripción que identifica la página. Es lo que el navegador se guarda en el "**Bookmarks**" (libro de marcas o libro de direcciones), con lo que crea la lista que aparece en la orden "**Go**" de la barra de órdenes, lo que aparece en la esquina superior izquierda cuando se imprime el documento, y lo que aparece en el marco de la ventana del navegador. También lo guarda en su caché, y servirá para mostrar la página, cuando sea llamada otra vez, sin necesidad de conectarse de nuevo al servidor de origen.

No hay que confundir el elemento <TITLE> con el nombre del fichero. Por ejemplo, esta página está contenida en un fichero llamado **head.htm** y el texto de su <TITLE> es: **Head de un documento**. Se escibirá asi:

```
<HEAD>
<TITLE>Head de un documento </TITLE>
</HEAD>
```

Dentro de la HEAD puede utilizarse otro elemento: **META**. Por ejemplo, si se escribe:

```
<HEAD>
<TITLE>Head de un documento </TITLE>
<META HTTP-EQUIV="Refresh" CONTENT="10">
</HEAD>
```

Esto hace que el visualizador vuelva a cargar la página activa al cabo de 10 segundos. También puede hacerse a un servidor. Así:

```
<HEAD>
<TITLE>Head de un documento </TITLE>
<META HTTP-EQUIV="Refresh" CONTENT="10;
URL=http://miservidor/mipagina.htm">
</HEAD>
```

Utiliza esto con precaución. Si sabes que el contenido de la página no va a cambiar, es inútil hacer esto, y si lo haces contra un servidor, puedes sobrecargarlo. Este elemento, sólo tendrá utilidad en casos muy especiales.

Otra opción es forzar la expiración inmediata en la caché del navegador de la página recibida, lo que provoca que la página sea solicitada de nuevo al servidor cada vez, en lugar de cargar la copia que ya existe en la máquina del cliente. Se escribe así:

```
<HEAD>
<TITLE>Head de un documento </TITLE>
<META HTTP-EQUIV="Expires" CONTENT="Tue, 20 Aug 1996 14:25:27 GMT">
</HEAD>
```

Si se pone una fecha ya pasada, como la que hay en el ejemplo, el navegador elimina inmediatamente de la caché la página recibida, y si no es pasada, lo hará en el momento indicado por la misma. También se le puede dar valor cero a la fecha de expiración:

```
<HEAD>
<TITLE>Head de un documento </TITLE>
<META HTTP-EQUIV="Expires" CONTENT="0">
</HEAD>
```

Otra opción es impedir directamente que el navegador guarde en caché la página. Esto es especialmente útil cuando se trabaja con formularios que consultan datos dinámicos:

```
<HEAD>
<TITLE>Head de un documento </TITLE>
<META HTTP-EQUIV="Expires" CONTENT="no-cache">
</HEAD>
```

Si tienes interés en que tus páginas aparezcan en los grandes buscadores de Internet, y puedan ser encontradas con facilidad, puedes poner las palabras clave que contiene la página separadas por comas. Por ejemplo:

```
<HEAD>
<TITLE>Head de un documento </TITLE>
<META NAME="keywords" CONTENT="HTML, internet  ">
</HEAD>
```

Este otro sirve para que los buscadores puedan ofrecer un breve resumen de los contenidos de tu página:

```
<HEAD>
<TITLE>Head de un documento </TITLE>
<META NAME="description" CONTENT="Manual para escribir HTML.">
</HEAD>
```

El siguiente es importante con los nuevos navegadores, ya que le indica la tabla de caracteres que se ha empleado al escribir la página. De no usarlo, puede ocurrir que el navegador no muestre correctamente los caracteres especiales no ascii, tales como acentos, letras de alfabetos no occidentales, etc., que se hayan quedado sin codificar de la forma típica en html.

```
<HEAD>
<TITLE>Head de un documento </TITLE>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</HEAD>
```

Los charset más habituales suelen ser:

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8 /
html5">
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-15">
<meta http-equiv="Content-Type" content="text/html; charset=windows-
1252">
```

Puedes poner todos los elementos <**META**> que creas necesarios, pero sin repetirlos.

Habrás notado que se ha utilizado la palabra "caché", y tal vez no sepas a qué se refiere. Todos los navegadores, por defecto, siempre que reciben una página de un servidor se hacen una copia de la misma en el disco de tu máquina. Con esto se pretende que si vuelves a solicitar la misma página, en lugar de pedirla de nuevo al servidor, te mostrará la que tiene guardada en el disco. A esta área del disco donde el navegador va poniendo las páginas visitadas, se le denomina **caché**. El tamaño de la caché lo puedes modificar desde las opciones de configuración del navegador, e incluso puedes darle tamaño cero, con lo que siempre que veas una página, ésta habrá sido solicitada al servidor independientemente de lo que digan las instrucciones META.

Hay otros elementos que pueden aparecer en la HEAD, como ISINDEX, NEXTID, LINK o BASE, pero son de uso muy especializado y poco corriente, algunos ya en desuso, y ninguno obligatorio (<TITLE> sí lo es).

Cuerpo (BODY) de un documento

El cuerpo (**BODY**) es la segunda y última de las dos partes en que se estructura un documento HTML. Por supuesto es obligatoria, ya que es aquí donde reside el verdadero contenido de la página, y por tanto, al contrario de la HEAD sí se ve cuando navegamos por ella.

Se escibirá asi:

```
<BODY>

texto texto texto texto texto texto texto texto texto texto texto texto
texto texto texto texto texto texto texto texto texto texto texto texto

</BODY>
```

Como ya habrás visto, los navegadores, por defecto, presentan el texto de una página ajustando los contenidos a la esquina superior izquierda de la pantalla. El elemento BODY tiene parámetros que permiten modificar los márgenes por defecto. Lo malo es que cada navegador los interpreta de una manera. Por ejemplo, Netscape utiliza solamente dos instrucciones que afectan simultáneamente a los márgenes superior e inferior e izquierdo y derecho respectivamente:

marginwidth="pixels", para los márgenes izquierdo y derecho.
marginheight="pixels", para los márgenes superior e inferior.

En cambio, el Internet Explores, utiliza uno para cada cual:

leftmargin="pixels", para el margen izquierdo
topmargin="pixels", para el margen superior
rightmargin="pixels", para el margen derecho
bottommargin="pixels", para el margen inferior

Donde **pixels** es el número de pixels que se quiere mover cada margen hacia el interior de la página. Estos parámetros también son accesibles desde instrucciones de estilo.

Otro problema que suele presentarse con los márgenes es que a la hora de imprimir una página, cada impresora, junto con el navegador, se las ingeniarán para hacer justo lo contrario de lo que se desea... Tal vez algun dia todo esto funcione bien.

Otra cosa que puede controlarse desde BODY es el color que tomarán los links que pongamos en la página. Por ejemplo, para hacer que los links sean amarillos antes de ser visitados, azul-verde cuando son activados, y verdes después de haber sido utilizados, se escribe:

<BODY LINK="#FFFF00" VLINK="#22AA22" ALINK="#0077FF">

Veamos a hora los elementos más habituales del lenguaje que deben escribirse en el BODY.

Tamaños y tipos de letra en HTML

Para definir distintos tamaños de letra, en HTML se utiliza el elemento lleno **<Hx>** </Hx> donde x es un número que puede variar entre 1 y 6, siendo 1 el tamaño mayor.

Se escribirán así:

```
<H1> Texto de prueba (H1)</H1>.
<H2> Texto de prueba (H2)</H2>
<H3> Texto de prueba (H3)</H3>
<H4> Texto de prueba (H4)</H4>
<H5> Texto de prueba (H5)</H5>
<H6> Texto de prueba (H6)</H6>
```

y este sería el resultado:

Texto de prueba (H1)

Texto de prueba (H2)

Texto de prueba (H3)

Texto de prueba (H4)

Texto de prueba (H5)

Texto de prueba (H6)

No hay que olvidar poner el cierre </Hx> a cada elemento utilizado. Este tipo de elemento se suele utilizar para escribir encabezamientos, ya que después del cierre automáticamente el visualizador inserta un salto de párrafo.

Por ejemplo: si escribes

```
<H1> Texto en H1 </H1> <H3> Texto en H3 </H3>
```

se verá:

Texto en H1

Texto en H3

Y no una cosa al lado de la otra, como cabría esperar. Este elemento admite un parámetro de alineación. Así, si escribes:

```
<H3 align=center> Texto en H3 </H3>
```

Texto en H3

Otra forma de cambiar los tamaños de letra es utilizar el elemento **<FONT** > con el atributo **VALOR**, que es un número entre 1 y 7. El valor por defecto del texto es 3. La gran diferencia de esta notación respecto a la anterior es que no se produce un salto de párrafo después de cada cambio, por lo que pueden hacerse cosas como esta:

```
<FONT SIZE=3>A</font><FONT SIZE=4>A</font><FONT SIZE=5>A</font>

<FONT SIZE=6>A</font><FONT SIZE=7>A</font><FONT SIZE=6>A</font>

<FONT SIZE=5>A</font><FONT SIZE=4>A</font><FONT SIZE=3>A</FONT>
```

Dará como resultado:

AAAAAAAAA

Se puede cambiar el tamaño por defecto (3) de toda la página con el elemento **<BASEFONT SIZE**=valor>. El texto tomará el tamaño indicado por **valor** y lo mantendrá hasta que aparezca otro elemento <BASEFONT SIZE=valor> y lo restaure o lo cambie por otro diferente. Tanto si se ha establecido un valor base como si se utiliza el valor por defecto, los tamaños también pueden indicarse de forma **relativa**, por lo que el valor puede ser positivo (+) o negativo (-) respecto al tamaño base. Por ejemplo estos dos valores dan el mismo resultado:

```
<FONT SIZE=5>ABcde</FONT>
<FONT SIZE=+2>ABcde</FONT>
```

ABcde

ABcde

Con la versión 3.0 de Netscape se ha implementado un nuevo atributo del elemento <FONT> que permite elegir tipos de letra entre los varios de que dispone por defecto Windows. Se trata del atributo **FACE**. Este atributo permite forzar el tipo de letra que el diseñador de la página quiere que vea el cliente, sin importar el que por defecto tenga establecido el visualizador.

Si escribes

```
<FONT FACE="arial">Texto de prueba 12345 con tipo ARIAL</FONT>
<FONT FACE="times new roman">Texto de prueba 12345 con tipo TIMES NEW ROMAN</FONT>
<FONT FACE="courier new">Texto de prueba 12345 con tipo COURIER NEW</FONT>
<FONT FACE="courier">Texto de prueba 12345 con tipo COURIER</FONT>
<FONT FACE="roman">Texto de prueba 12345 con tipo ROMAN</FONT>
<FONT FACE="small fonts">Texto de prueba 12345 con tipo SMALL FONTS</FONT>
```

Se verá:

Texto de prueba 12345 con tipo ARIAL
Texto de prueba 12345 con tipo TIMES NEW ROMAN
Texto de prueba 12345 con tipo COURIER NEW
Texto de prueba 12345 con tipo COURIER
Texto de prueba 12345 con tipo ROMAN
Texto de prueba 12345 con tipo VERDANA
Texto de prueba 12345 con tipo SMALL FONTS

Por supuesto, este atributo es compatible con todos los demás ya conocidos, como color y tamaño. Por ejemplo, si escribes

```
<FONT FACE="impact" SIZE=6 COLOR="red">
 Texto de prueba 12345 con tipo IMPACT</FONT>
```

Se verá:

Texto de prueba 12345 con tipo IMPACT

Se pueden hacer todas la combinaciones que se quieran, pero hay que tener presente que si en la máquina cliente no está instalada una determinada fuente, ésta no se verá y en su lugar aparecerá la fuente por defecto del visualizador. No es interesante por tanto, definir tipos raros, que probablemente, no llegarán a verse nunca.

Si a pesar de todo, se define un tipo del que se tienen dudas de que exista en el cliente, se pueden indicar otros tipos alternativos, de forma que el navegador si no tiene el primer tipo, utilizará el siguiente, y si tampoco lo tiene el próximo, etc. Así:

```
<FONT FACE="raro, courier" SIZE=4 COLOR="red">
 Texto de prueba 12345 con tipos alternativos</FONT>
```

Se verá:

Texto de prueba 12345 con tipos alternativos

Como puedes ver, se ha declarado como primer tipo de letra el llamado "raro" que, por supuesto, no existe, y el navegador pasa a utilizar el siguiente, "courier", que sí es habitual.

Texto en color

Se puede controlar el color del texto utilizando el elemento <FONT> con el atributo **COLOR="xxx"**, donde **"xxx"** es el nombre en inglés del color que se

desea. Hay que tener presente que algunos no se verán o se verán mal si la máquina no soporta, por lo menos, 256 colores. Por supuesto, este efecto es anidable con el de tamaño y todos los demás.

Si escribes:

```
<B><FONT COLOR="red">Texto ROJO </FONT>
<br>
<FONT COLOR="blue">Texto AZUL </FONT>
<br>
<FONT COLOR="navy">Texto AZUL MARINO </FONT>
<br>
<FONT COLOR="green">Texto VERDE </FONT>
<br>
<FONT COLOR="olive">Texto OLIVA </FONT>
<br>
<FONT COLOR="yellow">Texto AMARILLO </FONT>
<br>
<FONT COLOR="lime">Texto LIMA </FONT>
<br>
<FONT COLOR="magenta">Texto MAGENTA </FONT>
<br>
<FONT COLOR="purple">Texto PURPURA </FONT>
<br>
<FONT COLOR="cyan">Texto CYAN </FONT>
<br>
<FONT COLOR="brown">Texto MARRON </FONT>
<br>
<FONT COLOR="black">Texto NEGRO </FONT>
<br>
<FONT COLOR="gray">Texto GRIS </FONT>
<br>
<FONT COLOR="teal">Texto TEAL </FONT>
<br>
<FONT COLOR="white">Texto BLANCO </FONT>
<br>
</B>
```

Se verá:

Texto **ROJO**
Texto **AZUL**
Texto **AZUL** **MARINO**
Texto **VERDE**
Texto **OLIVA**
Texto **AMARILLO**
Texto **LIMA**
Texto **MAGENTA**
Texto **PURPURA**
Texto **CYAN**
Texto **MARRON**

Texto **NEGRO**
Texto **GRIS**
Texto **TEAL**

He aquí una combinación de colores y tamaños:
Si escribes:

```
<FONT SIZE=6 COLOR="red">E</FONT><FONT SIZE=4>sto es una </FONT>
<FONT SIZE=6 COLOR="red">P</FONT><FONT SIZE=4>rueba </FONT>
```

Resulta:

Esto es una Prueba

Los colores también se pueden definir en hexadecimal (por ejemplo el rojo es <FONT COLOR=#FF0000>). Cuando son colores básicos, como los del ejemplo de arriba, es más cómodo escribir el nombre aunque sea en inglés, pero cuando se trata de definir un color que "no tiene nombre" no hay más remedio que usar la codificación hexadecimal. En el índice encontrarás una tabla con los códigos de colores.

Las distintas definiciones de los bloques. Separadores.

Si se analiza una página cualquiera veremos que, en realidad, está compuesta de distintos **bloques**. Es algo así como un mosaico en el que cada parte de la composición final tiene sus propios contenidos, que pueden ser texto, gráficos o las dos cosas. Dependiendo de la composición que se quiera hacer, habrá que elegir los elementos más convenientes para que nuestros contenidos aparezcan finalmente con la alineación debida, el tamaño apropiado, etc.

Comenzaremos con el elemento **<Hx>** </Hx> donde x es un número que puede variar entre 1 y 6, siendo 1 el tamaño mayor.

Se escribirán así:

```
<H1> Texto de prueba (H1)</H1>.
<H2> Texto de prueba (H2)</H2>
<H3> Texto de prueba (H3)</H3>
<H4> Texto de prueba (H4)</H4>
<H5> Texto de prueba (H5)</H5>
<H6> Texto de prueba (H6)</H6>
```

y este sería el resultado:

Texto de prueba (H1)

Texto de prueba (H2)

Texto de prueba (H3)

Texto de prueba (H4)

Texto de prueba (H5)

Texto de prueba (H6)

No hay que olvidar poner el cierre </Hx> a cada elemento utilizado. Este tipo de elemento suele emplearse para escribir encabezamientos, ya que después del cierre, automáticamente, el visualizador inserta un salto de párrafo.

Por ejemplo, si escribes:

```
<H1> Texto en H1 </H1> <H3> Texto en H3 </H3>
```

se verá:

Texto en H1

Texto en H3

Y no una cosa al lado de la otra, como cabría esperar. Este elemento admite un parámetro de alineación. Así, si escribes:

```
<H3 align=center> Texto en H3 </H3>
```

Texto en H3

Para definir los párrafos se utiliza el elemento lleno **<P> </P>** (por **P**aragraph). Aunque a menudo no se utiliza el cierre </P>, ya que el texto continuará normalmente hasta que encuentre otro párrafo <P>, es conveniente acostumbrarse a ponerlo siempre. En la nueva definición del XHTML el cierre es obligatorio.

Se escribirá así:

```
<P> Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1
Texto 1 Texto 1 </P>
<P> Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 </P>
```

Y este sería el resultado:

Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1

Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2

Como puede verse, hay una línea en blanco entre los dos bloques. Si no se quiere dejar esa línea vacía entre los dos párrafos puede utilizarse el elemento **
** (por **BR**eak). Es decir, que el elemento
 es un separador, no un indicador de bloque.

Se escribirá así:

```
<P>Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1
Texto 1 Texto 1 <BR>
Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 </P>
```

y este sería el resultado:

Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1
Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2

El elemento <P> admite cuatro atributos de alineación: **ALIGN=LEFT** (por defecto), **ALIGN=RIGHT** , **ALIGN=CENTER** y **ALIGN=JUSTIFY**
Se escribirán así:

```
<P ALIGN=LEFT>
Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1
Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 </P>
<P ALIGN=RIGHT>
Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2
Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 </P>
<P ALIGN=CENTER>
Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3
Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 </P>
<P ALIGN=JUSTIFY>
Texto 4 Texto 4 Texto 4 Texto 4 Texto 4  Texto 4 Texto 4 Texto 4
Texto 4 Texto 4 Texto 4 Texto 4 Texto 4  Texto 4 Texto 4 Texto 4 </P>
```

y este sería el resultado:

Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1 Texto 1

Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2 Texto 2

Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3 Texto 3

Texto 4 tex Texto 4 Texto 4 tex Texto 4 Texto 4 tex Texto 4 Texto 4 Texto 4 Texto 4 Texto 4 Texto 4 Texto 4 Texto 4 tex Texto 4 Texto 4 Texto 4 Texto 4 tex Texto 4 Texto 4 tex Texto 4 Texto 4 tex Texto tex 4 Texto 4 Texto 4 tex Texto 4 Texto 4 tex Texto 4 Texto 4 tex Texto tex 4 Texto 4 Texto 4 tex Texto 4 Texto 4 tex Texto 4 Texto 4 tex Texto tex 4 Texto 4 Texto 4 tex Texto 4 Texto 4 tex Texto 4 Texto 4 tex Texto tex 4 Texto 4

Un elemento que se comporta de forma parecida a
 es **<DIV>** pero que admite los mismos atributos que <P>: **ALIGN=LEFT** **ALIGN=RIGHT** y **ALIGN=CENTER**

Se escribe así:

```
<DIV ALIGN=LEFT>
texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1
texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1
texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1
texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1
</DIV>
<DIV ALIGN=RIGHT>
texto2 texto2 texto2 texto2 texto2 texto2 texto2 texto2 texto2
texto2 texto2 texto2 texto2 texto2 texto2 texto2 texto2 texto2
texto2 texto2 texto2 texto2 texto2 texto2 texto2 texto2 texto2
texto2 texto2 texto2 texto2 texto2 texto2 texto2 texto2 texto2
texto2 texto2 texto2 texto2 texto2 texto2 texto2 texto2 texto2
< /DIV>
<DIV ALIGN=CENTER>
texto3 texto3 texto3 texto3 texto3 texto3 texto3 texto3 texto3
texto3 texto3 texto3 texto3 texto3 texto3 texto3 texto3 texto3
texto3 texto3 texto3 texto3 texto3 texto3 texto3 texto3 texto3
texto3 texto3 texto3 texto3 texto3 texto3 texto3 texto3 texto3
</DIV>
```

Fíjate en que aquí sí se utiliza el cierre **</DIV>**. Este sería el resultado:

texto1 texto1

texto1 texto1

texto2 texto2

texto3 texto3

Como puedes ver, salvo por las alineaciones, que ya se hacen con <P>, no tiene ninguna utilidad práctica, y en general solamente se utiliza para definir bloques especiales definidos con instrucciones de estilo (bordes, tamaño, situación, alineación, color, etc., etc.)

Otro elemento que es casi igual que <DIV> pero que no admite atributos de alineado, y no produce separación de párrafo ni de línea es **<SPAN**>. En realidad, de forma directa no sirve para nada, y ha sido creado también para aplicar las hojas de estilo.

Se escribe así:

```
<SPAN>
texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1
texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1 texto1
</SPAN>
```

Otro interesante efecto es el elemento lleno **<BLOCKQUOTE>** que sirve para presentar párrafos adentrados (como si estuviesen tabulados).

Se escribirá así:

```
<BLOCKQUOTE>
texto texto texto texto texto texto texto texto texto texto
texto texto texto texto texto texto texto texto texto texto
```

```
<BLOCKQUOTE>
texto texto texto texto texto texto texto texto texto texto
texto texto texto texto texto texto texto texto texto texto
</BLOCKQUOTE>
</BLOCKQUOTE>
```

Y este sería el resultado:

texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto

texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto

Fíjate que en este ejemplo hay un anidamiento, y por tanto, debe haber dos cierres **</BLOCKQUOTE>** al final

Otro separador de bloques de texto es el elemento vacío **<HR>** (por **H**orizontal **R**ule). Este sería el resultado:

Al igual que al texto, se le puede incluir un parámetro de color, pero no funciona en todos los navegadores. También se puede cambiar su apariencia añadiéndole el atributo **<NOSHADE>**. Así:

```
<HR NOSHADE>
```

El resultado es:

El elemento <HR> admite dos parámetros: **WIDTH** y **SIZE**. El primero define la longitud de la línea y el segundo su anchura. No es obligado usar los dos a la vez Por ejemplo, si escribes

```
<HR WIDTH=400 SIZE=5>
```

El resultado será:

El valor del atributo WIDTH se puede expresar, como en el ejemplo anterior, en número de puntos (píxels), o en tantos por ciento referidos al ancho total de la ventana. Así:

```
<HR WIDTH=80% SIZE=5>
```

El resultado será:

Además se puede indicar su posición respecto a los márgenes de la ventana con los atributos **ALIGN=CENTER** (por defecto) **ALIGN=LEFT** y **ALIGN=RIGHT**. Por ejemplo:

```
<HR WIDTH=80% SIZE=5 ALIGN=LEFT>
```

El resultado será:

O bien:

```
<HR WIDTH=80% SIZE=5 ALIGN=RIGHT>
```

El resultado será:

Hay otro elemento, aparecido en la versión 6 de Netscape, que permite rodear un texto con un marco, y opcionalmente ponerle una etiqueta. Se trata del elemento: **<FIELDSET>**. Recuerda que si no tienes la versión adecuada de navegador, en los siguientes ejemplos sólo verás el texto, pero no los enmarcados.
Si se escribe:

```
<FIELDSET>
   Esto es una prueba de enmarcado
</FIELDSET>
```

Se obtiene:

Esto es una prueba de enmarcado

Este elemento tiene un parámetro que permite etiquetar el recuadro: **<LEGEND>** Si se escribe:

```
<FIELDSET>
         <LEGEND>Esto es una etiqueta</LEGEND>
   Esto es una prueba de enmarcado
</FIELDSET>
```

Se obtiene:

Esto es una etiquetaEsto es una prueba de enmarcado

El parámetro **<LEGEND>** tiene tres atributos que indican la posición de la etiqueta: **LEFT** (por defecto), **RIGHT** y **CENTER**, aunque no funciona en todos los navegadores. Por ejemplo:

```
<FIELDSET>
        <LEGEND ALIGN=CENTER> Esto es una etiqueta </LEGEND>
   Esto es una prueba de enmarcado
</FIELDSET>
```

Se obtiene:

Esto es una etiquetaEsto es una prueba de enmarcado

Si en estos ejemplos no ves el recuadro o la etiqueta no está donde debiera, es porque no tienes una versión de navegador adecuada.

Por supuesto, dentro del recuadro generado por <**FIELDSET**> se puede poner cualquier cosa: formularios, imágenes, texto, etc.

Textos preformateados

Posiblemente ya te habrás dado cuenta de que, cuando escribes un texto con tu editor, a la hora de ver lo hecho con el visualizador, algunas cosas no están como tú las pusiste... Como por ejemplo poner dos o tres espacios en blanco entre palabras, o intentar encolumnar los datos de una pequeña tabla: ¡acaba todo junto y en la misma línea!

Este efecto se puede eliminar con el elemento lleno **<PRE>** . Al texto que va dentro del elemento PRE se le pueden aplicar todos los elementos que se quiera, siempre que sean elementos que no alteren la posición del texto. Por ejemplo si utilizas el elemento <H>, se rompería el preformateo pero no ocurrirá lo mismo con <FONT SIZE>. Por defecto, los textos preformateados se ven con tipo de letra "curier", es decir de paso fijo y los no preformateados en "Times New Roman". Estos tipos son configurables en el visualizador.

Se escribirá así:

```
<PRE>
       1  2  3  4  5  6  7      <B>Esto es una demostracion de</B>
       8  9 10 11 12 13 14         texto preformateado.
```

```
      15 16 17 18 19 20 21
</PRE>
```

y este sería el resultado:

```
       1  2  3  4  5  6  7          Esto es una demostracion de

       8  9 10 11 12 13 14          texto preformateado.

      15 16 17 18 19 20 21
```

Dentro de un bloque preformateado también se pueden poner links a otros documentos y codificación para caracteres especiales (se verá más adelante lo que es eso).
Este elemento admite el parámetro WIDTH=x donde x define la máxima longitud de línea visualizada. No funciona con Netscape.

El navegador, por defecto, elimina varios espacios en blanco y los convierte en uno solo. Si lo que deseas es separar una palabra de otra más de un espacio, se puede recurrir a una instrucción especial que le indica esto al navegador, pero hay que darla codificada; funciona como un avance de carro, y aunque en pantalla cumple su objetivo, cuando se imprime la página, no todas las impresoras son capaces de interpretarlo correctamente. Se escribe: ** ** y como alias se utiliza: ** **

Por ejemplo, si escribes:

```
texto texto     texto texto
```

Se ve así:

texto texto texto texto

Con Netscape 3.0 se ha implementado un nuevo elemento que permite definir espacios en blanco sin utilizar el elemento <PRE>
Se trata del elemento **<SPACER>**. este elemento puede ir acompañado de los atributos **TYPE** y **SIZE** y se puede utilizar en cualquier zona de una página, pero donde más se aprecia su efecto es en las columnas. Se escibirá asi:

```
<MULTICOL COLS=2>
<SPACER TYPE=HORIZONTAL SIZE=35>
texto texto texto texto texto texto texto texto texto texto texto texto
<P>
<SPACER TYPE=HORIZONTAL SIZE=35>
texto texto texto texto texto texto texto texto texto texto texto texto
<SPACER TYPE=VERTICAL SIZE=35>
texto texto texto texto texto texto texto texto texto texto texto texto
</MULTICOL>
```

Y se ve así:
texto texto

texto texto

Como puedes ver, con **TYPE=horizontal** deja espacio en blanco a la izquierda de la primera linea. Es el valor por defecto. **TYPE=vertical** deja espacio en blanco por arriba de la primera linea. Para ambos tipos el tamaño del espacio se da en puntos (pixels) con el atributo **SIZE**. El valor por defecto es 0.

Fíjate en que entre las dos instrucciones con tipo horizontal hay un elemento <P>. En efecto, es necesario indicarle a <SPACER> donde termina el párrafo para que ejecute la siguiente instrucción del mismo tipo. Este elemento no funciona con el navegador de Microsoft. Para conseguir columnas sin utilizar este elemento la única solución es emplear tablas: <TABLE>

Otros efectos en el texto

Una o varias líneas de texto, pueden estar centradas respecto a los márgenes de la ventana. Esto afecta sólo al texto que hay entre **<CENTER>** y **</CENTER>**, pero no a todo el párrafo. Por ejemplo:

```
texto texto texto texto texto texto texto texto texto texto texto texto
<CENTER>texto centrado texto centrado</CENTER>
texto texto texto texto texto texto texto texto texto texto texto texto
```

Se verá:

texto texto texto texto texto texto texto texto texto txto texto texto texto texto texto texto texto

texto centrado texto centrado
texto texto texto texto texto texto texto texto texto txto texto texto texto texto texto texto texto

Esto, además de a texto sólo, se puede aplicar a una tabla, una imagen o cualquier otra cosa.

Quedan algunos otros elementos que modifican el aspecto del texto. Algunos, aparentemente, hacen la misma cosa, y otros no funcionan con el visualizador de Netscape, por lo que se omiten aquí.

Texto en negrita:

```
<B>Texto en negrita</B>
```

Texto realzado:

```
<STRONG>Texto realzado</STRONG>
```

Texto en itálica:

```
<I>Texto en itálica</I>
```

Texto con énfasis:

```
<EM>Texto con énfasis</EM>
```

```
Texto ejemplo de código:
<CODE>Texto ejemplo de código</CODE>
Texto teletipo:
<TT>Texto teletipo</TT>
```

Texto subrayado:

```
<U>Texto subrayado</U>
```

~~Texto tachado:~~

```
<STRIKE>Texto tachado</STRIKE>
```

Texto de dirección:

```
<ADDRESS>Texto de dirección</ADDRESS>
```

Texto intermitente

```
<BLINK>Texto intermitente</BLINK>
```

Texto superíndice Texto normal

```
<SUP>Texto Superíndice</SUP>
```

Texto subíndice Texto nomal

```
<SUB>Texto Subíndice</SUB>
```

Texto grande

```
<BIG>Texto grande</BIG>
```

Texto pequeño

```
<SMALL>Texto pequeño</SMALL>
```

En los navegadores de última generación se ha implementado un efecto que permite incluir explicaciones ocultas que aparecen al pasar el ratón por encima (sin pulsar), pero sin cambiar de página ni abrir ninguna ventana nueva. Por ejemplo, si escribes:

<ACRONYM TITLE="Hyper Text Markup Language">HTML</ACRONYM>

Al pasar el ratón sobre la palabra HTML, se desvela su significado: HTML

De funcionamiento similar al anterior también tenemos el elemento **<abbr>**

<abbr TITLE="Hyper Text Markup Language">HTML</abbr>

Y este es el aspecto que toma el texto: HTML Este último elemento **NO** funciona con el navegador IE. Como puedes ver, los dos trabajan de la misma forma que el parámetro **title** que se aplica al elemento <A> y se diferencian de éste solamente por el tipo de subrayado, que es más ligero.

Listas y menús

Hay elementos que permiten crear texto con varios formatos de listado: Estos pueden ser **ordenados** <OL> (se refiere a numerados, no ordenados por algun criterio), **desordenados** <UL> (no numerados), **directorios** <DIR>, **menu** <MENU> y **listados de definición** <DL>. Veamos cómo es la sintaxis básica y apariencia de estos elementos, a partir de los cuales, pueden hacerse combinaciones muy complejas mediante anidamientos de unos con otros, hasta conseguir prácticamente cualquier presentación deseada:

Esto es una **lista ordenada** (numerada):

1. Primera linea
2. Segunda linea

Se escribe:

```
<OL>
<LI>Primera linea
<LI>Segunda linea
</OL>
```

Fíjate en que los elementos <LI> no necesitan cierre. Terminan cuando aparece otro igual o se cierra la lista. Puede ser conveniente poner el cierre </LI> si se van a aplicar instrucciones de estilo.

Esto es una **lista desordenada** (no numerada):

- Primera linea
- segunda linea

Se escribe:

```
<UL>
<LI>Primera linea
<LI>Segunda linea
```

```
</UL>
```
En este caso los números han sido sustituidos por unos puntos gruesos. Esa es la apariencia por defecto; se puede cambiar usando el atributo **TYPE** con tres valores (el valor por defecto es **DISC**). Con el valor **CIRCLE** se verá:

- Primera linea
- segunda linea

Se escribe:

```
<UL TYPE=CIRCLE>
<LI>Primera linea
<LI>Segunda linea
</UL>
```

También puede usarse el valor **SQUARE**. Así:

- Primera linea
- segunda linea

Se escribe:

```
<UL TYPE=SQUARE>
<LI>Primera linea
<LI>Segunda linea
</UL>
```

Las listas ordenadas no sólo se pueden ordenar con números. También se pueden utilizar letras y numeración romana tanto en mayúsculas como minúsculas. Para esto se utiliza el atributo TYPE del elemento <OL> con los siguientes valores: **TYPE=1** (por defecto) para números, **TYPE=A** para letras mayúsculas, **TYPE=a** para letras minúsculas, **TYPE=I** para numeración romana en mayúsculas y **TYPE=i** para numeración romana en minúsculas.

Esto es una **lista ordenada con letras mayúsculas**:

A. Primera linea
B. Segunda linea
C. Tercera linea
D. Cuarta linea

Se escribe:

```
<OL TYPE=A>
```

```
<LI>Primera linea
<LI>Segunda linea
<LI>Tercera linea
<LI>Cuarta linea
</OL>
```

Esto es una **lista ordenada con letras minúsculas**:

a. Primera linea
b. Segunda linea
c. Tercera linea
d. Cuarta linea

Se escribe:

```
<OL TYPE=a>
<LI>Primera linea
<LI>Segunda linea
<LI>Tercera linea
<LI>Cuarta linea
</OL>
```

Esto es una **lista ordenada con numeración romana en mayúsculas**:

I. Primera linea
II. Segunda linea
III. Tercera linea
IV. Cuarta linea

Se escribe:

```
<OL TYPE=I>
<LI>Primera linea
<LI>Segunda linea
<LI>Tercera linea
<LI>Cuarta linea
</OL>
```

Esto es una **lista ordenada con numeración romana en minúsculas**:

i. Primera linea
ii. Segunda linea
iii. Tercera linea
iv. Cuarta linea

Se escribe:

```
<OL TYPE=i>
<LI>Primera linea
<LI>Segunda linea
<LI>Tercera linea
<LI>Cuarta linea
```

```
</OL>
```

En algunos casos puede interesarnos que la lista no comience por el número 1 (por ejemplo si es una lista que continua en otra página). Se puede conseguir con el atributo **START** combinado con **TYPE**.

Esto es una **lista ordenada con letras mayúsculas y que comienza por la E**:

E. Primera linea
F. Segunda linea
G. Tercera linea
H. Cuarta linea

Se escribe:

```
<OL TYPE=A START=5>
<LI>Primera linea
<LI>Segunda linea
<LI>Tercera linea
<LI>Cuarta linea
</OL>
```

El número que pones en el atributo START indica en que número o letra comenzará la lista. la E es la quinta letra.

Esto es un **menú**:

- Primera linea
- Segunda linea

Se escribe:

```
<MENU>
<LI>Primera linea
<LI>Segunda linea
</MENU>
```

La diferencia entre un menú y una lista desordenada, es que las líneas en la lista desordenada comienzan en el margen izquierdo y las del menú unas posiciones más a la derecha (aunque eso depende del visualizador, con Netscape se ven igual).

Esto es un **directorio**:

- Primera linea
- Segunda linea

Se escribe:

```
<DIR>
<LI>Primera linea
<LI>Segunda linea
</DIR>
```

Ocurre lo mismo que con el menú, con Netscape no se aprecia diferencia.

Esto es una **lista de definicion**:

Primera linea

Segunda linea

Se escribe:

```
<DL>
<DT>Primera linea
<DD>Segunda linea
</DL>
```

Fíjate que ahora en lugar del elemento <LI> se utiliza <**DT**> y <**DD**>, y como tipo **<DL>**. Estos tres nuevos elementos también se pueden usar con cualquiera de los anteriores tipos de lista, alterando por completo su apariencia. Para hacerse una idea de la variedad de aspectos que se pueden conseguir, lo mejor es ver la página de ejemplos de listados.

Tablas

Las tablas son sin duda uno de los elementos más potentes del HTML. Con pocos elementos se pueden conseguir efectos espectaculares. En el interior de las celdas de una tabla, que pueden ser con borde visible o invisible, se puede poner cualquier cosa: texto de cualquier tamaño y color, imágenes, links... Por supuesto, además de permitir cualquier contenido, tienen sus propios atributos de alineación tanto horizontal como vertical, y atributos de dimensionado. Por defecto se autodimensionan, es decir, se adaptan en tamaño a su contenido, pero también es posible definirlo de forma fija. Las tablas han sido utilizadas largo

tiempo para maquetar el contenido de las páginas, y aunque las nuevas tendencias aconsejen hacer esto con bloques (capas XHTML), no es un elemento que vaya a desaparecer, y es mucho más sencillo de utilizar.

El elemento básico de definición de tabla es <**TABLE**> </**TABLE**> y en su interior se disponen los sub elementos <**TR**> para definir una fila (**R**ow) <**TH**> para definir una cabecera (**H**eader) <**TD**> para definir una celda de datos (**D**ata). Estos sub elementos también han de llevar sus correspondientes cierres: </**TR**> </**TH**> </**TD**>.

Una cabecera <**TH**> es lo mismo que una celda de datos <**TD**> pero de forma automática el texto de su contenido recibe los atributos de negrita y centrado. Sólo es posible definirlas al principio de las filas, de las columnas o de ambas a la vez.

He aquí una tabla-resumen de los elementos utilizados y los atributos que admite cada uno:

	TABLE	TR	TD	TH	CAPTION
BORDER	X	-	-	-	-
BORDERCOLOR	X	-	-	-	-
ROWSPAN	-	-	X	X	-
COLSPAN	-	-	X	X	-
ALIGN	-	X	X	X	X
VALIGN	-	-	X	-	-
WIDTH	X	-	X	-	-
HEIGTH	X	-	X	-	-
CELLPADDING	X	-	-	-	-
CELLSPACING	X	-	-	-	-
NOWRAP	-	-	X	-	-
EVENTS	X	X	X	X	-

Resumen de tablas

Veamos el significado de cada atributo:

- **BORDER** Indica el ancho de los bordes de la tabla. Se mide en píxels. Si no se escribe este atributo, es equivalente a BORDER=0 (por defecto).
- **BORDERCOLOR** Establece el color de los bordes de la tabla. No funciona igual en todos los navegadores.

- **CELLSPACING** Indica el número de píxels que separan una celda de otra. Aunque pueda parecerlo, no hace lo mismo que BORDER. Su valor por defecto es 2.
- **CELLPADDING** Indica los píxels de separación entre el borde de la celda y su contenido. Su valor por defecto es 1.
- **WIDTH** Según donde se escriba, sirve para controlar el ancho de toda la tabla o de sus columnas. Si se incluye en <TABLE> puede indicar el tamaño tanto en píxels como en porcentaje respecto al ancho de la pantalla.
- **ALIGN** Indica la alineación horizontal de los datos dentro de las celdas. Puede tener tres valores: **LEFT** (izquierda), **RIGHT** (derecha) y **CENTER** (centro).
- **VALIGN** Indica la alineación vertical de los datos dentro de las celdas. Puede tener tres valores: **TOP** (arriba), **BOTTOM** (abajo) y **MIDDLE** (centro).
- **ROWSPAN** Se utiliza en la definición de una celda (<TD>) o cabecera (<TH>) para indicar que su anchura o altura son equivalentes a un determinado número de filas.
- **COLSPAN** Se utiliza en la definición de una celda (<TD>) o cabecera (<TH>) para indicar que su anchura o altura son equivalentes a un determinado número de columnas.
- **NOWRAP** Para impedir que las líneas de texto dentro de una celda se trunquen en los espacios en blanco.
- **EVENTS** Se pueden capturar todos los eventos típicos de los navegadores en cualquiera de las partes de una tabla. Para ello es necesario JavaScript, y no es compatible con todos los navegadores. Por ejemplo: `<TABLE BORDER onMouseOver="javascript:alert('Aviso')"> ... </TABLE>`

Recientemente se han implementado al elemento <TABLE> algunos atributos muy interesantes que permiten definir qué bordes o líneas de la tabla se mostrarán:

Este atributo sirve para definir qué bordes del marco de la tabla serán visibles: **<TABLE FRAME="valor"> ... </TABLE>** donde **valor** puede ser:

- **void** - Ningún lado (por defecto).
- **above** - Sólo el borde superior
- **below** - Sólo el borde inferior.
- **hsides** - Sólo los bordes superior e inferior.
- **vsides** - Sólo los lados derecho e izquierdo.

- **lhs** - Sólo el lado izquierdo.
- **rhs** - Sólo el lado derecho.
- **box** - Los cuatro lados.
- **border** - Los cuatro lados (no es lo mismo que el ya conocido)

Y este otro sirve para definir qué bordes de la tabla serán visibles: **<TABLE RULES="valor"> ... </TABLE>** donde **valor** puede ser:

- **none** - Ninguna línea de división (por defecto).
- **groups** - Sólo aparecen líneas de división entre grupos de filas y grupos de columnas.
- **cols** - Sólo aparecerán líneas de división entre filas.
- **rows** - Sólo aparece líneas de división entre columnas.

Estos últimos atributos no funcionan igual en todos los navegadores, y no funcionan en absoluto si no son de la última generación. Según el navegador de que se trate, puede que haya que combinar más de un atributo para conseguir el efecto deseado. Lo mejor es hacer pruebas con algunas versiones para asegurarse.

Las posibilidades son tan amplias, que lo mejor es ver la página de ejemplos de tablas que encontrarás en el índice.