Unidad 3

CSS. Selectores básicos

”No se trata de donde vienes,
sino a donde vas”
Ella Fiztgerald

Las hojas de estilo sirven para separar completamente el contenido del diseño de una página web, de tal forma que, si lo estructuramos correctamente, es posible cambiar totalmente el aspecto de nuestra web haciendo modificaciones en las hojas de estilo (css) sin tocar ni una sóla línea de su contenido (html). Para empezar, podemos hacer una pequeña demostración con la página de ejemplo que hemos creado en la anterior unidad. Si añadimos la siguiente línea en la sección de cabecera (head), el navegador dejará de usar la hoja de estilos por defecto y tomará una de las que pone a nuestra disposición la W3C como demostración:

```
<link rel="stylesheet"
href="http://www.w3.org/StyleSheets/Core/Traditional" />
```

Podemos sustituir la hoja Traditional por otra de las siguientes: Midnight, Ultramarine, Chocolate, Oldstyle, Modernist, Steely o Swiss. Cada una de ellas nos proporcionará un estilo diferente.

Tenemos también un gran ejemplo de lo que perseguimos en W3School:

https://www.w3schools.com/css/css_intro.asp

Integración de HTML con CSS

Existen tres formas diferentes de integrar las hojas de estilo con HTML. La más rudimentaria consiste en usar el atributo style junto con cualquier etiqueta HTML. Por ejemplo:

```
<h1 style="color: red; font-family: Verdana, Arial;" >Título</h1>
```

La segunda forma sería introducir una nueva sección dentro de la cabecera (head) del documento HTML donde se insertarían los diferentes estilos:

```
<style type="text/css">
     h1 { color: red; font-family: Verdana, Arial; font-size: large; }
     p { color: gray; font-family: Times; font-size: medium; }
</style>
```

La tercera y ultima sería indicar, también en la sección head de nuestro documento HTML, la ruta a un archivo externo donde se incluirían los diferentes estilos. Ya hemos visto un ejemplo antes cuando tomábamos los ejemplos de la W3C. Uno propio podría ser así:

```
<link rel="stylesheet" href="estilos.css" />
```

En la sentencia anterior, en la etiqueta link podemos incluir un atributo denominado media que nos permitirá tener hojas de estilo diferentes según el medio donde se visualizará la página HTML. Los valores maś habituales de este atributo son screen, print, o all, que es el valor por defecto.

```
<link rel="stylesheet" href="estilos.css" media="print"/>
```

Dentro del archivo estilos.css escribiremos nuestros estilos con una sintaxis idéntica a la vista en el caso anterior:

```
h1 { color: red; font-family: Verdana, Arial; font-size: large; }
p { color: gray; font-family: Times; font-size: medium; }
```

Un documento HTML puede utilizar una de las tres formas anteriores, dos de ellas o las tres, incluso, a la hora de aplicar estilos. El navegador, en caso de conflicto, tomará siempre como bueno el estilo más cercano al contenido del documento. En caso de conflicto usando la misma forma de aplicar estilos el navegador tendrá en cuenta lo definido en la última ocurrencia aparecida.

Comentarios en hojas de estilo CSS

Los comentarios en las hojas de estilo (cuando estas se guardan en ficheros independientes) no se hacen de la misma forma que en HTML, sino que usamos los símbolos /* para abiri el comentario y */ para cerrarlo. Un comentario puede ocupar más de una línea. Un ejemplo:

```
/* Esto es un comentario
   y puede ocupar tantas líneas como deseemos
   hasta encontrar el símbolo de cierre de comentario */
```

Selectores básicos

Toda regla de formato en CSS está formadas por dos partes: selector y declaración. La declaración indica que formato hay que aplicar y el selector indica donde hay que aplicarlo. La declaración de la regla CSS va encerrada entre llaves y está formada por parejas propiedad: valor; terminadas siempre en punto y coma. Si observamos alguno de los ejemplos anteriores es fácil identificar cada uno de estos componentes y comprobar lo sencillo de la sintaxis.

Así como en HTML la palabra clave es la etiqueta, el corazón de CSS lo componen los selectores. La versión 2.1 de CSS tiene una docena de selectores diferentes pero es posible diseñar casi cualquier web con los cinco tipos básicos que veremos a continuación. Veremos los avanzados más adelante.

Selectores de tipo o de etiqueta

El selector de tipo o de etiqueta sirve para definir las propiedades de formato de todos los elementos de la página cuya etiqueta HTML coincida con el valor del selector. Los dos ejemplos que hemos visto en el punto 2 en los que se definía el formato para h1 y p son ejemplos de selectores de este tipo. Podemos definir selectores de etiqueta para cualquier elemento HTML que esté dentro del cuerpo (body) de nuestra página HTML e, incluso, para el propio body:

```
body {
        font-family: Courier, Monospace;
        color: red;
        text-align: justify; }
```

Podemos aplicar estilos simultáneamente a más de una etiqueta separando los selectores con comas:

```
        h1, h2, h3, h4, h5, h6 {color: blue; }
```

Observa que si cometes un error en la sintaxis, el navegador sencillamente no tendrá en cuenta la declaración de ese selector.

Selector universal

Existe un selector denominado universal que se simboliza con un asterisco. Se trata de una especie de comodín que permite seleccionar a cualquer elemento:

```
* {font-family: Courier, Monospace; color: red; text-align: justify; }
```

Cuidado: así aplicado sería similar, pero no igual, a aplicar la misma declaración sobre la etiqueta body, como hemos visto antes. Lo mejor para que aprecies las diferencias es que practiques sobre algunos ejemplos.

Selector descendiente

Es una forma de seleccionar elementos que se encuentran dentro de otros elementos para aplicarles un estilo personalizado. Imaginemos, por ejemplo, que quiero aplicar un estilo particular a la etiqueta de cursivas (em) pero sólo cuando esta se aplica a un titular. Necesito usar un selector descendiente:

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

El estilo se aplicará a todas las etiquetas em dentro de un elemento h1 sin importar el nivel de profundidad. Es decir, en el siguiente fragmento HTML también se aplicarían:

```
<h1><a id="INICIO">Titular de la <em>pagina</em></a></h1>
```

El selector descendiente debe de estar formado por, al menos, dos selectores pero podría tener más si así lo precisamos:

```
p strong em {font-variant: small-caps; }
```

El estilo anterior se aplicaría a las cursivas (em), dentro de una etiqueta de negrita (strong) y, a su vez, dentro de un párrafo (p).

Podemos combinar este tipo de selectores con el selector universal para realizar restricciones. Por ejemplo, tengamos en cuenta las siguientes dos variantes de selectores y los dos fragmentos de código HTML a continuación:

```
p a { color: green; }
p * a { color: green; }

<p><a id="INICIO">Enlace</a></p>
<p><strong><a href="#">Enlace</a></strong></p>
```

Si usamos la primera regla CSS se colorearán en verde las dos líneas, mientras que si usamos la segunda regla, sólo se coloreará la segunda.

Selectores de clase

Los selectores que hemos visto hasta ahora fallan en una cosa: no son flexibles cuando queremos aplicar un estilo diferenciado a algunos párrafos, pero no a todos ellos. Una de las soluciones para esto es usar los selectores de clase. Para aplicarlos, lo primero que tenemos que hacer es usar el atributo class en las etiquetas HTML correspondientes a los párrafos cuyos estilos queremos diferenciar:

```
<p class="verde">Este párrafo debe de ir en verde.</p>
<p>Este párrafo va en color normal.</p>
```

La regla CSS se especifica anteponiendo un punto al valor indicado en el atributo class:

```
.verde {color: green; }
```

La regla CSS se aplicará a cualquier etiqueta HTML que incluya un atributo class cuyo valor sea verde, y no sólo a las etiquetas p. Es decir, sería perfectamente válido que, además de los párrafos anteriores, tuvieramos algo así en nuestro HTML:

```
<p> Si aplicamos el atributo class con valor verde a <strong
class="verde">estas negritas</strong> o a <em
class="verde">estas cursivas</em> también aparecen en
verde.</p>
```

Si quisiéramos restringir el uso del selector de clase para que sólo sea válido en combinación de una determinada etiqueta, podemos combinarlo con el selector de tipo o etiqueta:

```
p.verde {color: green; }
```

Ojo, no debemos de confundir las tres diferentes formas de combinación de selectores que hemos visto hasta ahora:

```
p, .verde {color: green; }
p .verde  {color: green; }
p.verde   {color: green; }
```

En el primer caso la regla se aplica a cualquier etiqueta p o a cualquier etiqueta que tenga un atributo class con valor verde. En el segundo caso se aplica a cualquier etiqueta con un atributo class con valor verde que se encuentre dentro de un elemento de tipo p. En el último caso, sólo se aplica en las etiquetas de tipo p que tengan un atributo de tipo class con valor igual a verde.

Podemos aplicar también el formato de más de un selector de clase al mismo elemento HTML. Imaginemos las siguientes reglas CSS:

```
.codigo {font-family: monoespace; color: rojo; }
.versalita {font-variant: small-caps;}
```

Y en nuestra página HTML:

```
<p class="codigo versalita">Párrafo en rojo con fuentes
monoespaciadas y usando versalitas</p>
```

Por último, podemos usar también los selectores de clase múltiple. La regla CSS se expresa así:

```
.codigo.versalita {font-family: monoespace; color: blue; font-variant:
small-caps;}
```

Si incluimos esta regla en nuestro ejemplo anterior, tendrá preferencia sobre las reglas individuales y nuestro párrafo aparecerá en azul en lugar de en rojo.

Selectores de ID

El selector de id es muy similar al de clase. La principal diferencia es que se debería de usar sólo cuando es preciso aplicar estilos a un elemento único en la página. En ese caso usaremos el atributo id en lugar de class en la etiqueta HTML

```
<p id="destacado">Parrafo destacado en negritas y color azul.</p>
```

y el simbolo almohadilla (#) en lugar del punto en la definición de la regla CSS.

```
#destacado {font-weight: bold; color: blue; }
```

Una nota muy importante: la restricción de usar la etiqueta id en un único elemento es algo que nosotros debemos de imponernos pero que, si la obviamos y la ponemos en más de un sitio el navegador no se quejará e, incluso, aplicará correctamente los estilos indicados en todas las ocurrencias de la misma. No obstante, no podríamos considerar como correcto de forma estricta el código así escrito y, por tanto, deberíamos de evitarlo.

Casi todo lo demás dicho en referencia a los selectores de clase (la forma de restringir su uso, de combinarlo con otros selectores, etc) es aplicable a los selectores de id con una salvedad: no existen selectores de id múltiple y no podemos poner dos id diferentes a una misma etiqueta HTML al igual que hacíamos con el selector de clase.

Colores y propiedades básicas del texto

Los colores en la declaración de cualquier selector de CSS se definen con dos propiedades: color (del texto o del elemento en si) y background-color (del fondo). Por ejemplo:

```
body {color: white; background-color: green; }
```

Los valores por defecto son black para el color del elemento y transparent para el fondo. Podemos aplicar estas propiedades en la declaración de cualquier selector de los ya vistos y los que veremos en adelante:

```
hr {color: red;}
```

¿Qué colores son válidos? Por un lado tenemos la posibilidad de usar una paleta de colores básicos usando su nombre predefinido de entre los que aparecen en la siguiente lista:

https://en.wikipedia.org/wiki/Web_colors#Extended_colors

Una segunda opción es indicar el color exacto que queremos mediante su valor RGB, ya sea en hexadecimal, decimal o porcentual:

```
h1 { color: rgb(255, 20, 147); }
h2 { color: rgb(100%, 8%, 58%); }
h3 { color: #FF1493; }
```

Ciertos colores expresados en RGB hexadecimal se pueden presentar mediante una forma abreviada. Son aquellos en los que los tres componentes de color tienen sus cifras iguales, por ejemplo #660099 puede expresarse también como #609. Un subconjunto formado por 216 de estos colores constituyen lo que se conoce como paleta de colores Web Safe y se usan cuando queremos garantizar que los colores que escojamos se verán exactamente igual en todos los dispositivos, incluidos aquellos que sólo pueden visualizar 256 colores (muchos ordenadores y, sobre todo, dispositivos móviles antiguos). Puedes consultar esta paleta en este enlace:

http://en.wikipedia.org/wiki/Web_colors#Web-safe_colors

Mediante la propiedad font-family elegimos la tipografía que queremos usar. Podemos indicar, el tipo de letra concreta que queremos o la familia tipográfica. Por ejemplo:

```
p {font-family: Arial; }
strong {font-family: serif; }
```

Si el tipo de letra consta de más de una palabra tenemos que indicarlo entre comillas dobles:

```
em {font-family: “Times New Roman”;}
```

Las familias tipográficas genéricas admitidas son serif, sans-serif, monospace, cursive, o fantasy. Las tres primeros son las únicas que se usan habitualmente.

El tipo de letra así indicado de cualquiera de estas formas debe de estar instalado en el ordenador o ser accesible a través de Internet (y, en ese caso, indicar donde) para que se muestre correctamente, por eso es más seguro indicar una familia tipográfica o, mejor aún, una lista de preferencias que siempre acaba en una familia tipográfica. Por ejemplo:

```
h1 {font-family: Verdana, Helvetica, Arial sans-serif; }
p { font-family: Georgia, "Times New Roman", Times, serif; }
pre {font-family: "Courier New", Courier, monospace; }
```

Otra posibilidad es escoger una fuente disponible en Internet e indicar el lugar donde está disponible para que el navegador se la descargue y la use. El directorio más popular de fuentes es el que Google pone a nuestra disposición:

http://www.google.com/webfonts

El funcionamiento es sencillo: elegimos las fuentes que vamos a usar y Google nos proporciona la línea que tenemos que incluir en el head de nuestro HTML y la pareja propiedad:valor que debemos de usar en nuestras reglas CSS. Por ejemplo, para usar la fuente Gochi Hand debemos incluir esto en la sección head de nuestro HTML:

```
<link href="http://fonts.googleapis.com/css?family=Gochi+Hand"
rel="stylesheet" type="text/css">
```

Y nuestra regla CSS sería así:

```
h1 {font-family: "Gochi Hand", cursive;}
```

La propiedad font-weight nos permitirá cambiar el grosor del trazo de la fuente. Tenemos dos opciones para ello: especificar este mediante una palabra clave predefinida (lighter, normal, bold o bolder) o usar un valor numérico también de entre una lista de posibilidades (100, 200, 300, 400, 500, 600, 700, 800 o 900) donde el 400 se corresponde con normal y el 700 con bold. Sólo tenemos garantía de que funcionen normal y bold. El resto dependerá de la forma en que esté diseñada nuestra tipografía.

La propiedad font-style nos permite elegir entre normal, italic u oblique. La diferencia entre italic y oblique es que la italic debe de existir dentro del diseño tipografíco de la fuente (y si no no se muestra) mientras que la oblique se consigue forzando una inclinación de entre 8 y 12 grados la fuente regular. Siempre que exista esa posibildad obtendremos resultados de mejor calidad con italic.

font-variant es una propiedad que nos permite dos únicas opciones: normal o small-caps (lo que en español conocemos como versales o versalitas).

```
p {font-weight: bold; font-style: italic; font-variant: small-caps; }
```

La justificación del texto la realizamos mediante la propiedad text-align cuyos valores posibles son left, right, center o justify. text-align con el valor center es el sustituto de la etiqueta center cuyo uso está desaconsejado.

```
h1 {text-align: center; }
```

Esta propiedad no sólo alinea el texto, sino que hace lo mismo con las imágenes que estén contenidas en un texto cuyo párrafo la tenga aplicada. Por ejemplo, si queremos una imagen alineada no podemos aplicarla sobre la etiqueta img, pero si sobre la etiqueta p del párrafo que contiene a esa imagen:

```
p {text-align: center; }
```

Y luego, meter la imagen dentro de un párrafo:

```
<p><img src="logotipo.jpg" alt="Logo de mi empresa" /></p>
```

Dos propiedades más antes de terminar este punto: text-decoration nos permite cinco efectos adicionales sobre nuestro texto: none, underline, overline, line-throught y blink. La propiedad text-transform tiene cuatro valores posibles: none, capitalize, uppercase y lowercase.

Existen más propiedades referentes al texto La propiedad font-size, por ejemplo, es la que nos permite cambiar el tamaño de las fuentes. También tenemos las relativas al interlineado y alguna más. Las veremos más adelante cuando hayamos hablado de las diferentes unidades de medida que podemos usar.

Combinando selectores

Todos los selectores aquí vistos pueden combinarse entre si. Algunos ejemplos ya los hemos visto en este documento. Pero podemos complicarlo cuanto queramos o necesitemos. Veamos algunos ejemplos:

```
.aviso .especial { color: red; background-color: yellow; }
```

El anterior selector solamente selecciona aquellos elementos con un class="especial" que se encuentren dentro de cualquier elemento con un class="aviso", es decir, estamos aplicando simultaneamente selectores de clase y selectores descendientes.

```
p.aviso strong.especial { color: blue; }
```

Ahora, el selector solamente selecciona aquellos elementos de tipo con un atributo class="especial" que estén dentro de cualquier elemento de tipo <p> que tenga un atributo class="aviso". O sea, estamos aplicando selectores descendientes junto con selectores de clase que tienen su uso restringido a ciertas etiquetas.

```
ul#menu li.destacado a#inicio { color: pink; }
```

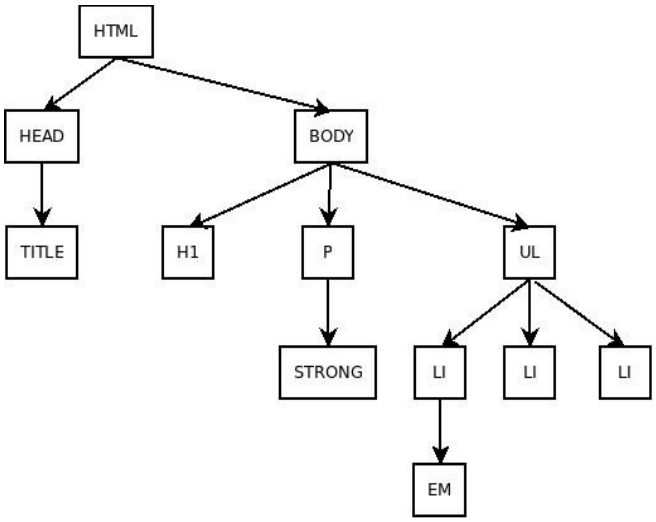
El anterior selector hace referencia al enlace con un atributo id=”inicio” que se encuentra dentro de un elemento de tipo <li> con un atributo class=”destacado”, que forma parte de una lista <ul> con un atributo id=”menu”. De nuevo selectores descendientes pero en esta ocasión con tres niveles y en cada uno de ellos tenemos un selector de clase o de ID con uso restringido.

Herencia

Podemos expresar cualquier documento HTML mediante una estructura en forma de árbol. Por ejemplo, tengamos el siguiente HTML sencillo (en el que, intencionadamente, hemos suprimido algunas de las etiquetas y atributos para simplificarlo):

```
<html>
   <head>
      <title>Mi web</title>
   </head>
   <body>
      <h1>La web del profe de Lenguaje de Marcas</h1>
      <p>José María Morales es el <strong>profesor</strong>
       de este año de la asignatura y, entre otras cosas, vamos
       a ver con él:</p>
      <ul>
         <li>HTML y <em>XHTML</em></li>
         <li>CSS</li>
         <li>XML</li>
      </ul>
   </body>
</html>
```

Podríamos ver la estructura de nuestro documento HTML esquematizada como el siguiente árbol:



Pues bien, la herencia de CSS hace que las propiedades de los diferentes elementos se traslade siempre hacia abajo a través del árbol salvo que dichas propiedades sean reescritas, bien por los estilos por defecto, bien por una regla apropiada que hayamos introducido nosotros.

Por ejemplo, el elemento con la etiqueta strong aparecerá en negrita, no porque nosotros lo hayamos especificado así, sino porque así está definido por defecto y aunque definamos la propiedad font-weight: normal para el selector body seguirán siendo negritas.

Sin embargo, si definimos la propiedad color: orange; y background-color: black en el selector body, todo aparecerá con fondo negro y letras naranjas.

Si nuestra página web sencilla llevara un enlace (etiqueta a) el fondo si aparecería negro, pero el color sería azul porque esa etiqueta si tiene la propiedad color sobre escrita.

En definitiva: las propiedades de estilo se heredan siempre hacía abajo en el árbol salvo que dicha propiedad sea reescrita, bien explícitamente, bien con una regla por defecto.

Unidades de medida y otras propiedades

CSS tiene un amplio y flexible conjunto de formas de expresar tamaños y medidas. Manejarlas correctamente es necesario para expresar el tamaño de las fuentes que usamos y para otras características esenciales en un buen diseño: interlineado, márgenes, separaciones, etc.

Para experimentar con ellas presentaremos una nueva propiedad: font-size que nos permite definir el tamaño de la tipografía que estamos usando. Algunas de las formas de especificar medidas que veremos aquí son sólo válidas para los tipos de letras mientras que otras son más generales y nos servirán para cualquier otra cosa.

Unidades de medida específicas de las tipografías

La forma más sencilla es expresar esta medida como una palabra clave que exprese un valor fijo absoluto. Los valores permitidos son xx-small, x-small, small, medium, large, x-large y xx-large.

```
h1 {font-size: xx-large;}
p#piedepagina {font-size: x-small;}
```

La segunda forma de expresar el tamaño de la tipografía sería mediante una palabra clave que expresa un tamaño relativo. Dos son las palabras claves permitidas en este caso: larger y smaller. El primer valor elevará la fuente sobre lo que le correspondería por defecto mientras que el segundo lo reducirá.

```
p {font-size: medium;}
p.grande {font-size: larger;}
```

Unidades absolutas

Podemos expresar medidas absolutas en pulgadas (in), centímetros (cm), milímetros (mm), puntos (pt) o picas (pc). Estas dos últimas son medidas específicas de las imprentas y medios gráficos. Un punto equivale a 1/72 pulgadas (aproximadamente 0,35 milímetros) y una pica equivale a 12 puntos (alrededor de 4,23 milímetros). El punto también es conocido porque es la medida que suele usarse en los editores de texto para elegir el tamaño de una fuente.

Cuatro notas a tener en cuenta en todas ellas:

- Para separar las cifras decimales usaremos el punto que es la norma en las medidas anglosajonas.
- Si la parte entera de una medida es cero se puede suprimir.

- Si la medida es cero no se tiene que poner unidad de medida. Un cero es 0 y basta.
- Entre la magnitud y la unidad de medida no debe de haber nunca espacios en blanco.

Las siguientes reglas serían correctas:

```
h1 {font-size: .5in;}
p {width: 25cm; font-size: 8.5pt;}
p.destacado {width: 100mm; }
td {width: 10pc; }
```

Unidades relativas

Las unidades relativas son mucho más flexibles que las absolutas. Además, es altamente recomendable usarlas siempre que se pueda porque ante la gran diversidad de dispositivos que se pueden usar a la hora de ver una página web (portátiles, grandes televisores de plasma, tablets, teléfonos móviles...), estas son las que mejor se adaptan a cualquier situación.

Los píxeles son la medida relativa más conocida. Se trata de una medida relativa a las dimensiones del dispositivo donde estamos viendo la página web. Un ejemplo:

```
hr {width: 500px;}
```

em (ojo: no confundirlo con la etiqueta em de HTML) y ex son medidas relativas a las dimensiones de la tipografía que estamos usando. Son medidas muy conocidas por los profesionales de la tipografía. 1em representa, aproximadamente, el ancho de la letra M mayúscula que estamos usando teniendo en cuenta tanto la tipografía como el tamaño de la misma: no es lo mismo una m si estamos usando una Arial Narrow que si estamos usando una Courier New. Igualmente, para cualquiera de ambas el tamaño varía si estamos usando una fuente a 12 puntos o a 18.

1ex equivale a la altura de una letra x minúscula. Aunque no se trata de una regla exacta, 1em suele aproximarse como el tamaño de la fuente que estamos usando (es decir, si estamos usando una fuente a 12 puntos, 1em equivale a 12 puntos).

```
body {font-size: .9em;}
```

Si consideramos cierta la aproximación que hemos dicho antes, podemos decir que una fuente con un tamaño de .9em es, aproximadamente, el 90% de la fuente normal que debería de tener esa etiqueta. Una con 1.5em sería del 150%

```
p {font-size: 1.5ex;}
```

También se suele aproximar, aunque esto es bastante más irregular, que 1ex equivale aproximadamente a 0.5 em

Aunque a primera vista son medidas que parece que sólo tienen sentido aplicadas a tipos de letra, podemos usarlas sobre cualquier elemento de nuestra web:

```
hr {width: 1em;}
```

Y una nota importante: la "referencia" sobre la que se calcula el tamaño relativo de un elemento cuando usamos estas medidas es siempre es el tamaño de letra de su elemento "padre", es decir, del elemento en el que se encuentra. Si el elemento no se encuentra dentro de ningún otro elemento, la referencia es el tamaño de letra del elemento body. Si no se indica de forma explícita un valor para el tamaño de letra del elemento body, la referencia es el tamaño de letra por defecto del navegador. Lo podemos ver con este ejemplo:

```
body { font-size: 10px; }
h1 { font-size: 1em; font-weight: normal;}
```

Al aplicar esto a nuestra web, definimos el tipo de letra base a 10 píxeles y el tamaño de los titulares de primer orden a 1em que, como

hemos dicho, equivale aproximadamente al tamaño de la letra base, es decir, que no habría diferencia entre los titulares de primer orden y el texto normal. Otro ejemplo:

```
body { font-size: 12px; }
strong { font-size: 1.2em;}
```

Con esto, todas las negritas serían, aproximadamente, un 20% más grandes sobre el tamaño del texto en el que se colocan.

Existe otra unidad que es el rem. Es similar al em pero en este caso se encuentre donde se encuentre el tamaño está referido de forma relativa al ancho de una letra m situada en el elemento raíz (body) del documento html.

```
strong { font-size: 1.2rem;}
```

No existe una unidad rex.

Porcentajes

El porcentaje es otra forma de expresar una medida usada ampliamente en CSS que, además, también se considera una unidad relativa. Se pueden usar para expresar el tamaño de una fuente:

```
p {font-size: medium;}
p.grande {font-size: 200%;}
```

Cuando aplicamos un porcentaje como una unidad de medida general, este se refiere al espacio que el objeto debería de ocupar. Por ejemplo, una linea horizontal ocupa normalmente el ancho de la pantalla completa. Si aplicamos la siguiente regla CSS ocupará solamente la mitad:

```
hr {width: 50%;}
```

La W3C recomienda encarecidamente usar píxeles y porcentajes como unidades de medida para especificar el tamaño de los diferentes objetos y em como unidad para especificar el tamaño de las fuentes. Cuando se

prepara un documento HTML cuya finalidad es tener salida impresa en papel y no en un dispositivo óptico, es mejor el uso de puntos y picas.

Más propiedades para el texto

Veremos a continuación algunas propiedades aplicables al texto de nuestros documentos que aún no conocemos.

La propiedad line-height ajusta el interlineado del texto. El valor normal es el que debería de tener por defecto. Podemos variar este poniendo un porcentaje, un número o una medida en valor absoluto o relativo. Las siguientes reglas CSS serían prácticamente equivalentes:

```
p { line-height: 1.5;}
p { line-height: 1.5em;}
p { line-height: 150%;}
```

La propiedad **text-indent** nos permite que la primera línea de cada párrafo aparezca desplazada hacía la derecha respecto al resto del párrafo. Se puede usar una medida (por ejemplo, 2.5em) o un porcentaje relativo al ancho total de la línea del párrafo (un 50% haría que la primera línea empezara justo en su mitad). Admite valores negativos que harían que la primera línea del párrafo empiece más a la izquierda del mismo.

letter-spacing y word-spacing son dos propiedades que nos sirven para controlar el espacio en blanco que separa cada dos letras o cada dos palabras, respectivamente. Los posibles valores son normal o una medida, representando esta el espacio adicional que se suma o resta (admite valores negativos) al espaciado por defecto.

```
h1.juntito { letter-spacing: -.2em; }
p.separado { word-spacing: .5em; }
```

La propiedad **white-space** controla la forma en que se trataran los espacios en blanco, los saltos de línea y el ajuste de las mismas. Los posibles valores son normal, pre, nowrap, pre-wrap y pre-line. normal y

pre son las más usadas. La primera representa el comportamiento normal del navegador (sólo se respeta el primer espacio en blanco y ningún salto de línea pero estas se ajustan) y pre el que obtendríamos usando la etique pre de HTML (se respetan todos los espacios en blanco y saltos de línea que hagamos pero las líneas no se ajustan y si son muy largas desbordan el espacio asignado). Las tres restantes son variantes de estas tres posibilidades. no-wrap seria como normal pero sin ajustar las líneas, pre-wrap sería como pre pero ajustando las líneas y, por último, pre-line sería como pre pero sin respetar los espacios en blanco.

La propiedad **vertical-align** nos ayuda a alinear elementos de diferentes alturas situados en la misma línea. La mejor forma de verlo es colocando una pequeña imagen junto a una línea de texto y hacer pruebas con los diferentes valores posibles. Supongamos la siguiente línea de HTML:

```
<p><img src="http://moodle.uponaday.net/logo.jpg" alt="FP" />
IES Francisco de Goya</p>
```

Como observamos, texto e imagen se alinean por la base que es el valor por defecto. Sería equivalente a usar el valor baseline para esta propiedad. Creemos ahora la siguientes reglas CSS:

```
img {border:1px solid; vertical-align: text-bottom;}
```

La diferencia entre el valor text-bottom y baseline es prácticamente inapreciable: apenas veremos que la línea de texto asciende un poco. El primero alinea la parte inferior de la imagen con la parte más baja de las letras, mientras que baseline realiza la alineación con la línea base de escritura de estas. middle es uno de los valores más corrientes y alinea el centro de la imagen con el centro de la línea de texto, mientras que text-top alinearía la parte superior de ambos elementos. top y bottom son dos atributos similares a, respectivamente, text-top y text-bottom pero más orientados a cuando queremos alinear diversas imágenes de diferentes

alturas. sub y super son dos valores raramente usados. Alínearian la base de la imagen con la línea base de los subíndices o de los superíndices del texto, respectivamente. Por último, podemos poner una medida o un porcentaje. La medida puede ser positiva o negativa y representa la distancia desde la base de la imagen con la base de la línea de texto (medidas negativas harían descender la línea de texto). El porcentaje tiene el mismo efecto correspondiendo el 100% con el alto de la línea de texto . Con una tipografía de 12 puntos, un desplazamiento de -100% o de -12pt debería de ser prácticamente equivalente.

Aplicar estilos a las listas

Los estilos se aplican a las listas mediante tres propiedades: list-style-type, list-style-image y list-style-position.

list-style-type define el tipo de "encabezado" de cada uno de los items de la lista. circle, disc, square o none son algunos de los más usados, pero también existen otros muchos que puedes consultar aquí:

http://www.w3schools.com/cssref/pr_list-style-type.asp

list-style-image nos permite usar una imagen de nuestra elección como encabezado de cada item en lugar de uno de los predefinidos con la propiedad anterior. Como siempre en estos casos hay que especificar la URL de acceso a la imagen. Si se especifica junto con list-style-type, esta segunda sólo se tendrá en cuenta si la imagen que especificamos no es accesible.

list-style-position es una de esas propiedades de "ajuste fino" del formato y se usa para especificiar si el marcador de principio del item debería de aparecer dentro (inside) o fuera a la izquierda (outside) de la caja en la que debería de aparecer cada item de la lista. El valor por defecto es outside.

Existe una propiedad abreviada llamada list-style que permite definir a las tres anteriores en una sóla regla. Las siguientes declaraciones, por tanto, serían equivalentes:

```
ul.milista {
       list-style-type: square;
       list-style-image: url(redicon.jpg);
       list-style-position: inside; }

ul.milista2 { list-style: square url(redicon.jpg) inside; }
```

Existen muchas otras propiedades CSS que admiten reglas abreviadas que engloban a varias otras relacionadas. Algunas las iremos viendo más adelante y otras son fáciles de identificar en los ejemplos.

Aplicando estilos a las tablas

La propiedad **border** (que se puede aplicar a muchos elementos además de a las tablas está formada por otras tres propiedas que se pueden aplicar de forma individual o combinada (al igual que ocurría ocn list-style):

```
       border: 5px solid red;
```

o

```
       border-width:4px;
       border-style:dotted;
       border-color:#BA7854;
```

Nada que apuntar sobre el ancho o el color que no hayamos visto ya. Algunos de los estilos válidos son, además de solid y dotted, double, dashed, groove, ridge, inset y ouset.

Podemos aplicar diferentes propiedades a los bordes así:

```
border-color: red cyan blue green; /* arriba derecha abajo izquierd */
border-color: red cyan blue; /* arriba derecha-izquierda abajo */
border-color: red cyan; /* arriba-abajo derecha-izquierda */
```

También citando la dimensión concreta (**border-top-color, border-left-color**, etc.)

Los bordes se pueden aplicar a las tablas (table) y a las celdas (td). A las filas no

Las propiedades **border-collapse** y **border-spacing** controlan la forma en que vemos los bordes de una tabla. La primera tiene dos posibles valores: collapse y separate. En el primer caso los bordes adyacentes se unirán en uno sólo siempre que sea posible mientras que con el segundo valor la separación entre bordes se controlará mediante la propiedad border-spacing. El valor por defecto es separate.

border-spacing indica el espacio que habrá entre los bordes de celdas adyacentes, por ejemplo:

```
border-spacing: 2px 0;
```

Si se indican dos valores el primero hace referencia a la separación horizontal y el segundo a la vertical. En el caso de indicar un sólo valor se aplica a ambas distancias

Por último, la propiedad **empty-cells** tiene dos posibles valores (show o hide) controla que se vean o no los bordes y fondo de las celdas vacías de una tabla. El valor por defecto es show.

Cambiar el aspecto del puntero del ratón

Podemos cambiar el aspecto del cursor cuando pasamos sobre un elemento con la propiedad cursor. Los valores posibles y su apariencia son:

- auto. El navegador determina por sí mismo el cursor según el contexto.
- crosshair. El cursor muestra una cruz.

- default. El cursor por defecto del sistema operativo, a menudo una flecha.
- e-resize. Cursor apuntando hacia el este.
- ne-resize. Cursor apuntando hacia el noreste.
- nw-resize. Cursor apuntando hacia el noroeste.
- n-resize. Cursor apuntando hacia el norte.
- se-resize. Cursor apuntando hacia el sureste.
- sw-resize. Cursor apuntando hacia el suroeste.
- s-resize. Cursor apuntando hacia el sur.
- w-resize. Cursor apuntando hacia el oeste.
- help. El cursor indica una ayuda. A menudo se muestra un signo de interrogación.
- move. El cursor indica un objeto que se puede desplazar.
- pointer. El cursor presenta un dedo que indica un enlace.
- progress. El cursor muestra una flecha con un reloj de arena.
- text. El cursor indica que es posible seleccionar el texto.
- wait. El cursor indica una progresión. A menudo se muestra un reloj de arena.
- url. Especifica un archivo donde se encuentra la imagen que se desea usar como cursor. El archivo de imagen especificado en la URL debe tener el formato cur (cursor) o ani (cursor animado).

Un ejemplo:

```
.caja { width: 50%;
```

```
  padding: 5px;
border: 1px dashed black;
  cursor: crosshair;}
```