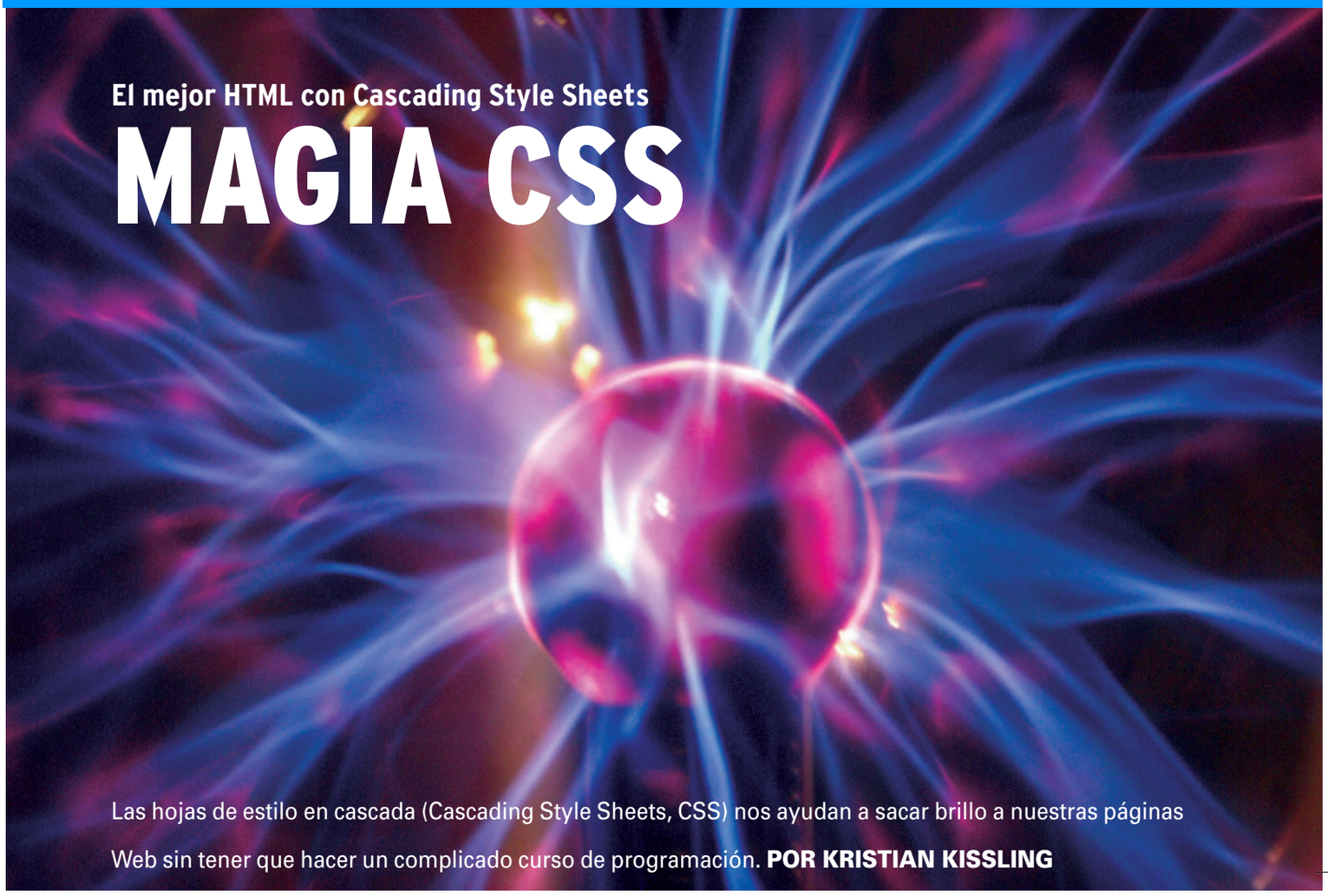

El mejor HTML con Cascading Style Sheets

MAGIA CSS

Las hojas de estilo en cascada (Cascading Style Sheets, CSS) nos ayudan a sacar brillo a nuestras páginas Web sin tener que hacer un complicado curso de programación. **POR KRISTIAN KISSLING**

A principios de los 90, lo único que se esperaba del HTML era que renderizara texto y proporcionara enlaces. Las funcionalidades de diseño llegaron más tarde debido a los lobbies de la industria. La consecuencia es lo que tenemos hoy en día: tablas con columnas increíblemente largas y definiciones de tipos de letra chapuceras. El diseño para un texto de 100 caracteres puede requerir el doble de código HTML.

Listado 1: Ejemplo de HTML

```
01 <p>
02 <b>Red</b> is a beautiful
   color. <a
   href="http://www.example.org">
   Sentences like this</a> are
03 <br> often used by Web
   Designers to demonstrate their
   layouts.<br> The content
   should be pointless
04 <b>pointless</b>, to avoid
   distracting the observer <br>
   from the layout.
05 </p>
06 These b-Tags are
   <b>outside</b> the p-Tags.
```

Para resolver el problema, el World Wide Web Consortium (W3C)[1] aprobó en 1996 la primera versión de las hojas de estilo en cascada (CSS). Las CSS proporcionan un medio flexible para definir elementos de estilo. Podemos usar CSS para conseguir un control más granular y eficiente de nuestros diseños Web. Nos permiten definir un diseño para cada elemento HTML, incluso para una sola letra. Podemos cambiar el tamaño de cada elemento, crear un marco y añadir espacio. En este artículo daremos un repaso de la versión actual de CSS, la Version 2.1. Necesitaremos un cierto conocimiento de HTML para seguir este artículo.

Ahorra Esfuerzo, Ahorra Tiempo

Para escribir hojas de estilo en cascada, todo lo que realmente necesitamos es un editor de textos estándar. El editor de KDE, Kate, verifica automáticamente nuestra sintaxis. Aunque la verificación de ésta puede no ser estrictamente necesaria, es magnífico tenerla, teniendo en cuenta que CSS, en contraste con HTML, no funcionará si la sintáxis es errónea. No tenemos que ser programadores para trabajar con CSS.

Supongamos que tenemos una página HTML y que hemos definido un color para el enlace de texto en una etiqueta *<body>*.

```
<body link="#000000" ➥
alink="#CCCCCC" ➥
vlink="#666666">
```

Debido a que no podemos cambiar el color de un único enlace usando HTML, todos nuestros enlaces se mostrarán de ese color. Con CSS podemos, sin embargo, manipular los enlaces individualmente de manera sencilla:

```
<a href=➥
"http://www.linuxmagazine.com"➥
style="color: white; ➥
text-decoration: none;">
```

El elemento *style* contiene las propiedades de diseño para el enlace. Éste ahora será blanco (color: white;) y sin subrayado (text-decoration: none).

Veamos otro ejemplo. Imaginemos que tenemos varias páginas de texto llenas

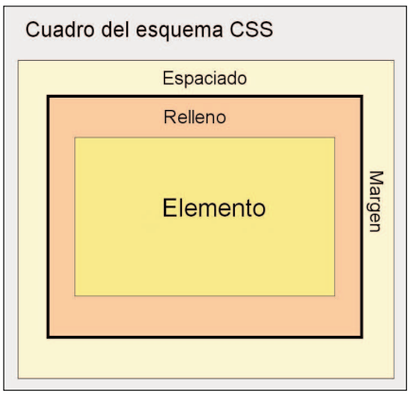

Figura 1: Los parámetros de anchura y altura fijan el tamaño y anchura del elemento en sí mismo. Para calcular la anchura y altura del cuadro completo, añadimos los valores de relleno, bordes y margen, y los duplicamos.

de citas textuales (*blockquote*), y queremos cambiar el tipo de letra para resaltar las citas. Para ello, normalmente abriríamos todas nuestras páginas HTML y usaríamos la herramienta de búsqueda y reemplazamiento para hacer los cambios. Por supuesto, este método no es muy eficiente. Con CSS, sólo tendríamos que modificar una sola línea en el archivo central de CSS.

```
blockquote ➥
{font-family: Arial, ➥
Helvetica, Courier;}
```

Todos los archivos HTML que usen la hoja de estilo central pasan a usar uno de los tres tipos de letra especificados para las citas textuales. CSS puede ahorrarnos un montón de codificación manual. Al mismo tiempo, nos ofrece una herramienta para definir y modificar la apariencia de elementos recurrentes de manera centralizada. Podemos incluso crear una hoja de estilo alternativa que optimice nuestro HTML a la hora de imprimir. Si un usuario quisiera entonces imprimir una página, se habilita una hoja de estilo que elimina los menús y las imágenes del texto, escala los tipos de letra o los cambia.

Csszengarden [2] nos ofrece una buena idea de qué es lo que podemos hacer con una sosa página HTML. Los visitantes pueden diseñar sus propias hojas de estilo para tener un completo documento HTML. Los asombrosos resultados demuestran la capacidad gráfica de CSS. Dicho esto, necesitaremos un navegador bastante reciente para apreciar el poder de CSS 2.1. La tabla en [3] nos indica qué navegadores soportan las propiedades de las CSS.

¿Dentro o Fuera?

Existen tres métodos para añadir definiciones de estilos a nuestro propio HTML. Ya conocemos uno de ellos: podemos añadir las definiciones a las etiquetas del código fuente HTML.

```
$A href=➥
"www.linuxmagazine.com" ➥
style="color: #FFFFFF; ➥
text- decoration: none;"➥
This is a link $/A
```

El elemento *style* introduce el código CSS que sigue. Tenemos que poner las propiedades entre comillas y separarlas con signos de punto y coma. Un segundo método consiste en definir el diseño de la página en una etiqueta *$style* dentro de la cabecera de la página, de la manera siguiente:

```
<head>
  <title>
    My webpage
  </title>
    <style type="text/css">
    h1 {color: red;}
  </style>
</head>
```

Un tercer método es poner las hojas de estilo en un archivo separado. Esto nos ofrece la ventaja de integrar las hojas de estilo usando un solo enlace en la cabecera de la página HTML. En otras palabras, todas nuestras páginas HTML toman su diseño de un único archivo llamado *sheet1.css*.

```
<head>
<link rel="stylesheet" ➥
type="text/css" ➥
href="sheet1. css" ➥
title="test" media="screen">
</head>
```

No es necesario abrir cada archivo HTML para modificar la apariencia de la página Web. En su lugar, sólo tenemos que modificar el archivo CSS al que hace referencia la página.

La Vida en un Cuadro

¿Qué ocurre cuando aplicamos CSS a un único elemento de HTML? Por ejemplo,

Figura 2: Usamos un editor de textos como Kate, para componer rápidamente una página HTML estándar. Un enlace en la cabecera integra un archivo de hojas de estilo externo. El "body" es simple texto de muestra.

un trozo de texto, una imagen o incluso una sola letra. CSS crea un cuadro (véase la Figura 1) que comprende el contenido, el relleno interno, el borde y los márgenes exteriores. El borde en sí mismo tiene un efecto en el ancho total. El margen es la distancia entre el borde y el cuadro vecino. Podemos modificar el relleno, el margen y el borde de casi cualquier elemento.

El cálculo del ancho total de un cuadro requiere algo de matemáticas. Dado un elemento con un ancho de 12 píxeles, 3 de relleno, 1 de borde y 4 de margen, el ancho total del elemento serían 28 píxeles: 12 (contenido) + 6 píxeles (relleno izquierdo y derecho) + 2 píxeles (bordes izquierdo y derecho) + 8 (márgenes izquierdo y derecho) lo que nos da 28 píxeles en total. Con CSS, al contrario que con HTML, nosotros definimos los bordes, relleno y márgenes de manera individual para cada página. Para añadir un marco por encima de un elemento podríamos introducir border-top: *1px solid black;. padding-right: 10px;* lo cual modificaría el relleno del elemento a su derecha. CSS distingue así mismo entre elementos de bloque e incrustados. Los elementos de bloque crean párrafos, mientras que los elementos incrustados pueden ser parte de un elemento de bloque, pero no crean un párrafo. La etiqueta *$b* es un elemento incrustado típico. La etiqueta *display:*

Listado 2: Constructor de Contenedores

```
01 div.container_central {
02 position: fixed;
03 top: 0%; left: 30%;
04 background: #CCCCCC;
05 width: 400px;
06 height: 600px;
07 color: red;
08 border: 1px solid black;
09 padding: 20px;
10 overflow: scroll;
11 }
```

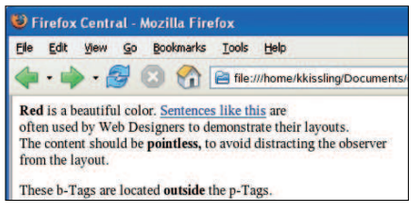

Figura 3: Página común y corriente. Sin una hoja de estilo o etiquetas de diseño HTML tendremos letras negras con los remarques en negrita. Los enlaces se subrayan en azul, tal y como se define en las preferencias.

Figura 4: Unas pocas instrucciones cambian la página por completo. Las secciones con las etiquetas b se muestran en rojo y con un tamaño de letra diferente.

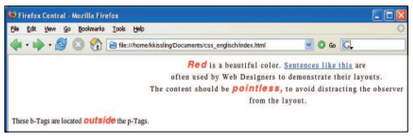

Figura 5: El diseño puede no llegar a los estándares de los diseñadores Web profesionales, pero es suficientemente bueno para demostrar cómo se puede modificar el tipo de letra.

block; puede convertir un elemento incrustado en uno de bloque. *display: inline;* hace justamente lo contrario.

Poner Bonito Nuestro HTML

Como las páginas Web básicamente transportan información en texto formateado, enfoquémonos en el diseño del texto sin CSS. Dejamos a elección del lector la decisión de guardar las instrucciones en CSS en un archivo separado o bien usar alguno de los otros métodos comentados. Nosotros hemos elegido un archivo externo para el siguiente ejemplo. En primer lugar creamos un archivo HTML con las típicas etiquetas como <*html*>, <*head*> y <*body*>, y las guardamos como *index.html* (véase la Figura 2). Ahora tecleamos el breve texto del Listado 1 como el <*body*> del archivo, incluyendo todas las etiquetas.

Abrimos un nuevo archivo con Kate y lo guardamos como *sheet1.css* en el mismo directorio. Podemos integrar la hoja de estilo usando un enlace a la cabecera de *index.html*:

```
<head>
<link rel="stylesheet" ➲
type="text/css" ➲
href="sheet1.css" ➲
title="sheet1">
</head>
```

Sin la información del diseño, el navegador mostraría el texto de *index.html* con letras grandes y negras, con los fragmentos etiquetados con < *b* > en negrita y los enlaces subrayados en azul (véase la Figura 3). Para modificar el tipo de letra, color y tamaño, debemos editar ahora el archivo *sheet1.css*:

```
01 b {
02   font-family: Arial,
  Helvetica,Courier;
```

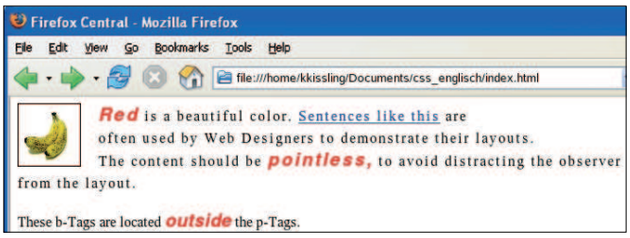

Figura 6: Bananas. La imagen está flotando en la zona superior izquierda. La distancia de 20 píxeles hasta el texto es cortesía del "margin-right".

Figura 7: Pedimos disculpas a los diseñadores gráficos por esto, pero al menos demuestra el método. Podemos asignar imágenes de fondo a la mayoría de los elementos HTML.

```
03   font-size: 130%;
04   color: red;
05   font-weight: bold;
06   font-style: italic;
07   }
08
09 p {
10   font-size: 16px;
11   }
```

Ahora, cuando abramos el archivo *index.html* en nuestro navegador, veremos la diferencia en las instrucciones del diseño que ha realizado la hoja de estilo (véase Figura 4). Podemos usar *font-family* para especificar qué fuente elegirá nuestro navegador para mostrar el texto. Los navegadores bajo Windows probablemente elegirán *Arial*, mientras que los de Mac usarán *Helvetica* y los Linux *Courier*. Si el navegador no tiene ninguno de estos tipos de letra instalados, el tipo de letra será el predeterminado. La propiedad *font-size* nos permite fijar el tamaño del tipo de letra. El navegador generalmente adopta un *font-size* del 100% del tipo de letra prefijado. Sin embargo, el código HTML tiene una etiqueta *< p >* con un tamaño de tipo de letra de 16 píxeles a continuación de la etiqueta *< b >*. El tamaño de tipo de letra se basa en el tamaño del elemento padre. Por lo tanto, si los 16 píxeles de la etiqueta *< p >* suponen el 100% para la etiqueta *< b >*, un *font-size* del 130% suponen 20.8 píxeles.

Los colores se cambian justamente igual que con HTML. Podemos especificarlos tanto con un nombre de color como con un valor hexadecimal como *#FFFFFF* para la propiedad *color*. La propiedad *font-weight* nos ayuda a habilitar un tipo de letra en negrita. Podemos seleccionar un valor para *bolder*, *lighter* o *normal*. La opciones *font-style* son *italic*, *normal* u *oblique*. Ahora añadimos cuatro líneas adicionales dentro de las llaves para la etiqueta *p*:

```
letter-spacing: 2px;
line-height: 25px;
text-align: center;
vertical-align: top;
```

El texto en el interior de la etiqueta *< p >* cambia apreciablemente (véase Figura 5). *letter-spacing* cambia el espacio entre letras y *line-height* cambia el espacio entre líneas. Las siguientes dos propiedades alinean el texto, *text-align* selecciona la posición horizontal del texto, que puede ser *center*, *right*, *left* o *justify*. Por su parte, *vertical-align* se encarga de la posición vertical. En nuestro ejemplo, el texto ajusta con *top* encima del bloque vecino. Las otras posiciones pueden ser *middle* y *bottom*.

Para Enmarcar

El punto más interesante acerca de CSS es que podemos diseñar cada elemento de manera individual. Por ejemplo, podemos añadir *border: 1px solid #000000;* al elemento *b* en nuestra hoja de estilo. Todo lo que etiquetemos con *< b >* se rodeará por un marco negro con una línea continua de un solo píxel. Podemos modificar el marco, por supuesto, cambiando el tipo de línea con puntos o rayas. Si sólo necesitamos un marco para una sola página, podemos especificar *border-left: 1px dashed red;* o *border-bottom: 1px dotted green;*.

La propiedad *float* nos ayuda con la posición de las imágenes. Creamos una imagen llamada *testfig.jpg* y la guardamos en el mismo directorio que nuestro archivo *index.html*. Podemos añadir ahora la imagen al archivo HTML con la etiqueta *< p >*:

```
<img src="testfig.jpg">
```

Para usar el efecto, en primer lugar justificamos el texto. Luego nos vamos a la hoja de estilo y borramos la propiedad *text-align: center;* para el elemento *p* sin olvidarnos de añadir una definición para manipular las imágenes:

```
img {
    width:50px;
    height:50px;
    float: left;
    margin-right: 20px;
    margin-bottom:0px;
}
```

float: left; posiciona la imagen a la izquierda. Los valores *width* y *height* escalan la imagen al tamaño requerido, y *margin-right: 20px;* añade un margen entre el borde derecho de la imagen y el texto (véase la Figura 6).

Los diseños más atractivos dependen a menudo de las imágenes de fondo. Creamos una imagen llamada *bgimage.jpg* en el directorio principal y modificamos las propiedades del elemento *p* en la hoja de estilo.

```
background-image: ➥
url(bgimage.jpg); ➥
background-position: center; ➥
background-position: top;
background-repeat: no-repeat; ➥
background-attachment: fixed;
```

Aparece una imagen de fondo detrás del texto (véase la Figura 7). Debido al parámetro *background-repeat: no-repeat;* la imagen sólo se muestra una vez. Las propiedades *background-position: center;* y *background-position: top;* centran el gráfico y lo ajustan al borde superior del elemento vecino, al mismo tiempo que lo mantiene visible. Con *background-attachment: fixed;* conseguimos que la imagen se quede clavada pase lo que pase. CSS nos permite añadir imágenes de fondo a una gran variedad de elementos, incluyendo enlaces.

Podemos cambiar el color de fondo para el enlace, eliminar los adornos del mismo

o asignarle un marco simplemente con modificar la hoja de estilo principal:

```
a:link { color: white; ➥
background-color: #CCCCCC; ➥
text-decoration:none; }

a:hover { color: white; ➥
background-color: red; ➥
text-decoration:none; }

a:active { color: orange; ➥
text-decoration:none; }

a:visited { color: grey; ➥
background-color: red; ➥
text-decoration:none; }
```

Los enlaces sin usar aparecen ahora en blanco con fondo gris. Si posicionamos el ratón sobre un enlace, el fondo se vuelve rojo, y el texto naranja. Los enlaces que hayamos visitado cambian su color a gris con fondo blanco. ¿Qué pasa si necesitamos aún algunos enlaces en azul y negro? ¡Y definitivamente no queremos todas nuestras imágenes clavadas a la izquierda de la página! En este caso, podemos recurrir al poder de la herencia.

¡Ojo a los Crios!

Un elemento padre precede al elemento hijo que lo rodea. La etiqueta < *b* > en el archivo *index.html* es el elemento hijo de < *p* > que en definitiva es el elemento hijo de < *body* >. La etiqueta < *body* > es el ancestro del elemento < *img* >, o dicho de otra forma: el elemento < *img* > es el descendiente de < *body* > (pero no su hijo). Los elementos hermanos están al mismo nivel, como el elemento < *p* > y el tercer elemento < *b* > de la imagen (véase Figura 8).

Lo interesante de la herencia es que los elementos hijos generalmente heredan las propiedades de sus padres. Por ejemplo, una imagen no necesita usualmente más espacio que la tabla o celda en la que esté ubicada. El elemento hijo < *b* > hereda el tamaño de tipo de letra de < *p* >. La herencia puede darnos una gran ventaja, ya que permite pasar propiedades a través de las secciones. Existen varios métodos para esto:

```
p b { color:green; }
p > b { color:blue; }
p + b { color:orange; }
```

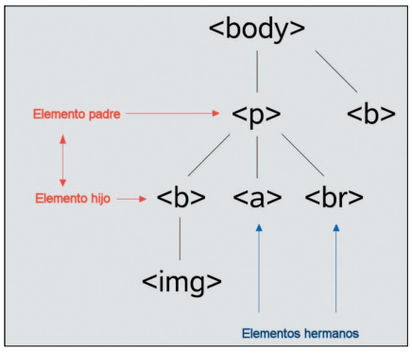

Figura 8: Esquema del árbol de jerarquía en el código HTML, mostrando los elementos hermanos, padre e hijo.

```
* { color:violet; }
```

Podemos añadir estos cuatro secciones alternativamente a nuestra hoja de estilo, activando diferentes efectos. El selector de contexto de la primera línea muestra cualquier etiqueta **dentro de las etiquetas < *p* > en verde. El selector de hijos de la segunda línea colorea el elemento** de azul si es un hijo directo, en contraposición de si es un simple descendiente de < *p* >. El signo + de la tercera línea aplica el formateo a los elementos siguientes a < *p* > y descendiendo desde el mismo elemento padre. En nuestro ejemplo, el elemento **de la línea 2 cumple estas condiciones. Sigue a < *p* > y tiene el mismo elemento padre, la etiqueta < *body* >, por lo que el texto dentro de las etiquetas** se muestran en naranja. La cuarta línea colorea cualquier elemento que no tenga asignación específica de color.

¡Un Trabajo Con Clase!

Los selectores de clase, o simplemente clases para abreviar, nos ofrecen la capacidad de dirigir cada hoja de estilo individualmente. Para ello, añadimos la siguiente línea a la hoja de estilo:

```
*.word {color: blue;}
```

Ahora abrimos el archivo *index.html* y cambiamos el texto así:

```
... as <b class="word">➥
non-descript</b> ➥
as possible ...
```

En primer lugar, creamos un valor {*color: blue;*}, que se asignará a todas las etiquetas (*) pertenecientes a la clase *.wort*. El segundo paso es establecer qué etiquetas pertenecen a la clase *.word*, de nuevo en el archivo *index.html*. Cualquier etiqueta que especifiquemos como *class* = "*worr*" asumirá automáticamente los valores de la clase *.word*, es decir, se volverá azul.

Podemos diseñar también nuestras propias etiquetas con CSS. Para ello, en primer lugar definimos la nuestra propia en la hoja de estilo, por ejemplo *redletter* {*color: red;*}. Añadiendo < *redletter* > *cualquier texto* < */redletter* > a nuestro archivo HTML colorearemos el texto entre las etiquetas rojas.

Amor de Contenedor

Si decidimos construir un sitio completo usando hojas de estilo, definitivamente tiene sentido construir *containers*. Para ello, dividimos nuestra página Web en sectores fijos usando la propiedad *position*, etiquetas < *div* > y clases. Añadimos las líneas del Listado 2 a nuestra hoja de estilo. Luego ponemos las etiquetas < *div* > alrededor del contenido de la etiqueta < *body* > en el archivo *index.html*:

```
<body>
<div class="container_central">
...
</div>
</body>
```

El contenido de la página está dentro del contenedor < *div* >. Esto significa que podemos definir tantos contenedores como queramos y ahorrar marcos y tablas.

Como la propiedad *position* contiene el valor de *fixed*, el contenedor se ajusta a una posición fija, al 0% del borde superior y al 30% del borde izquierdo de la pantalla. *width* y *height* nos permiten definir el área. Podemos fijar el *color* del tipo de letra a rojo y aplicar 20 píxeles de *padding* para asegurarnos de que tenemos suficiente espacio entre el contenido y el borde. ■

RECURSOS

[1] Páginas CSS de la W3C: *http://www.w3.org/Style/CSS/*

[2] CSS al máximo: *http://www.csszengarden.com/*

[3] Verificación de compatibilidad del navegador: *http://www.css4you.de/browsercomp.html*