Taller de CSS

Autores del manual

Este manual ha sido realizado por los siguientes colaboradores de DesarrolloWeb.com:

Miguel Angel Alvarez
Director de DesarrolloWeb.com
http://www.desarrolloweb.com
(36 capítulos)

Carlos Luis Cuenca
http://www.helloworldsolutions.com/
(1 capítulo)

Federico Elgarte
http://www.cssboulevar.com.ar/
(5 capítulos)

Diego Pinilla
Responsable relaciones exteriores de MercadoProfesional.com
http://www.mercadoprofesional.com
(1 capítulo)

Lim Chee Aun
http://cheeaun.phoenity.com/
(2 capítulos)

Oscar Alcalá
Desarrollador web freelance
(1 capítulo)

Javier Chaure
Desarrollador freelance
http://www.xski.net/
(2 capítulos)

Ana Alvarez
Directora de contenidos de Estilo y Moda .com
http://www.estiloymoda.com/
(1 capítulo)

Pol Salvat
http://www.mistrucos.net
(1 capítulo)

Artemio Artigas
Redactor de guiarte.com – Turismo, naturaleza, arte.
http://www.guiarte.com
(1 capítulo)

Manu Gutierrez
http://www.tufuncion.com
(1 capítulo)

Aplicación de estilo avanzada a los enlaces

En este artículo vamos a ver cómo podríamos crear una barra de navegación bastante dinámica utilizando únicamente las Hojas de Estilo en Cascada. En el ejemplo vamos a construir una barra de navegación que contiene enlaces de varios colores que cambian de tonalidad al pasar el ratón por encima.

Podemos ver el resultado de nuestra barra pulsando este enlace, para tener una idea más exacta de lo que queremos conseguir.

Cómo poner estilo a los enlaces

Ya lo vimos en capítulos anteriores de nuestro manual de CSS, pero lo repetimos aquí. Se define el estilo de los enlaces asignando su apariencia en sus distintos estados:

- Enlace no visitado. Se define con el atributo link.
- Enlace visitado. Se define con el atributo visited.
- Enlace activo (cuando se está pulsando). Se define con active.
- Enlace con el reton encima. Se define con hover.

Esta definición se realiza en la cabecera de la página, entre las etiquetas <STYLE> Y </STYLE>, y es global a toda la página.

Un ejemplo de esto se puede ver en esta declaración de estilos:

```
<STYLE type="text/css">
   A:link {text-decoration:none;color:#0000cc;}
   A:visited {text-decoration:none;color:#ffcc33;}
   A:active {text-decoration:none;color:#ff0000;}
   A:hover {text-decoration:underline;color:#999999;}
</STYLE>
```

Cómo dar estilo a un enlace en concreto

Si queremos resaltar nuestra barra de navegación probablemente querramos colocarla en una tabla de nuestra página web, con un color que contraste un poco con el fondo. En un caso como este, será necesario que los enlaces de la barra de navegación y los enlaces normales de la página tengan colores distintos, por estar situados sobre dos tipos de fondos distintos.

Es por esto que los enlaces de la barra van a tener un color distinto de los definidos en la cabecera de la página, con los estilos. Esto lo podemos conseguir de esta manera.

```
<a href="#" style="color:#ff0000">Mi enlace</a>
```

Hemos definido el color de un enlace de una manera específica, utilizando el atributo style, de modo que este enlace siempre tendrá el color indicado, independientemente de su estado.

Es un enlace amarillo, que quedaría muy bien resaltado sobre un fondo oscuro, como se puede ver en el ejemplo de barra de navegación siguiente.

Enlace 1

Otro enlace
Enlace 3

En la tabla anterior tenemos enlaces amarillos en una web donde los enlaces son azules por defecto.

Cómo utilizar las clases al aplicar estilo a los enlaces

También vimos en anteriores capítulos que el uso de clases puede ser muy útil a la hora de definir estilos especiales que podemos aplicar a las etiquetas que queramos. A la hora de trabajar con los enlaces también podemos aplicar las clases para definir distintos tipos de enlaces, que tienen distintos tipos de estilos.

```
A.clase1:visited {color:#ff0000;}
A.clase1:active {color:#ff0000;}
A.clase1:link {color:#ff0000}
A.clase1:hover {color:#00ff00}
```

La ventaja al utilizar las clases con los estilos de los enlaces es que podemos especificar un formato distinto al enlace dependiendo de su estado: visitado o no, activo o con el ratón sobre él.

Por si no quedó claro, al especificar el estilo con el atributo style del enlace sólo podíamos decir que el enlace lo queremos en amarillo, y siempre lo tendremos en amarillo (sea visitado o no, activo, o estemos o no con el ratón encima). Con las clases definimos un nuevo tipo de enlace al que podemos dar distintos formatos dependiendo su estado.

Otras ventajas de utilizar las clases consisten en que escribimos una única vez los estilos y que podemos cambiar el color de todos los enlaces de la clase con cambiar la delaración.

A partir de lo que acabamos de aprender podemos crear el ejemplo de barra de navegación dinámica utilizando CSS que habíamos visto al principio del capítulo. El código sería el siguiente.

```
<html>
<head>
   <title>Ejemplo CSS para enlaces</title>
<style type="text/css">
   A:link {color:#0000cc;}
   A:visited {color:#0000cc;}
   A:active {color:#0000cc;}
   A:hover {color:#0000ff;}

   A.clase1:visited {color:#ffff00;}
   A.clase1:active {color:#ffff00;}
   A.clase1:link {color:#ffff00}
   A.clase1:hover {color:#00ff00}

   A.clase2:visited {font-size:12;color:#ffffff;}
   A.clase2:active {font-size:12;color:#ffffff;}
   A.clase2:link {font-size:12;color:#ffffff;}
   A.clase2:hover {font-size:12;color:#ffff33;}

   body {font-family:arial;font-size:11;font-weight:bold}
   td {font-family:arial;font-size:11;font-weight:bold}
</style>
</head>

<body>
```

```
<a href="#">Este enlace es normal</a>
<br>
<br>
<br>
<span style="font-weight:normal;font-size:10">
Los enlaces de esta barra son especiales,
<br>
están definidos por clases
</span>
<br>
<table width="110" cellspacing="1" cellpadding="2" border="0">
<tr>
   <td bgcolor="#aa0000"><a href="#" class="clase2">Opciones 1</a></td>
</tr>
<tr>
   <td bgcolor="red"><a href="#" class="clase1">Enlace clase1</a></td>
</tr>
<tr>
   <td bgcolor="red"><a href="#" class="clase1">Otro de clase1</a></td>
</tr>
<tr>
   <td bgcolor="red"><a href="#" class="clase1">Más enlaces</a></td>
</tr>
<tr>
   <td bgcolor="red"><a href="#" class="clase1">Todavía más</a></td>
</tr>
</table>

</body>
</html>
```

Se puede ver la página en funcionamiento en este enlace. Esperamos que sirva de provecho este pequeño taller de CSS para vuestras creaciones.

Quiero agradecer la ayuda prestada para la elaboración de este artículo a Rafael Chacón, que nos mandó un generoso mail con el truquillo que hemos expuesto aquí.

*Artículo por **Miguel Angel Alvarez***

Cómo evitar que una página se imprima

Para ello, hay que echar mano de las hojas de estilo. Tanto si el documento tiene una hoja ya asociada como sino, lo que vamos a hacer es asociarle un nueva hoja de estilos. Dicha hoja contendrá un único estilo, con el código necesario para ocultar un elemento:

```
.nover{
visibility:hidden
}
```

A la hora de asociar la hoja de estilos, se le añade un modificador a la etiqueta HTML que enlaza con el fichero .css que permite especificar para qué tipo de medio se aplicará esta hoja. En nuestro caso, se aplica en el ámbito de la impresión, por lo que se utiliza el atributo media="print".

```
<link href="nombre_hoja" rel="stylesheet" type="text/css" media="print">
```

Una vez hecho esto, basta que toda nuestra página este dentro de un elemento div, que pertenezca a la clase nover.

```
<body>
<div class="nover">

-- Contenido --


</div>
</body>
```

Al hacer esto se provoca que en pantalla se visualice la página, pero que si por el contrario se decide imprimir, se le aplicará la hoja de estilos de impresión, en la que el elemento esta puesto como no visible, por lo que no se imprimirá.

Codigo Completo:

Veamos el código íntegro de la página web y la hoja de estilos asociada.

Página HTML

```
<html>
<head>
<link href="estilos.css" rel="stylesheet" type="text/css" media="print">
</head>
<body>
<div class="nover">

… contenido de la pagina

</div>
</body>
</html>
```

Hoja estilos: estilos.css

```
.nover{
visibility:hidden
}
```

> **Nota:** Esta catacterística de las hojas de estilos funciona con éxito en navegadores IExplorer 6, Netscape 7 y Opera 7. No la hemos probado en otras versiones.

Se puede ver el ejemplo relatado en el artículo en una página a parte.

> **Referencia:** Tenemos un artículo donde se explica cómo hacer un css que altere los estilos para impresión, en imprimir css.

*Artículo por **Carlos Luis Cuenca***

Títulos para tablas decorados con CSS

CSS son Hojas de Estilo en Cascada. Muchos de vosotros debéis conocerlas ya y supongo que las habréis utilizado en más de una ocasión. En cualquier caso, tanto para aprender lo que son

como para afianzar los conocimientos, podéis acceder al Manual de CSS de DesarrolloWeb.com.

Vamos a ver un uso de las CSS que puede ser muy interesante para hacer tablas con títulos que tengan un cierto estilo. Lo bueno de las CSS es que podemos definir el estilo una vez y se puede utilzar en múltiples elementos de la página. Con todo ello vamos a tratar en este artículo: la definición de un estilo y la aplicación para hacer distintos tipos de decoración de tablas.

Podemos ver el resultado final que vamos a obtener.

Definición de los estilos

En la cabecera de la página colocamos la etiqueta <style> que sirve para definir los estilos a utilizar en la página. Vamos a definir un estilo por defecto para las celdas (etiqueta <td>) y un par de clases, la primera para las tablas y la segunda para las celdas titular (las que tienen color de fondo).

```
<style type="text/css">
td {
   font-family:verdana,arial;
   font-size:8pt;
}
.estilotabla{
   background-color:ffffff;
   border-style:solid;
   border-color:666666;
   border-width:1px;
}
.estilocelda{
   background-color:ddeeff;
   color:333333;
   font-weight:bold;
   font-size:10pt;
}
</style>
```

Los atributos de estilos se pueden conocer en el manual de CSS. En este caso, para la clase estilotabla estamos definiendo un color de fondo, un borde sólido, un color del borde y un ancho del borde, por ese orden. Para la clase estilocelda estamos definiendo un color de fondo, un color del texto, un grosor de la fuente y un tamaño de la fuente.

Utilización de los estilos para obtener una tabla decorada

Veamos el código de la primera tabla del ejemplo.

```
<table width=280 height=18 cellpadding=2 cellspacing=2 class="estilotabla">
<tr><td class="estilocelda">Título de sección</td></tr>
<tr><td>Este es un texto de lo que podría ser la parte de abajo de la tabla con el contenido relacionado con este título.</td></tr>
</table>
```

Lo único que tiene de especial es que utiliza las clases que se han definido previamente. En la etiqueta <table> se utiliza la clase estilotabla y en la etiqueta <td> que queremos que sea el titular se utiliza la clase estilocelda. La otra celda tendrá el estilo definido para todas las celdas en general.

La otra tabla tendría este código.

```
<table width=280 height=18 cellpadding=2 cellspacing=2 class="estilotabla">
<tr><td class="estilocelda">Título de sección</td></tr>
</table>
<table width="280" cellpadding="2" cellspacing="2"><tr><td>
Este es un texto de lo que podría ser la parte de abajo de la tabla con el contenido relacionado con este título.
</td></tr></table>
```

En este caso utilizamos dos tablas para hacer el efecto. La tabla de arriba tiene un borde y la de abajo no. Para ello aplicamos la clase definida para la tabla y la celda solamente en la tabla de arriba, así el borde definido en la declaración de estilos sólo afecta a la tabla de arriba.

Podemos ver el resultado en una página aparte.

*Artículo por **Miguel Angel Alvarez***

Bordes punteados con CSS

Vamos a ver otra manera de utilizar las CSS (Hojas de Estilo en Cascada) para crear un estilo que podemos utilizar en tablas HTML. Es un estilo para realizar los bordes de tablas con líneas punteadas, lo que resulta bastante original, aunque nuestro consejo es no abusar de ello.

El efecto buscado es el siguiente:

Nota: Para ver correctamente este efecto es necesario disponer de Internet Explorer versión 6. En Mozilla también se pueden ver los bordes punteados, pero el punto es mucho más fino, por lo que el efecto queda más discreto.

Para conseguirlo simplemente tenemos que utilizar la propiedad de CSS "border-style", asignando el valor "dotted", que quiere decir punteado en inglés. Además, podemos definir otros valores al estilo de borde punteado, como puede ser el color del borde o el color de fondo de la tabla.

Para este ejercicio práctico hemos definido una clase que aplicaremos a las tablas u otros elementos que deseemos que tengan el borde punteado.

Nota: una clase es una declaración de estilo que podemos utilizar en cualquier elemento de una página. Para ello se incluye el atributo class="nombre_clase" en la etiqueta HTML que deseamos que tenga dicho estilo. Podemos aprender más sobre clases y estilos CSS en el manual de Hojas de Estilo en Cascada. http://www.desarrolloweb.com/manuales/2

```
<style type="text/css">
.punteado{
  border-style: dotted;
   border-width: 1px;
   border-color: 660033;
   background-color: cc3366;
   font-family: verdana, arial;
   font-size: 10pt;
}
</style>
```

Esta clase tiene definidos una serie de atributos de estilos. Desde el primero hasta el último son: tipo de borde, ancho del borde, color del borde, color de fondo, tipo de letra y tamaño de la fuente. El atributo que nos interesa a nosotros es el primero, en el que se indica que se desea un tipo de borde punteado.

Para utilizar el estilo tenemos que asignar a un elemento de HTML la clase generada antes. Para ello utilizamos el atributo class de HTML. Tal como se ha definido la clase se puede utilizar en cualquier elemento de la página web, como una tabla o una celda.

```
<table class="punteado" width=80% align="center">
<tr>
  <td>
  <b>Esto es un texto</b>
   <br>
   Lo pongo para ver como queda. Me gustará seguro! y a vosotros también.
  </td>
</tr>
</table>
```

En este ejemplo, toda la tabla tendrá un efecto de borde punteado. Si colocásemos esa clase únicamente en una celda, sólo esa celda tendría el borde punteado. Es el caso del ejemplo siguiente:

```
<table width=300>
<tr>
  <td class="punteado"><b>Título de la tabla</b></td>
</tr>
<tr>
  <td>
   Aquí podríamos poner cualquier cosa. Sería como el cuerpo de la tabla que correspondiese con el titular que sí tendría algún estilo majete.
  </td>
</tr>
</table>
```

Estos dos ejemplos se pueden ver en una página aparte, que incluye también una tercera propuesta de uso del estilo de borde punteado.

No abusar del uso de los bordes punteados

Los bordes con líneas punteadas pueden ser muy útiles y vistosos, pero si nos pasamos en su utilización puede darse el caso que el efecto quede poco agradable.

El borde punteado llama la atención sobre el elemento que lo utiliza, aunque provoca una sensación de inestabilidad. También hace que parezca que no está terminado, o bien, no está integrado con el resto del diseño.

*Artículo por **Miguel Angel Alvarez***

Estilos en campos de texto

Vamos a ver unos ejemplos sobre cómo aplicar estilos avanzados a campos de texto en páginas web. Nos referimos a campos de texto de los normales <input type=text> y campos de texto que soportan varias líneas <textarea>.

Con estilos, como ya se ha debido aprender en el manual de CSS, podemos definir el formato de presentación de cualquier elemento de la página. Los campos de texto, que siempre tienen una forma muy específica, también aceptan especificaciones de estilos para variar su apariencia típica.

Vamos a ver varios ejemplos para comprobar las posibilidades de las hojas de estilos, aplicadas sobre campos de texto y textareas.

```
<input type="text" name="campotexto0" size="12" style="border-width: 2px; border-style: solid; font-size:8pt; color: #009900; letter-spacing : 3px;">
```

Este campo de texto se presentará con un borde de 2 pixels, un borde sólido, tamaño de la letra de 8 puntos, color del borde y de las letras verde un poco oscuro. También se define un espaciado entre las letras de 3 pixel.

```
<input type="text" name="campotexto1" size="12" style="background-color:e3e3e3; border: 1px solid #666666; font-size:8pt; color: #000099">
```

Este campo de texto tiene los estilos declarados con una sintaxis un poco distinta, ya que se han agrupado varios estilos relativos al borde en un solo atributo. Los estilos con los que se presentará son: color de fondo grisáceo, borde de 1 pixel, borde de estilo sólido, color del borde gris más oscuro, tamaño del texto de 8 puntos y color de las letras azul.

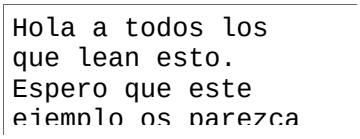
```
<textarea cols="20" rows="3" name="campotexto2" style="overflow:auto; border: 1px solid #ff6666;"></textarea>
```

Este campo de texto con varias líneas, también llamado textarea, tiene viarios estilos, que son parecidos a los que hemos visto para el anterior campo de texto, con la salvedad del atributo overflow, que está definido como auto. El atributo overflow tiene relación con las barras de desplazamiento que aparecen en los textarea. El valor auto sirve para que la barra de desplazamiento vertical del campo de texto sólo aparezca en caso que se necesite, es decir, si el texto del campo supera las líneas que tiene reservadas el textarea. Si queremos que no se vean las líneas nunca, podemos asignarle el valor hidden al atributo overflow. Por lo que

respecta a los otros estilos de este campo de texto de múltiples líneas, se han definido 3 valores para el estilo del borde, en un único atributo. Los estilos son borde de un píxel, borde de estilo sólido y borde de color rojo pastel.

```
<textarea cols="20" rows="3" name="campotexto3" readonly style="overflow:auto; border-style:dashed; border-color:555555; border-width: 1px;">

Hola a todos los que lean esto.
Espero que este ejemplo os parezca interesante!!
Saludos y suerte!
Miguel
</textarea>
```



En este campo textarea, hemos incluido también un texto con el que se inicializará su contenido. Primero llamamos la atención sobre el atributo de HTML readonly, que sirve para que el campo textarea no sea editable, es decir, que no se pueda cambiar su contenido. También con estilos CSS se han definido una serie de valores para la apariencia, estos son: mostrar las barras de desplazamiento sólo cuando toca, un borde del campo punteado, un color del borde gris oscuro y un ancho del borde de 1 pixel.

Conclusión

Esperamos que con estas indicaciones podáis aprender un poco más sobre cómo modificar el estilo de un campo de texto, para adaptarlo mejor al diseño de vuestras páginas.

Hay que tener en cuenta que las características de estilos no siempre están disponibles en todos los navegadores. Las más importantes sí que las podremos ver en todos los navegadores que soporten estilos, aunque ciertos valores, como el borde punteado, no se pueden visualizar correctamente en navegadores antiguos. Pasa lo mismo con el atributo readonly, que no siempre ha estado disponible en HTML.

*Artículo por **Miguel Angel Alvarez***

Maquetar una página con CSS

Vamos a realizar un ejercicio de maquetación de una página web utilizando únicamente hojas de estilo en cascada (CSS), separando completamente el contenido del archivo HTML de las definiciones del aspecto, que se guardarán en un archivo .css. El ejercicio lo realizaremos paso a paso, partiendo de una imagen diseñada previamente con un programa de edición gráfica como Photoshop.

> **Referencia:** Tenemos un manual para aprender CSS en DesarrolloWeb.com, donde además se explican las primeras nociones y conceptos que hay que conocer sobre la maquetación.

Imágenes de partida

Podemos ver la imagen que hemos creado y que vamos a intentar maquetar lo más parecido

posible. No es el objetivo de este manual ofrecer las técnicas para realizar esta imagen, aunque en otros manuales de DesarrolloWeb.com podemos ver tutoriales para aprender algunos de los trucos de diseño utilizados.

Se trata de un diseño sencillo, pero en el que se encuentran elementos distintos y variados con los que trabajar.

De esta imagen hemos extraído algunos gráficos, que utilizaremos a la hora de maquetar el diseño. Sería interesante descargarlo para poder realizar el ejercicio por vuestra cuenta.

Para los impacientes, tenemos un enlace a la página resultado que vamos a conseguir realizar al final del artículo. Puede ser bueno verla para hacerse una idea de donde queremos llegar.

Desarrollo de la página y la hoja de estilos

Vamos a generar los archivos HTML y CSS a la vez, pero paso a paso, de modo que podamos explicar las etiquetas y estilos que hemos utilizado para cada parte de la página.

Como primer paso, en la cabecera <head> del documento HTML, enlazaremos con una hoja de estilos externa.

```
<head>
<title>La web del invierno</title>
<link rel="STYLESHEET" type="text/css" href="estilo.css">
</head>
```

El cuerpo de la página <body>

En la declaración de estilos CSS, para el cuerpo de la página, hemos definido una imagen de fondo "fondo.gif", que se repetirá por toda la página en un mosaico. También se definen unos márgenes y el alineamiento del texto, en este caso centrado, para que el contenido de la página aparezca en el centro (esto es necesario para Internet Explorer, el centrado en Mozilla y otros navegadores se realiza en la capa principal con el atributo "margin" definido como "auto").

Además se definen otros atributos para el cuerpo de la página, que luego heredarán otros elementos, como el tipo de letra o el color del texto.

```
BODY {
background : #C0D9D9 url(images/fondo.gif) repeat;
font : 8pt Verdana, Geneva, Arial, Helvetica, sans-serif;
color : #666666;
margin : 20px 0px 20px 0px;
text-align: center;
}
```

La capa contenedor

Generalmente, se utiliza una capa principal, a la que hemos llamado contenedor. Dentro de esta capa se colocan todos los elementos que va a tener la página.

```
<div id="contenedor">

</div>
```

En esta capa definimos el alineamiento del texto a la izquierda (porque en el cuerpo habíamos centrado el texto, para que Internet Explorer centre la capa contenedor y deseamos que la alineación por defecto sea a izquierda). También definimos una anchura de 700px, un color de fondo blanco y el margen, con el atributo "margin", lo definimos como "auto", para que Mozilla y otros navegadores centren la capa.

```
#contenedor{
text-align: left;
width: 700px;
background-color : #ffffff;
margin: auto;
}
```

Por cierto, nos hemos dejado deliberadamente el borde de la capa, que habíamos definido en el diseño original. Se podría haber definido el atributo "border", pero eso nos repercute negativamente en la maquetación en Explorer. Veremos más adelante cómo colocarlo para que se vea correctamente en todos los navegadores.

Este ejercicio lo vamos a ver en varios pasos. En el siguiente bloque mostraremos cómo se maqueta la cabecera y la barra de navegación.

*Artículo por **Miguel Angel Alvarez***

Maquetar una página con CSS II

Continuamos el ejercicio práctico para realizar la maquetación de una página paso a paso con capas y hojas de estilo en cascada. Se puede ver el artículo anterior de esta serie en Maquetar una página con CSS.

La cabecera de la página

La imagen de la parte de arriba de la página la vamos a colocar en un único archivo gráfico. Es lo más cómodo para este diseño, pues la cabecera no tiene otro motivo que decoración.

```
<div id="cabecera"><img src="images/cabecera.jpg" width="700" height="106" alt="La Web del Invierno"
border="0"></div>
```

Vemos que es una simple imagen, pero atención, que tenemos que colocar el a continuación de <img> sin ningún espacio ni salto de línea, porque si no, Internet Explorer, nos introducirá un pequeño márgen debajo de la imagen, que queremos evitar.

Los atributos de estilo definidos para la cabecera son las dimensiones de la capa, que queremos que sean las mismas que las de la imagen. Aunque en este caso podríamos habernos ahorrado definir esos valores porque son los que se tomarían por defecto.

```
#cabecera{
height : 106px;
width: 700px;
}
```

La barra de navegación

Vamos con la capa utilizada para definir la barra de navegación horizontal que hay debajo de la cabecera.

```
<div id="navegador">
<a href="#" class="enlacenav">Portada</a> |
<a href="#" class="enlacenav">Invierno</a> |
<a href="#" class="enlacenav">Diciembre a marzo</a> |
<a href="#" class="enlacenav">La chimenea</a> |
<a href="#" class="enlacenav">Deportes de invierno</a> |
<a href="#" class="enlacenav">Contacto</a>
</div>
```

Como se puede ver, simplemente hemos definido una serie de enlaces dentro de una capa. Hay que fijarse que además los enlaces tienen una clase, llamada "enlacenav", que utilizaremos para darle un estilo específico a a estos enlaces, independiente del definido por defecto en la página.

Por lo que respecta a la capa, se define un color y una imagen de fondo, unos márgenes internos (atributo padding) y un borde, tanto para la parte de arriba de la capa como para la de abajo.

```
#navegador{
background : #F5F4C3 url(images/fondonav.gif);
padding : 3px 10px 5px 10px;
border-top : 1px solid #cccccc;
border-bottom : 1px solid #cccccc;
}
```

Para los estilos de los enlaces utilizamos una clase. Para definir los estilos de cada uno de los estados de los enlaces (visitados, activos, no visitados, etc), se utilizan las pseudo-clases VISITED, ACTIVE, FOCUS, LINK Y HOVER. Simplemente definimos el color de los enlaces, el mismo para todas las pseudo-clases, menos para HOVER, que tiene un color distinto. HOVER es el estado del enlace cuando el puntero ratón está situado encima. En este caso, cuando el ratón esté encima, cambiará de color.

```
A.enlacenav, A.enlacenav:VISITED, A.enlacenav:ACTIVE, A.enlacenav:FOCUS, A.enlacenav:LINK{
color: #494E6B;
}
A.enlacenav:HOVER{
color: #3F7DE3;
}
```

Podemos ver cómo queda el ejercicio realizado hasta el momento.

*Artículo por **Miguel Angel Alvarez***

Maquetar una página con CSS III

Este ejercicio trata de maquetar una página utilizando capas y css. La primera parte se puede ver en: Maquetar una página con CSS.

El cuerpo de la página

La parte de la página donde colocamos la información principal. Crearemos una capa independiente para el cuerpo y colocaremos dentro el título, el texto y otros elementos que queramos situar. Los elementos los introducimos con las etiquetas HTML que deberían tener en una página básica. Luego, con CSS definiremos el estilo para el cuerpo y cada una de las etiquetas que colocamos dentro.

```
<div id="cuerpo">
<h1>Título de la página</h1>
<p>
En este artículo vamos a conocer la maquetación de paginas utilizando Hojas de estilos en cascada (CSS). Veremos cómo realizar este tipo de maquetación, junto con algunas ventajas e inconvenientes. Para muchos será todavía un campo por explorar. Aunque no vamos a entrar en grandes detalles, vamos a intentar dar a conocer la maquetación con CSS para que cubrir la posible laguna por parte del lector. En capítulos sucesivos ampliaremos la información y ofreceremos tutoriales más prácticos.
</p>
<p>
Como se ha podido aprender en el Manual de CSS, las hojas de estilo en cascada ayudan a separar el contenido de la forma, es decir...
</p>
<div id="navabajo">
<a href="#">Volver</a> |
<a href="#">Portada</a> |
<a href="#">Mapa del sitio</a>
</div>
</div>
```

Vemos que el cuerpo tiene un título, varios párrafos y un div, incluido dentro del propio cuerpo, con una segunda barra de enlaces que faciliten la navegación para las personas que lleguen al final del scroll vertical de la página.

Los estilos del cuerpo definen la anchura, margen, margen interno, y un color de fondo. Además, se define el atributo "float:left" para hacer que el cuerpo "flote" a la izquierda. El resultado es que la capa del cuerpo se coloque a la izquierda y el contenido escrito a continuación se sitúe, rodeando a esta capa, a la derecha. (El efecto es el mismo que si asignamos en HTML el atributo align=left en una imagen)

Para posibilitar la disposición en dos columnas que hemos definido en el diseño original, vamos a hacer que la capa de la izquerda -el cuerpo- "flote" a la izquierda. Posteriormente, la capa de la derecha que aun no hemos colocado-el lateral-, haremos que "flote" a la derecha.

```
#cuerpo{
width:480px;
margin-left: 8px;
padding: 12px 0px 10px 0px;
background-color : #ffffff;
float:left;
}
```

También se define un estilo para cada algunas de las etiquetas que hemos situado dentro del cuerpo:

```
H1{
font-size: 12pt;
}
```

Los encabezados de nivel 1, que tengan tamaño de letra 12pt.

```
#navabajo{
font-weight : bold;
}
```

Para asignar una negrita en el div de la parte inferior del cuerpo, que tiene enlaces para facilitar la navegación.

Podemos ver el ejercicio tal como queda con los pasos realizados hasta el momento.

*Artículo por **Miguel Angel Alvarez***

Maquetar una página con CSS IV

En los pasos anteriores de este taller vimos cómo crear la cabecera y cuerpo de la página. Ahora vamos a ver cómo hacer el lateral derecho de la página.

La capa lateral

En el lateral derecho situamos una nueva capa, que ofrece acceso a servicios y otras informaciones.

```
<div id="lateral">
... contenido lateral...
</div>
```

El contenido que vamos a situar dentro de esta capa lo veremos por partes, pues tiene bastantes detalles que destacar tranquilamente. Los estilos son los siguientes:

```
#lateral{
width: 200px;
background-color: #EBF2FE;
border-bottom : 1px solid #cccccc;
border-left : 1px solid #cccccc;
float:right;
}
```

Se define una anchura, un color de fondo y bordes de color gris claro en la parte lateral izquierda y abajo, los otros dos lados no tendrán borde por estar en contacto con los bordes de otros elementos.

Además, con el atributo float:right, indicamos que este lateral debe "flotar" hacia la derecha. Así, el cuerpo flota a la izquierda y el lateral a la derecha, con lo que conseguimos una disposición en 2 columnas.

Veremos a continuación los elementos que vamos a colocar dentro de la capa lateral, en una especie de cajas independientes. Aunque, antes de ver esas cajas una a una, merece la pena conocer en líneas generales cómo van a crearse. Cada caja tendrá este código HTML, compuesto por un titulo y un contenido de la caja:

```
<h2 class="titlat">Titulo de la caja</h2>
<div id="idunico" class="cuerpolateral">
Contenido de la caja
</div>
```

El título lo incluimos con una etiqueta <h2> y la parte de la caja con el contenido, se define

con un div. Cada uno de estos elementos tiene una clase, que se aplicará a los mismos elementos en cada una de las cajas, de modo que todos los elementos del lateral compartan un mismo estilo.

```
.titlat{
background-color:#68729E;
color:#ffffff;
font-size:8pt;
text-transform : uppercase;
padding: 7px 3px 7px 8px;
font-weight : normal;
letter-spacing : 2px;
margin: 0px 0px 8px 0px;
}

.cuerpolateral{
padding: 5px 4px 13px 10px;
}
```

El encabezado de nivel 2 utiliza la clase "titlat", que define un color de fondo, un color del texto, un tamaño de letra, un cambio a mayúsculas de las letras del título, unos márgenes internos, peso de letra normal (no negrita, como suelen ser los encabezamientos por defecto), un espaciado de letras de 2 pixel y un margen. Los titulares llevan asociado un salto de línea doble arriba y abajo, que deseamos evitar y para ello hemos definido un margen de 0 pixels, menos en la parte de abajo, que tendrá 8 pixel.

Las cajas laterales también tienen un estilo, que se aplica a todos los cuerpos de las cajas que hay en el lateral. Ese estilo simplemente define unos márgenes internos.

Caja de buscar

Uno de los elementos que vamos a colocar dentro del lateral es una caja de búsqueda, con un formulario para realizar búsquedas internas, dentro del sitio, y en todo el web.

Esa caja de búsqueda se coloca en un formulario. Hemos puesto diversos identificadotes a los elementos que hay dentro del fomulario, para poder aplicar estilos a cada componente por separado. Aunque algunos de estos selectores ni siquiera los hemos llegado a utilizar, pueden venir bien si queremos hacer en el futuro modificaciones de la hoja de estilos para actualizar el diseño del web.

```
<h2 class="titlat">Buscar</h2>
<div id="fbuscar" class="cuerpolateral">
<form>
<div id="campotexto"><input type="text" name="criterio"></div>
<div id="botonbuscar"><input type=image src="images/go.gif" width="25" height="15"></div>
<div class="radio"><input type="radio" name="op" value="1"> En la Web del invierno</div>
<div class="radio"><input type="radio" name="op" value="2"> En toda la Web</div>
</form>
</div>
```

Los elementos que hemos definido en la hoja de estilos para este pequeño formulario son los siguientes:

```
INPUT {
font-size : 8pt;
}
```

Con ello definimos que los campos de texto tienen un tamaño de letra de 8 puntos.

```
#fbuscar form{
margin-bottom : 0px;
margin-top : 0px;
}
```

El formulario, que está situado dentro de la capa fbuscar, no debe tener márgenes, ni arriba ni abajo.

```
#campotexto{
float: left;
}
```

La capa "campotexto", donde está el campo de texto, hemos definido que debe "flotar" a la izquerda.

```
#campotexto input{
width:100px;
}
```

El input que hay dentro de la capa campotexto debe tener 100 pixels de ancho.

```
#botonbuscar {
padding-top : 3px;
padding-left: 106px;
}
```

La capa donde está el botón de submit, que en este caso es una imagen de submitir (<input type="imagen">), tiene un margen interno de 3 pixels por arriba, y de 106 por el lado izquierdo. Los 106 pixels de ancho salen de los 100 que ocupa el campo de texto que hay a la izquierda, más 6 pixeles adicionales, que son el verdadero margen que habrá entre el campo de texto y la imagen de submitir.

```
#botonbuscar input{
border : 0px none;
}
```

Con esta última definición estamos indicando que la imagen de submitir (el <input type="image"> que hay dentro de la capa botonbuscar) no tenga borde.

```
.radio{
clear:both;
}
```

Esta clase, que afecta a las capas donde están los botones de radio, define que no deben haber elementos "flotando" ni a la izquierda ni a la derecha, de los botones de radio.

La caja de registro

En la siguiente caja del lateral aparece un pequeño texto invitando a registrarse al visitante.

```
<div id="registro" class="cuerpolateral">
<a href="#">Registrese con nosotros</a> y obtenga muchas ventajas.
</div>
```

Esta capa no tiene ningún estilo específico, simplemente se comporta heredando los estilos de otras capas y con los que se han definido en las clases que se utilizan.

La caja de otras informaciones

Situaremos una última caja dentro del lateral, que contiene enlaces a otras informaciones. Dentro de la caja colocaremos varios enlaces dentro de una lista.

```
<h2 class="titlat">Otras informaciones</h2>
<div id="otras" class="cuerpolateral">
<ul>
<li><a href="#">Quienes somos</a>
<li><a href="#">Nuestra misión</a>
<li><a href="#">Agenda de eventos</a>
</ul>
</div>
```

Para personalizar el estilo de la lista de enlaces se utilizan los siguientes estilos.

```
#otras ul{
margin : 5px 10px 0px 0px;
padding: 0px 0px 0px 4px;
list-style: none;
}
```

Por un lado tenemos el estilo definido para toda la lista de elementos. En este caso se eliminan los márgenes que este tipo de listas tienen implícitos. Se coloca también un margen interno 4 pixel a la izquierda y cero en el resto de las posiciones. Con "list-style:none" se indica que no se desea ninguna bolita a la izquierda de los elementos, puesto que la vamos a colocar a continuación nosotros manualmente como fondo de los <li>.

```
#otras li{
padding-left: 14px;
background: transparent url("images/bullet.gif") 0 2px no-repeat;
margin-bottom: 10px;
}
```

Por otra parte, para cada uno de los elementos de la lista, de define un espacio de 14 pixel a la izquierda. Esos 14 pixel sirven para hacer espacio, para que quepan unas pequeñas imágenes que vamos a poner de fondo en las listas, que van a hacer las veces de bolita. También se define un fondo de los <li> que es la imagen con la bolita personalizada, a juego con nuestro diseño. También se define un margen en la parte inferior.

Después de integrar todo lo que hemos visto en este ejercicio para crear el lateral de la página, el diseño queda tal como se puede ver en esta página.

*Artículo por **Miguel Angel Alvarez***

Maquetar una página con CSS V

Apuntaremos los últimos retoques en el diseño de la página con CSS para finalizar el taller de maquetación con CSS. Se puede ver la primera parte del artículo.

Pie de la página

Este elemento no lo habíamos previsto en la imagen original, creada previamente, pero lo hemos decidido colocar porque lo necesitamos, para que en la parte donde está el cuerpo y el lateral, aparezca el fondo de color blanco. En Explorer aparece el fondo blanco sin ningún problema, pero en Mozilla y otros navegadores, al estar las dos capas de cuerpo y lateral "flotando" a izquierda y derecha, no entiende que deba mantener el fondo blanco definido en el container.

No se si se entiende esto bien, pero lo mejor es hacer una prueba y ver lo que hemos definido hasta el momento en la plantilla en Firefox o Mozilla. Veremos que el fondo blanco no está continuado hacia abajo.

```
<div id="pie">
Pruebas de maquetación CSS © 2005 DesarrolloWeb.com
</div>
```

Esta capa tiene el siguiente estilo definido:

```
#pie{
clear : both;
color : #cccccc;
text-align : right;
margin : 10px 10px 0px 10px;
padding-bottom:10px;
}
```

Con "clear:both" indicamos que la capa debe mostrarse sin elementos flotando a izquierda y derecha, de modo que la posición de la capa será inmediatamente por debajo de la capa cuerpo y lateral.

Luego se define un color para el texto una alineación de texto, unos márgenes y un margen interno por la parte de debajo de 10 pixel.

El borde externo

El diseño original incluía un borde de 2 pixel rodeando a toda la capa principal. Podemos hacer la prueba de incluir un borde en la capa contenedor. Para ello hay que añadir en el estilo para la capa contenedor el atributo border, de la siguiente manera.

```
#contenedor{
text-align: left;
border: 2px solid #cccccc;
width: 700px;
margin: auto;
background-color : #ffffff;
}
```

En Mozilla y navegadores similares, todo es correcto. Pero en Internet Explorer la cosa tiene su problema. Esto es debido a que el espacio de los bordes, en Explorer, se toma del que se haya asignado a la propia capa y en Mozilla y otros navegadores, se toma como espacio adicional, aparte del que se haya asignado a la capa en si.

Lo mejor es probarlo y verlo por uno mismo, o bien encontrarse con el problema y encontrarle solución sin tener que romperse la cabeza.

Nosotros lo hemos arreglado quitando el borde en la capa contenedor y creando una nueva capa, en la que situaremos el contenedor. Esa nueva capa la hemos llamado borde y es la que

va a tener el estilo de borde definido.

```
<div id="borde">
<div id="contenedor">
.... contenido de toda la página
</div>
</div>
```

Para conseguir el borde se han definido el siguiente estilo para la capa borde.

```
#borde{
border: 2px solid #cccccc;
text-align: left;
width: 700px;
margin: auto;
}
```

Primero hemos definido un borde de 2 pixel. Luego un centrado a la izquierda (para contrarrestar el centrado al centro que tiene el body y que habíamos puesto para que Explorer centrase la capa del contenido. También se incluye una anchura de 700 pixel y un margen "auto" para que Mozilla y otros navegadores centren la capa.

El resultado final del ejercicio se puede ver en una página aparte. Por supuesto, conviene ver el resultado final utilizando varios navegadores distintos.

Conclusión

Hemos visto cómo maquetar una página utilizando CSS paso a paso. Esperamos que hayáis podido seguir el ejercicio y que ninguna dificultad os haya frenado. Realmente el trabajo con CSS para la maquetación es una tarea fácil, pero también es muy sencillo encontrarnos con escollos o problemas misteriosos que no parecen tener respuesta.

Sin ser un diseño complicado, realizar esta maquetación nos ha llevado varias horas de trabajo y algún que otro padecimiento, que por suerte no ha llegado a desesperación. Sobretodo existen dificultades a la hora de conseguir el diseño que se vea correctamente en todos los navegadores del mercado. Este diseño lo hemos probado con éxito en Mozilla, Firefox, Netscape, Opera y Explorer.

Para que la compatibilidad entre navegadores no signifique un problema muy pesado, nuestro consejo y el de otros desarrolladores, es diseñar con Mozilla o navegadores similares. Luego se puede ver el resultado en Explorer y adaptar lo que fuera necesario para terminar de cuadrar el diseño. En este caso habrán pocas cosas que cambiar, mientras que si diseñamos para Explorer y luego vemos el resultado en otros navegadores, seguramente nos tiremos de los pelos porque nada esté en su sitio.

La experiencia en el trabajo con CSS, nos dice que a menudo surgen los mismos problemas o similares. Una vez que ya los hemos resuelto unas pocas veces y nos hemos acostumbrado a ello, igual que hicimos con los detalles relativos al HTML y la maquetación con tablas, CSS se torna mucho más sencillo, potente y rápido de desarrollar.

Resultado final del ejercicio.

*Artículo por **Miguel Angel Alvarez***

Variar el diseño y maquetación con la hoja de estilos

Hemos visto en una serie de artículos anteriores un ejemplo de cómo maquetar una página utilizando únicamente CSS para posicionar sus distintos elementos. Una de las principales ventajas de CSS es que se puede cambiar el aspecto de una página radicalmente, sin necesidad de cambiar su código HTML. Por ello, nos ha parecido interesante seguir profundizando en la maquetación de páginas web con CSS, ofreciendo una nueva propuesta de diseño para el mismo archivo HTML que habíamos utilizado anteriormente.

Para empezar, podemos echar un vistazo al diseño que hemos creado, utilizando un programa de edición gráfica tipo Photoshop o Fireworks. Vamos a trabajar sobre esta imagen, para que el diseño resultante sea lo más parecido posible.

También ofrecemos para descarga un archivo comprimido con todas las imágenes que vamos a utilizar en este diseño. Será interesante tenerlo a mano para tratar de hacer por nosotros mismos el ejemplo.

El mismo código HTML

Insistimos en la idea de que vamos a utilizar el mismo código HTML que hemos construido al hacer el ejemplo del artículo de maquetación CSS, dado que las hojas de estilo en cascada nos proporcionan herramientas para alterar el aspecto de la página sin editar siquiera el archivo HTML.

La anterior maquetación ya se hizo pensando en que se iba a utilizar para proponer más de un diseño, por lo que se añadió alguna etiqueta, clase o identificador adicional para facilitar este paso.

Aunque durante la creación de este segundo ejemplo hemos estado tentados de editar el código HTML, sólo hemos cambiado un aspecto que vamos a señalar a continuación.

Se trata de la imagen de la cabecera. Si nos fijamos en el archivo HTML anterior, comprobaremos que la imagen está incluida por medio de una etiqueta <img>. Al definirse la ruta de la imagen y sus valores de ancho y alto por medio de los atributos de <img>, no podemos cambiar esos datos con la hoja de estilos. Como deseamos cambiar la imagen en distintos diseños, en lugar de colocar la imagen con la etiqueta directamente en el código HTML, vamos a utilizar un truco que hemos aprendido en CSSZenGarden, que se basa en incluir un titular de texto, que luego vamos a sustituirlo por la imagen que deseemos. A su vez, hay que decir que este truco es original de Douglas Bowman http://www.stopdesign.com/articles/css/replace-text/.

Antes, habíamos definido el siguiente pedazo de código para situar la imagen de cabecera:

```
<div id="cabecera"><img src="images/cabecera.jpg" width="700" height="106" alt="La Web del Invierno" border="0"></div>
```

Ahora, el código de la cabecera será el siguiente:

```
<div id="cabecera">
  <h1><span>La Primavera</span></h1>
</div>
```

Simplemente hemos definido un titular, que luego no aparecerá en la página, porque lo

ocultaremos por medio de el atributo visibility de CSS. En su lugar, definiremos un fondo para la capa "cabecera" y asignaremos sus atributos por medio de hojas de estilo.

Así quedarán los estilos para el elemento cabecera y el encabezamiento <h1>:

```
#cabecera{
   background: transparent url(images/cabecera.jpg) no-repeat;
   height: 288px;
   width: 549px;
}

#cabecera h1 {
   margin: 0px 0px 0px 0px;
}
#cabecera h1 span {
   display:none;
}
```

El nuevo código CSS

Aparte de lo comentado para la cabecera, el código CSS creado para aplicar los estilos no aporta mucha novedad a lo que hemos visto hasta el momento.

Básicamente se ha utilizado nuevas imágenes para los fondos y hemos variado los tamaños y márgenes de las capas. Aparte, en la parte central o cuerpo de la página, se ha alineado de manera distinta los elementos, quedando los cuadrados del buscador y enlaces a otras secciones a la izquierda y el texto de la página a la derecha.

También se puede apreciar como se han utilizado unas imágenes para decorar el fondo de los titulares de los recuadros de la izquierda. También se ha colocado una imagen en el fondo donde está el texto de la página. Esta imagen está muy difuminada para permitir leer el texto con comodidad.

Vamos a dejar de lado, tal vez para próximos artículos, la explicación detallada de la declaración de estilos utilizada. En lugar de eso ponemos los enlaces hacia el archivo HTML y el CSS.

Página con el resultado final del ejercicio.

Declaración de hojas de estilo utilizada.

*Artículo por **Miguel Angel Alvarez***

Taller de CSS, Opacity

En esta ocasión mostraré un efecto bastante simpático que podemos aplicar en capas, imágenes, formularios, etc…

La propiedad opacity funciona tanto en Internet Explorer como en Firefox:

IE:

filter: alpha(opacity=50)

Firefox:

opacity: .5

Mozilla:

-moz-opacity:0.5

El siguiente ejemplo funciona en los dos navegadores:

```
<style type="text/css">
.ejemplo {width: 100%; background-color: red}
.opaco {filter: alpha(opacity=50); opacity: .5}
</style>

<p class="ejemplo">Ejemplo sin opacity.</p>
<p class="ejemplo opaco">Ejemplo con opacity a 50%.</p>
<p><img src="imagen.gif" width="100" height="109"><img src="imagen.gif" class="opaco" width="100"
height="109"></p>
```

Ver ejemplo en marcha

Referencia: Os dejamos un link a otro artículo de DesarrolloWeb.com donde se ofrecen explicaciones adicionales y actualizadas sobre cómo crear transparencia con CSS.

*Artículo por **Federico Elgarte***

Enlaces con estilos CSS que simulan botones

En este taller de CSS vamos a mostrar cómo realizar un enlace que tenga aspecto de botón. Dicho de otra manera, vamos a crear botones a partir de enlaces, aplicando una hoja de estilo que hará que los enlaces se muestren de forma similar a como serían los botones. Para ello, vamos a hacer que, al pasar el ratón por encima de un enlace, este parezca como que está pulsado.

Lo mejor para darse cuenta del objetivo de este taller es ver un ejemplo.

El código HTML

Vamos a partir básicamente de un enlace, al que asignaremos una clase definida con CSS. Como los estilos los vamos a aplicar con CSS, el enlace es tan sencillo como este:

```
<a href="#" class="enlaceboton">Hola!!</a>
```

El código CSS

Vamos a definir la clase "enlaceboton", que es el estilo que se ha asignado para el enlace. Como sabemos, los enlaces tienen diferentes estados (visitados, no visitados, o con el cursor del ratón por encima), así que tendremos que definir el estilo para cada estado.

```
.enlaceboton {    font-family: verdana, arial, sans-serif;
  font-size: 10pt;
  font-weight: bold;
  padding: 4px;
  background-color: #ffffcc;
  color: #666666;
  text-decoration: none;
}
.enlaceboton:link,
.enlaceboton:visited {
  border-top: 1px solid #cccccc;
  border-bottom: 2px solid #666666;
  border-left: 1px solid #cccccc;
  border-right: 2px solid #666666;
}
.enlaceboton:hover {
   border-bottom: 1px solid #cccccc;
  border-top: 2px solid #666666;
  border-right: 1px solid #cccccc;
  border-left: 2px solid #666666;
}
```

En principio, con la clase .enlaceboton se han definido estilos para cualquier estado del enlace. Después, para los diferentes estados del enlace se han definido sus correspondientes estilos, que son el mismo para los estados visitado y no visitado y diferentes para el estado hover, que es el que se aplica cuando está el ratón sobre el enlace.

Tan sencillo como esto. Si se desea, se puede ver el ejemplo en marcha.

*Artículo por **Miguel Angel Alvarez***

Crear un menú dinámico con CSS

El siguiente estilo nos permite crear un menú similar a los de javascript, en donde representamos su estado de reposo (out) con un color y su estado sobre (over) con otro.

```
<style type="text/css">
#menu div.barraMenu,
#menu div.barraMenu a.botonMenu {
font-family: sans-serif, Verdana, Arial;
font-size: 8pt;
color: white;
}

#menu div.barraMenu {
text-align: left;
}

#menu div.barraMenu a.botonMenu {
background-color: #556975;
color: white;
cursor: pointer;
padding: 4px 6px 2px 5px;
text-decoration: none;
}

#menu div.barraMenu a.botonMenu:hover {
```

```
background-color: #637D4D;
}

#menu div.barraMenu a.botonMenu:active {
background-color: #637D4D;
color: black;
}
</style>

<div id="menu"><div class="barraMenu">
<a class="botonMenu" href="">Opción 1</a>
<a class="botonMenu" href="">Opción 2</a>
<a class="botonMenu" href="">Opción 3</a>
<a class="botonMenu" href="">Opción 4</a>
</div></div>
background-color de a.botonMenu : color de estado reposo (out).
background-color de a.botonMenu:hover : color de estado sobre (over).
background-color de a.botonMenu:active : color de estado seleccionado.
```

Ver ejemplo en marcha

*Artículo por **Federico Elgarte***

Utilizar CSS para maquetar un boletín

La maquetación con CSS ofrece muchas ventajas para la accesibilidad de la página, carga en bytes y claridad del código. En este artículo vamos a contar cómo podemos aprovecharnos de esas ventajas también en los boletines de novedades enviados en formato HTML.

Para maquetar con CSS se utiliza una hoja de estilo en cascada, donde se especifica cualquier atributo de aspecto de la página, separando por completo el contenido y presentación. El contenido se define en el código HTML de la página y cualquier especificación del aspecto se incluye en un archivo externo CSS.

> **Referencia:** Para saber más de CSS podemos consultar la sección CSS a fondo. También podemos consultar acerca de la maquetación CSS.

Un newsletter, como se ha comentado en nuestro manual de boletines de novedades, se puede enviar en formato HTML para dar vistosidad al correo. En esos casos, a la hora de crear el boletín se debe hacer una página web normal y luego se enviará como cuerpo del mensaje.

Como cualquier otra página web, la que creamos para hacer el boletín, puede realizarse utilizando CSS, lo que redundará en ventajas para el creador del boletín y suscriptor.

- El peso del mensaje se podrá reducir, al no ser necesario incluir ninguna etiqueta HTML para definir los estilos. Esto nos ahorrará mucho código.
- Podremos cambiar el aspecto del mensaje con sólo cambiar la hoja de estilos, sin necesidad de modificar el código HTML que venimos utilizando para hacer el envío. Esto nos ofrece una mayor capacidad de innovación en el envío del mensaje.
- Si se pierde la hoja de estilos por cualquier razón o el sistema de correo del suscriptor no soporta CSS, simplemente verá el mensaje sin el aspecto definido por el creador del boletín. Pero por lo menos verá perfectamente el envío, con toda la información del mensaje, presentada con los estilos predeterminados del sistema del usuario.
- En caso de que a un usuario no soporte formato HTML, podría darse el caso de que

viese el código de la página (con las etiquetas y todo), en lugar de ver el formato web. En ese caso, por lo menos recibiría un código mucho más legible y comprensible por el suscriptor, que si estuviera mezclado con etiquetas HTML para definir los estilos.
- A la hora de crear los distintos boletines cada vez, se ahorra tiempo, puesto que no hay que preocuparse por definir los estilos. Es decir, la modificación es mucho más sencilla.

Como todo en esta vida, maquetar un boletín con CSS, también tiene algunas desventajas.

- La principal que veo es que hace falta tener mayores conocimientos para crear un boletín con CSS. Es decir, cualquier persona es capaz de hacer una página con HTML básico utilizando un editor de webs como Frontpage o Dreamweaver. Sin embargo, con CSS deberemos dominar una tecnología adicional y algún que otro programa para realizar el trabajo.
- Otra desventaja es que algunos sistemas de correo no incluyen enlaces con archivos externos, ya sean imágenes o declaraciones CSS. Esto hace que no muestren los estilos del boletín, aunque por lo menos se mostrará la página básica sin los estilos. Una posibilidad para evitar esto es incluir los estilos dentro del propio código de la página, aunque entonces estaremos contaminando un poco el código limpio de nuestro boletín.

Ejemplo de boletín en formato HTML con CSS

Nosotros llevamos tiempo enviando en nuestra web MercadoProfesional.com un boletín semanal en formato HTML, que está maquetado por completo con CSS. Vamos a mostrar aquí su código HTML y el código CSS que utilizamos para definir los estilos.

Sería bueno ver el boletín en una página aparte, para hacernos una idea previa del contenido y estilo creados.

Código HTML

Podemos ver a continuación el código HTML de uno de nuestros boletines de novedades.

```
<html>
<head>
   <title>Boletín de novedades 25 .:MercadoProfesional.com:.</title>
  <link rel="STYLESHEET" type="text/css" href="estilos.css">
</head>

<body>
<div id="container">
  <div id="cabecera">
   <div id = "titulo">
   <h1>Boletín de novedades<BR> MercadoProfesional.com</h1>
   </div>
   <div id="logo">
     <a href="http://www.mercadoprofesional.com/"><img
src="http://www.mercadoprofesional.com/mail_boletin_novedades/imgs/logo.gif" width = "155px" height = "78px"
border="0px" valign="top"></a>
    </div>
   </div>
   <div id="topCuerpo"></div>
   <div id="cuerpo">

    <div id="numBoletin"> Boletín de Novedades 25 # 18/05/2005 # </div>
   <div id="cuerpo1">
   <p>Saludos cordiales,
  </p>
  <p>
```

```
Te llega este correo por ser uno de los profesionales registrados en MercadoProfesional.com
Te informamos de las nuevas solicitudes que han llegado, para que no dejes pasar la oportunidad de enviar tus
presupuestos. </p>

</div>
    <div id="cuerpoNov"><h2 class="icoTit"> Nuevos Proyectos: </h2>
    <h3 class="icoNovedad"> <a href="http://www.mercadoprofesional.com/profesionales/solicitudes/364.php">
Tienda Online</a></h3>
    <p class="par">Tienda online, muy bien definida, para venta de servicios de revelado online</p>
    <h3 class="icoNovedad"> <a href="http://www.mercadoprofesional.com/profesionales/solicitudes/366.php">
Desarrollo de portal inmobiliario</a></h3>
    <p class="par">Como su nombre indica, un portal inmobiliario, hacen especial hincapié en la funcionalidad.</p>
    <h3 class="icoNovedad"> <a href="http://www.mercadoprofesional.com/profesionales/solicitudes/368.php">
Desarrollo de portal</a></h3>
    <p class="par">Desarrollo y promoción de un portal, exactamente, no sabemos que temática quieren utilizar.</p>
    <h3 class="icoNovedad"> <a href="http://www.mercadoprofesional.com/profesionales/solicitudes/369.php">
Formulario de flash y PHP</a></h3>
    <p class="par">Breve desarrollo en php para una aplicación en flash, con opciones de seguir colaborando en el
futuro.</p>
   </div>
   <p>Esta semana tenemos 4 proyectos nuevos.</p>
   <p>Simplemente despedirnos de todos vosotros y desearos suerte con los presupuestos.</p>
   <p>© Guiarte Multimedia S.L. - +34 915440837 - <a href
="mailto:info@mercadoprofesional.com">info@mercadoprofesional.com</a> -</p>
</div> <div id="pieCuerpo"></div>
</body>
</html>
```

Como este código no tiene ningún estilo definido a través de HTML, resulta bastante sencillo de interpretar.

Código CSS

Ahora podemos ver el código del archivo CSS que estamos utilizando para definir los estilos del documento. Seguramente alguno de los estilos definidos no lo estaremos utilizando en esta edición del boletín de novedades. No os estrañaros por eso. En general, no extrañarse si no está totalmente optimizada la declaración de estilos.

```
BODY {
    margin : 0 0 0 0px;
   background-color: #CCCCCC;
   font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
   text-align : center;
}

#cabecera {
    background-image: url(imgs/fondo_cab.gif);
    background-repeat : no-repeat;
    margin : 0 0 0 0px;
    background-position : right;
    padding : 0 0 0 0px;
    width : 500px;
    height: 96px;
}

#logo {
    padding : 7 20 11 20px;
}

#titulo {
    padding : 18 20 0 20px;
    float : right;
    margin-right: 24px;
```

```
}

#container {
   width : 500px;
   padding : 0 0 0 0px;
   margin : auto;
   text-align : left;
}

#topCuerpo {
   margin-bottom : 0px;
   margin-left : 0px;
   margin-right : 0px;
   margin-top : 5px;
   border-bottom : 1px solid #9b9b9b;
   width : 493px;
   float : right;
}

#cuerpo {
   font-family: Verdana, Arial, Helvetica, sans-serif;
   font-size: 10px;
   padding : 10 10 10 10px;
   background-image : url(imgs/fondo_cuerpo.gif);
   margin : 0 0 0 6px;
   background-position : right;
   background-repeat : repeat-y;
   clear : both;
}

A:ACTIVE{
   color : #003366;
}

A:HOVER{
   color : #003366;
   text-decoration : none;
}

A:LINK{
   color : #003366;
}

A:VISITED {
   color : #003366;
}

#numBoletin {
   background-color : #e9e9e9;
   border : 1px solid #666666;
   width : 304px;
   float : right;
   padding : 6 0 6 10px;
}

#cuerpo1 {
   clear : both;
   padding-top: 10px;
}

#cuerpoNov {
   background-color : #d2e3fb;
   border : 1px solid #666666;
   padding : 5 10 10 5px;
}
```

```
#pieCuerpo {
   background-image : url(imgs/bajo_cuerpo.gif);
   height:9px;
   background-repeat : no-repeat;
   margin-left : 6px;
}

H1 {
   font-size : 16px;
   font-weight : bold;
   color : #003366;
   text-align : center;
}

H2 {
   font-size : 11px;
   font-weight : bold;
   color : #003366;
}

.icoTit{
   background-image : url(imgs/IcoTit.gif);
   background-repeat : no-repeat;
   padding-left:12px;
   margin-top:0px;
   background-position : left;
}

H3 {
  font-size: 10px;
  font-weight : bold;
  color : #003366;
}
.icoNovedad{
   background-image : url(imgs/IcoNovedad.gif);
   background-repeat : no-repeat;
   padding-left:10px;
   margin-left:3px;
   margin-top:0px;
   margin-bottom:2px;
   background-position : left;
}

.par{
   margin-top:2px;
   padding-left:10px;
   margin-left:3px;
}

.resaltado {
   background-color : #e9e9e9;
   border : 1px solid #666666;
   padding : 6 0 6 10px;
}
```

La declaración de estilos resulta bastante larga, pero como decíamos, se puede podría optimizar bastante todavía.

Podemos ver el boletín en una página aparte.

Conclusión

El paso más difícil para realizar este boletín es hacer la maquetación propiamente dicha en CSS. Pero siempre resulta interesante perder un poco de tiempo para mejorar nuestra manera de hacer las cosas.

Antes de acabar, queremos poner un enlace a una página para visualizar el boletín sin la definición de estilos asociada.
Así es como se vería el boletín si por cualquier cuestión el sistema del suscriptor no acepta CSS o no se llega a enlazar con la hoja de estilos por cualquier motivo. Se verá que el boletín queda bastante comprensible, aunque no tan vistoso.

*Artículo por **Diego Pinilla***

Recuadro sencillo y elegante con CSS

En muchas ocasiones, al maquetar una página web, se necesita recuadrar una información para remarcar su contenido, destacándolo de otros textos del documento.

Es básicamente lo que vamos a hacer en el ejercicio de esta ocasión. Se trata de un ejemplo muy sencillo, pero que sirve como continuación de otro artículo que hemos publicado en el taller de HTML. (ENLACE A http://www.desarrolloweb.com/articulos/2025.php) En dicho taller realizábamos los recuadros utilizando únicamente HTML, sin declaración de estilos, lo que no es muy adecuado debido a las tendencias actuales.

CSS es, en estos momentos, la tecnología más adecuada para definir los estilos de un documento. Lo que antes habíamos realizado con HTML se puede realizar con CSS obteniendo varias ventajas de la maquetación con Hojas de Estilo en Cascada.

> **Nota:** Las ventajas y fundamentos de las Hojas de Estilo en Cascada (CSS) se pueden ver en el artículo Maquetar una web con CSS.

El ejercicio

Para empezar, sería bueno observar el objetivo de este ejercicio, para tener una idea exacta de lo que vamos a hacer.
El ejercicio dispone de tres recuadros con estilos distintos, aunque a pesar de ello tienen códigos muy similares. Para variar su aspecto, simplemente se cambia la declaración de estilos para cada uno.

La maquetación con estilos se realiza utilizando etiquetas `<DIV>` en lugar de tablas. Por eso el código incluye las etiquetas `<DIV>` necesarias y poco más.

Este sería el código de la primera tabla.

```
<div id=tabla1>
  <div id=cabtab1>
  Recuadro curioso con HTML
  </div>
  <div id=cuerpotab1>
  Este es el interior del recuadro. Esperamos que os resulte elegante... es muy sencillo.
  </div>
</div>
```

Contiene tres etiquetas <DIV> una para englobar el recuadro entero, y asignarle estilos como el borde o estilos que deseemos que se apliquen a todo el recuadro. También tendremos un <DIV> para el encabezamiento del recuadro y otro para el cuerpo.

Los estilos que utilizamos para este recuadro son los siguientes.

```
#tabla1{
  border: 1px solid #1E679A;
  width: 280px;
}
#cabtab1{
  background-color: #1E679A;
  font-weight: bold;
  color: #ffffff;
  padding: 2 2 2 2px;
}
#cuerpotab1{
  padding: 4 4 4 4px;
  background-color: #ffffcc;
}
```

Cada <DIV> tiene asignado un estilo distinto dependiendo de nuestras necesidades.

Para el segundo recuadro podremos ver un código HTML casi idéntico. Lo único que cambiamos son los identificadores de los <DIV> , para poder asignar unos estilos distintos al recuadro.

```
<div id=tabla2>
  <div id=cabtab2>
  Recuadro curioso con HTML
  </div>
  <div id=cuerpotab2>
  Este es el interior del recuadro. Esperamos que os resulte elegante... es muy sencillo.
  </div>
</div>
```

El código CSS para definir el aspecto es el siguiente.

```
#tabla2{
  border: 1px solid #165480;
  width: 200px;
}
#cabtab2{
  background-color: #5fa6d7;
  font-weight: bold;
  font-size: 8pt;
  padding: 2 2 2 2px;
}
#cuerpotab2{
  font-size: 8pt;
  padding: 4 4 4 4px;
  background-color: #ffffcc;
}
```

Como se puede ver, no tiene ninguna dificultad adicional con respecto al primer ejemplo, pues sólo se definen estilos para cada uno de los <DIV> .

En el tercer recuadro hemos complicado un poquito el código, aunque nada reviste ninguna complicación. En este caso, como el recuadro contenía un texto con varias opciones de una lista, hemos incluido, dentro del cuerpo del recuadro, un <ul> (unordered list) con cada una de las opciones a visualizar.

```
<div id=tabla3>
  <div id=cabtab3>
  Recuadro curioso con HTML
  </div>
  <div id=cuerpotab3>
   <ul>
     <li>Opción uno</li>
     <li>Otra opción con texto en varias líneas</li>
     <li>Lo que sea que desees destacar</li>
     <li>Última opción</li>
   </ul>
  </div>
</div>
```

En la declaración de estilos también hemos definido el aspecto de la lista, para que se ajuste a nuestras necesidades.

```
#tabla3{
  border: 1px solid #80A93E;
  width: 200px;
}
#cabtab3{
  background-color: #B7F259;
  font-weight: bold;
  font-size: 8pt;
  padding: 2 2 2 2px;
}
#cuerpotab3{
  font-size: 8pt;
  padding: 4 4 4 4px;
  background-color: #F5ECB9;
}
#cuerpotab3 ul{
  margin: 0 2 0 20px;
  padding: 0 0 0 0px;
}
#cuerpotab3 li{
  margin: 0 2 0 2px;
  padding: 0 0 0 0px;
}
```

Para definir el estilo de la lista indicamos el identificador del `<DIV>` donde se encuentra la lista, seguido de la etiqueta sobre la que deseamos declarar los estilos, en este caso "ul" para definir los estilos de la lista y "li" para declarar los estilos de cada una de las opciones. En este ejemplo hemos declarado los estilos necesarios para definir un margen adecuado para la lista y para cada una de sus opciones.

Conclusión

Hemos visto una manera sencilla de hacer cajas con CSS. Tal vez este artículo es demasiado básico, pero se trataba de mostrar cómo se pueden hacer con CSS algunas cosas que habíamos hecho previamente con sólo HTML.

Como se puede ver, comparando este ejemplo con su contrapartida en HTML , Con CSS se maqueta con mucha más coherencia y se obtiene un código mucho más claro.

*Artículo por **Miguel Angel Alvarez***

Decorar un campo select de formulario con CSS

CSS ofrece infinitas opciones para decorar todos los elementos soportados por html. Esta vez mostraré como aplicar nuestro estilo personalizado a los drop down menú.

Primero definiremos el tag option, que contendrá el estilo de letra, el tamaño, el color, etc...

```
option {font-family: verdana; font-size: 10px; color: white}
```

Luego definiremos dos estilos vinculados a option que contendrán los colores de fondo de cada opción:

```
option.uno {background-color: #CCC}
option.dos {background-color: #666}
```

El último paso es colocar el drop down menú con nuestro estilo personalizado:

```
<select>
<option class="uno">Opción</option>
<option class="dos">Opción</option>
<option class="uno">Opción</option>
<option class="dos">Opción</option>
<option class="uno">Opción</option>
<option class="dos">Opción</option>
</select>
```

Aparte de asignar estilos a los option, también se deben definir los estilos del campo select en se. será necesario hacerlo así, por lo menos, para el navegador Firefox y otros de la familia de Mozilla.

Para definir el estilo del campo select utilizaremos este código CSS:

```
SELECT{ font-family: verdana; font-size: 10px; color: white; background-color:#666;}
```

Podemos ver el ejemplo en marcha.

*Artículo por **Federico Elgarte***

Esconder con CSS el email a los spambots

Cuando publicamos una dirección de correo en una página web debemos saber que no tardará mucho en ser rastreada e incorporada a bases de datos de emails para hacer spam. Es una auténtica paliza tener que recibir decenas o cientos de mensajes basura al cabo del día o la semana, así que merece la pena poner en marcha algún mecanismo para evitar que los spambots (robots en busca de direcciones de email) cacen nuestro correo electrónico.

En DesarrolloWeb.com ya publicamos algunos trucos posibles para evitar que encuentren nuestro email escrito en la web, en el artículo Ocultar un email de un enlace para evitar el spam. Ahora vamos a mostrar otro mecanismo que hemos encontrado en una página web. En concreto, este artículo es una traducción libre de este otro artículo en inglés: Hiding email address from spambots, escrito por Lim Chee Aun.

En este caso vamos a mostrar un código que serviría para mostrar por CSS la dirección de correo electrónico. El email aparece en la hoja de estilos, nunca en el cuerpo de la página, por lo que el spambot lo va a tener muy difícil para obtener nuestro correo.

Se trata de utilizar unas características avanzadas de las hojas de estilo en cascada, que permiten definir cierto contenido, en este caso una dirección de correo electrónico, para colocar antes o después de un texto.

Concretamente vamos a utilizar CSS2 (hojas de estilo en cascada especificación 2), que incluye la definición de pseudo-elements (pseudo elementos) "before" y "after", que sirven para insertar contenidos antes y después de ciertos elementos.

En este caso, vamos a definir con CSS 2 la inclusión de un contenido después de una etiqueta HTML, en concreto la etiqueta <ADDRESS>, que sirve en principio para escribir una dirección en una página.

> **Nota:** Conocimos los pseudo-element en un artículo del manual de CSS: Pseudo-element en CSS

El código CSS sería el siguiente:

```
address:after{
/* \40 es un código para escribir el caracter '@' */
content: " <nombre\40 dominio.com>";
}
```

> **Nota:** El carácter @ en hojas de estilo en cascada se puede escribir con el código especial \40. Ponemos un espacio después de \40 para que quede claro que el carácter especial llega hasta allí. Podéis probar a quitar el espacio y veréis como en ocasiones la @ se transforma en otro carácter, dependiendo de lo que se haya escrito después. Ese espacio en blanco no afecta al texto, es decir, no se verá en la página.

El código HTML que deberíamos escribir para mostrar la dirección de correo sería el siguiente:

```
<address>© 2005 loquesea.com</address>
```

Como se puede ver, en el código HTML no aparece la dirección de correo electrónico en ninguna parte, con lo que el spambot no se enterará de que allí se muestra un correo.

Se puede ver el ejemplo en una página aparte.

Atención los usuarios de Internet Explorer 6. Este navegador no soporta los pseudo-elementos after o before, con lo que este ejemplo no funcionará.

En nuestro ejemplo hemos incorporado la declaración de estilos en el mismo fichero HTML, pero tal vez sería más efectivo si colocásemos la declaración de estilos en un archivo externo, que luego incluiríamos con la etiqueta en la cabecera de la página.

Conclusión

Hemos podido ver otra ingeniosa manera de ocultar la dirección de email. Sin embargo, cabe destacar que las características avanzadas de CSS2 no están soportadas por todos los navegadores.

En general, desventaja a destacar es que, con este código, la accesibilidad de la página disminuye considerablemente. Puesto que sólo ciertos navegadores mostrarán las direcciones

de correo electrónico. Asimismo, esas direcciones no se podrán pinchar para enviar un correo electrónico directamente, es decir, no se pueden mostrar como enlaces. En Firefox, ni siquiera podemos seleccionar el texto de la dirección de correo, simplemente se nos permite verlo.

No obstante, es una nueva manera de mostrar un correo electrónico no accesible a programas de rastreo de emails. Hoy todavía no es una manera muy adecuada, pero tal vez lo será con el tiempo.

Recordamos que este artículo es una traducción de otro, escrito en inglés y publicado en la URL: http://www.phoenity.com/newtedge/hide_email_spambots/

*Artículo por **Lim Chee Aun***

Efecto de sombra con CSS

He encontrado 3 métodos distintos para hacer un efecto de sombra. Son los siguientes, que veremos comentados en este artículo:

1. Método background-color
2. Método canal alpha
3. Método estirar imagen

Método Background color

Este método es bastante simple. Se basa en definir tres cajas trasladadas, con diversos colores de fondo. Los elementos con las clases "blur" y "shadow" se definirán con tonos grisáceos para crear el efecto de sombreado.

Código HTML:

```
<div class="blur">
<div class="shadow">
<div class="content">bla bla</div>
</div>
</div>
```

Código CSS:

```
.blur{
  background-color: #ccc; /*shadow color*/
  color: inherit;
  margin-left: 4px;
  margin-top: 4px;
}

.shadow,
.content{
  position: relative;
  bottom: 2px;
  right: 2px;
}

.shadow{
  background-color: #666; /*shadow color*/
  color: inherit;
```

```
}

.content{
  background-color: #fff; /*background color of content*/
  color: #000; /*text color of content*/
  border: 1px solid #000; /*border color*/
  padding: .5em 2ex;
}
```

La única desventaja de este método es que usa colores definidos para las sombras y ello puede dar lugar a que no se pueda mezclar este efecto con otros elementos. Con un fondo blanco, las sombras en grises quedan bien, pero pongamos el caso de que el fondo de la página fuera de color rojo. Entonces el efecto de sombreado debería realizarse con tonos rojos oscuros.

Podemos ver el ejemplo en funcionamiento en una página aparte.

Método canal alpha

Este método es muy parecido al anterior, pero soluciona el problema de mezclarse con otros elementos. Es indiferente cual sea el fondo de la página donde se va a mostrar el elemento sombreado, incluso si el sombreado se realiza en la misma página sobre distintos fondos. Utiliza imágenes de fondo en formato PNG "alpha transparentes", en lugar de colores definidos en la hoja de estilo.

El código HTML necesario es el mismo que para el ejemplo anterior, simplemente debes modificar el código CSS, en concreto para las clases "blur" y "shadow". Mostramos el código CSS para este ejemplo.

```
<style type=text/css>
.blur{
  background: transparent url(shadow1.png);
/*ruta para el 80%-transparente 1x1 pixel coloreado de negro */
  color: inherit;
  margin-left: 4px;
  margin-top: 4px;
}

.shadow{
  background: transparent url(shadow2.png);
/*ruta para el 60%-transparent 1x1pixel coloreado de negro */
  color: inherit;
}

.shadow,
.content{
  position: relative;
  bottom: 2px;
  right: 2px;
}

.content{
  background-color: #fff; /*background color of content*/
  color: #000; /*text color of content*/
  border: 1px solid #000; /*border color*/
  padding: .5em 2ex;
}
</style>
```

Para probar la ventaja de este tipo de fondo, podemos cambiar el color de fondo de la página web y veremos como la sombra también cambia de color.

Las dos imágenes, de 1x1 pixel medio transparentes que se necesitan para componer este ejemplo están aquí:

Pulsar para descargar imagenes-transparentes.zip

Podemos ver el ejemplo en funcionamiento en una página aparte .

Método estirar imagen

Opino que los dos métodos anteriores no son demasiado buenos, debido a que la sombra no parece muy natural. Dicho de otro modo, no resulta un efecto suficientemente realista. De modo que abro mi editor gráfico, creo una caja rectangular con efecto de sombra y lo exporto a una imagen. Posiblemente pueda utilizar esa imagen para crear el efecto de sombra.

El código HTML experimental

```
<div class="shade">
<img src="shadow.png" width="0" height="0" alt="" class="shade" />
bla bla</div>
```

El código CSS experimental

```
img.shade{
   width: 37ex;
   height: 9em;
/* specify the dimension of the image */
   display: block;
   position: absolute;
   z-index: -1;
/* force the image to show below the content */
   right: -3ex;
   bottom: -1em;
}

div.shade{
   width: 30ex;
   height: 6em;
/* specify the dimension of the content, slightly smaller than the image */
   position: relative;
   z-index: 1;
/* force the content to show above the image */
   background-color: #fff;
   border: 1px solid #000;
   padding: 1em 2ex;
   margin-right: 6ex;
   margin-bottom: 3em;
}
```

Tenemos tres desventajas en este método

1. Como la imagen se estira, puede que no quede muy bonito.
2. En Mozilla Firefox la imagen a veces desaparece (se puede recuperar refrescando o desplazando la página). En Internet Explorer no se muestra bien el efecto, por lo menos en la versión 6.
3. El contenido no puede flotar (no podemos utilizar el atributo float)

Podemos obtener la imagen para hacer el ejemplo.

Podemos ver este ejemplo en marcha en una página aparte.

Un momento. ¿Cómo haríamos texto con sombreado?

Si utilizas un navegador basado en Gecko, podrías visualizar otro efecto interesante para realizar sombreado de textos sin utilizar imágenes y redimensionable simplemente cambiando el tamaño del texto o las fuentes que usa el navegador (con el menú view>text size > increase / Decrease).

El código HTML sería el siguiente:

```
<span id="text">Texto sombreado</span>
```

El código CSS

```
#text{
  font-size: 3em; /* optional. just to increase the font size. */
  display: block;
  line-height: 1em;
  color: #666; /* shadow color */
  background-color: transparent;
  white-space: nowrap; /* wrapping breaks the effect */
}

#text:before,
#text:after{
  content: "Texto sombreado"; /* El mismo texto que queramos mostrar sombreado */
  display: block;
}

#text:before{
  margin-bottom: -1.05em;
  margin-left: 0.1ex;
  color: #ccc; /* shadow color */
  background-color: transparent;
}

#text:after{
  margin-top: -1.05em;
  margin-left: -0.1ex;
  color: #fff; /* text color */
  background-color: transparent;
}
```

Este efecto puede ser útil por ahora. Sin embargo, las especificaciones de CSS2 incluyen una propiedad CSS llamada text-shadow, que sirve para definir un efecto de sombra a un texto. Sin embargo, la mayoría de los navegadores todavía no soportan esta propiedad.

Interesantes recursos sobre este tema:

- Simple CSS drop shadows http://www.saila.com/usage/shadow/
- A List Apart: CSS Drop Shadows http://alistapart.com/articles/cssdropshadows/ de Sergio Villarreal (revisión del artículo Easy CSS drop shadows http://1976design.com/blog/archive/2003/11/14/shadows/ de Dunstan Orchard)
- CSS Drop Shadows II: Fuzzy Shadows http://alistapart.com/articles/cssdrop2/ by Sergio Villarreal

Este artículo es una traducción del inglés. El original está en:
http://www.phoenity.com/newtedge/drop_shadow/

*Artículo por **Lim Chee Aun***

Texto en vertical usando CSS

Junto con CSS3 llegan nuevos atributos para aplicar a nuestros documentos html.

Este es el turno de presentar writing-mode, que permite enderezar un texto en dirección vertical.

Lamentablemente el único explorador en soportarlo es Internet Explorer, por eso ten precaución en su uso.

```
<style>
#textovertical {writing-mode: tb-rl; filter: flipv fliph}
</style>
```

Puedes ver un ejemplo en marcha en una página aparte.

Actualmente utilizando CSS no podremos solucionar este problema de una manera adecuada para todos los entornos. Dado que este ejemplo sólo funciona con Internet Explorer, otros navegadores como Firefox u Opera verán el texto en horizontal. Si nuestro diseño es muy preciso, seguramente no podamos utilizar este ejemplo, porque si el texto aparece unas veces en hortizontal y otras en vertical, seguramente acabe descuadrándo la página.

Una posible solución a este problema sería guardar el texto como una imagen, modificarla con Photoshop o con cualquier otro programa de este tipo, para colocar el texto en vertical. Luego habría que añadirla a la web tal como cualquier otra imagen. De esta forma el texto quedaría en vertical del mismo modo que con CSS y funcionaría en todos los navegadores.

*Artículo por **Federico Elgarte***

Maquetación CSS a dos columnas

Vamos a ver cómo realizar una maquetación a dos columnas con CSS, sin utilizar tablas. Además de las dos columnas, para completar la estructura típica de una web, colocaremos una cabecera y un pie de página.

El ejemplo pretende ser el inicio de una serie de artículos para mostrar cómo realizar distintos tipos de plantillas, maquetando con CSS en lugar de tablas. Iremos publicando estos artículos en nuestro Taller de CSS.

Empezamos mostrando los dos ejemplos de maquetación que veremos en este artículo, siempre con dos columnas, dejando la columna con los enlaces de la barra de navegación a la izquierda o la derecha.

- Maquetación con dos columnas y enlaces a la izquierda
- Maquetación con dos columnas y enlaces a la derecha

El código HTML

El código HTML de los dos ejemplos que hemos adelantado es el mismo. Básicamente este:

```
<div id="contenedor">
  <div id="cabecera">
    Cabecera 01
  </div>
  <div id="cuerpo">
   <div id="lateral">
    <ul>
      <li><a href="#">Enlace 1</a>
      <li><a href="#">Vínculo 2</a>
      <li><a href="#">Otro enlace</a>
      <li><a href="#">Link chulo</a>
      <li><a href="#">Más enlaces</a>
      <li><a href="#">Otro último</a>
    </ul>
   </div>
   <div id="principal">
    Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
    ......
   </div>
  </div>
  <div id="pie">
  © 2005 DesarrolloWeb.com
  </div>
</div>
```

Se puede ver que tenemos una capa contenedor, que engloba todo el código. Luego, dentro del contenedor tenemos tres bloques. La cabecera, el cuerpo y el pie de página. La cabecera y el pie de página son dos capas tal cual, que ocupan todo el espacio del contenedor. El lugar donde tendremos las dos columnas es el cuerpo.

Dentro del cuerpo tenemos dos partes, para codificar cada una de las dos columnas. Una parte, que hemos llamado "lateral", con una lista de enlaces (sería la barra de navegación) y otra parte con el texto de la página, que hemos llamado "principal".

El código CSS

Como habíamos adelantado, veremos dos variantes de codificación a dos columnas, con los enlaces a la izquierda y a la derecha. No obstante, la mayor parte del código CSS de ambos ejemplos es la misma, pues sólo varía la declaración de estilos de la capa "lateral" y de la capa "principal".

Maquetar con los enlaces a la izquierda

Veamos la codificación CSS para la página con los enlaces a la izquierda.

```
<style type="text/css">
BODY {
   font: 8pt Verdana, Geneva, Arial, Helvetica, sans-serif;
   margin: 10 0 10 0px;
   text-align: center;
   background-color: #ebebeb;
}
#contenedor{
   text-align: left;
   width: 770px;
```

```
  margin: auto;
}
#cabecera{
  background-color: #d0d0ff;
  color: #333300;
  font-size:12pt;
  font-weight: bold;
  padding: 3 3 3 10px;
}
#cuerpo{
  margin: 10 0 10 0px;
}
#lateral{
  width: 160px;
  background-color: #999999;
  float:left;
}
#lateral ul{
  margin : 0 0 0 0px;
  padding: 0 0 0 0px;
  list-style: none;
}
#lateral li{
  background-color: #ffffcc;
  margin: 2 2 2 2px;
  padding: 2 2 2 2px;
  font-weight: bold;
}
#lateral a{
  color: #3333cc;
  text-decoration: none;
}
#principal{
  margin-left: 170px;
  background-color: #ffffff;
  padding: 4 4 4 4px;
}
#pie{
  background-color: #cccccc;
  padding: 3 10 3 10px;
  text-align:right;
}
```

La parte que vamos a remarcar es donde se define el estilo del lateral y la capa principal. El lateral hacemos que tenga un tamaño fijo de 160 pixel, pero lo más importante es que hacemos que "flote" a la izquierda con float:left;. Esto hace que el lateral se quede en la izquierda y permite que el contenido insertado después del lateral se coloque a la misma altura, pero a la derecha.

Por su parte, para la capa principal, simplemente se indica que tiene un margen a la izquierda de 170 píxeles. Esto hace que se coloque al lado de la capa lateral, dejando un espacio en blanco de 10 píxeles. Tiene un margen de 170, de los que 160 son para el espacio que va a ocupar la capa de los enlaces y 10 píxeles de espacio entre la capa principal y la lateral.

Como la capa principal no tiene definida su anchura, se hará tan grande como lo permita el contenedor.

> **Nota:** Estas explicaciones no son completas de todo el ejercicio. Se supone que el lector ya tiene asimilados algunos conceptos que se han explicado en el Manual de CSS o en el Taller de CSS.

Podemos ver el ejemplo en marcha en una página aparte. Como es un archivo HTML, podemos

ver el código fuente para ver cómo queda el conjunto de maquetación y declaración de estilos completo.

Maquetar con los enlaces a la derecha

Continuamos mostrando los cambios que habría que hacer para maquetar la página con la columna de enlaces a la derecha. Simplemente vamos a cambiar el código CSS de las capas lateral y principal.

```
#lateral{
  width: 160px;
  background-color: #999999;
  float:right;
}
#principal{
  background-color: #ffffff;
  padding: 4 4 4 4px;
  margin-right: 170px;
}
```

Ahora la capa lateral estamos indicando que flote a la derecha. Por su parte, en la capa principal hemos cambiado el margen que había antes hacia la izquierda para ponerlo en la parte de la derecha.

Podemos ver el ejemplo en una página aparte.

Diseño fluido

Hasta aquí en este artículo hemos visto cómo hacer un diseño con una anchura fija. Si queremos un diseño fluido (que se ajusta a la anchura de la ventana del navegador), bastaría con quitarle al contenedor el atributo width: 770px;. Si no tiene definida una anchura, la capa contenedor se ajustará al tamaño de la ventana del navegador que tenga el visitante.

Además, tendremos que darle un margen al BODY, para que el contenedor no se acople por completo al borde de la ventana. Por ejemplo, podemos darle un margen de 10 píxeles a cada lado.

Tendríamos ahora esta declaración de estilos para el BODY y la capa "contenedor":

```
BODY {
  font: 8pt Verdana, Geneva, Arial, Helvetica, sans-serif;
  margin: 10 10 10 10px;
  text-align: center;
  background-color: #ebebeb;
}
#contenedor{
  text-align: left;
  margin: auto;
}
```

Podemos ver el ejemplo en una página aparte.

*Artículo por **Miguel Angel Alvarez***

Maquetación CSS a tres columnas

Continuando nuestros talleres de CSS, vamos a ver ahora como hacer una estructura de tres columnas para una página web. Esta es una estructura bastante típica de portal. En una de las columnas se suele situar la barra de navegación, en otra el contenido y en la última una serie de banners con promociones.

Sería bueno ver el resultado que buscamos en una página aparte.

Este artículo va a presuponer que el lector comprende ya la maquetación con CSS y que ha leído el artículo Maquetación CSS a dos columnas. Nos basaremos en ese artículo para componer la página con CSS a tres columnas.

En líneas generales, la posibilidad que vamos a explorar a continuación para maquetar una web con tres columnas, consiste en lo siguiente: Pondremos la columna de la izquierda flotando a la izquierda, la columna de la derecha flotando a la derecha y por último pondremos la parte principal, que aparecerá en el centro de la página.

El código HTML para hacer este ejemplo será el siguiente:

```
<div id="contenedor">
   <div id="cabecera">
     Cabecera 01
   </div>
   <div id="cuerpo">
     <div id="lateral">
       <ul>
         <li><a href="#">Enlace 1</a>
         <li><a href="#">Vínculo 2</a>
         <li><a href="#">Otro enlace</a>
         <li><a href="#">Link chulo</a>
         <li><a href="#">Más enlaces</a>
         <li><a href="#">Otro último</a>
       </ul>
     </div>
     <div id="otrolado">
       <img src="bannerlateral.gif" width="120" height="600" alt="">
     </div>
     <div id="principal">
       Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nulla condimentum commodo orci. Nulla eget purus
nec massa
       ...
     </div>
   </div>
   <div id="pie">
     © 2005 DesarrolloWeb.com
   </div>
</div>
```

Vemos que la página contiene tres partes, la cabecera, el cuerpo y el pie. La cabecera y el pie se colocarán en el documento ocupando todo el ancho disponible. La parte donde colocaremos las tres columnas es el cuerpo.

Dentro del cuerpo, como podemos ver, tenemos tres capas. La capa "lateral", que es la que hemos pensado colocar a la izquierda. Luego está la capa "otrolado", que es la que planeamos colocar a la derecha. Por último está la capa "principal", que es la parte central. La capa "principal" aparecerá en el centro, porque a los dos lados estarán ocupados por las capas

laterales.

El CSS que vamos a utilizar para maquetar esto será el siguiente. Se parece mucho al ejemplo de maquetación CSS a dos columnas. Luego lo comentaremos.

```
BODY {
   font: 8pt Verdana, Geneva, Arial, Helvetica, sans-serif;
   margin: 10 0 10 0px;
   text-align: center;
   background-color: #ebebeb;
}
#contenedor{
   text-align: left;
   width: 770px;
   margin: auto;
}
#cabecera{
   background-color: #d0d0ff;
   color: #333300;
   font-size:12pt;
   font-weight: bold;
   padding: 3 3 3 10px;
}
#cuerpo{
   margin: 10 0 10 0px;
}
#lateral{
   width: 160px;
   background-color: #999999;
   float:left;
}
#lateral ul{
   margin : 0 0 0 0px;
   padding: 0 0 0 0px;
   list-style: none;
}
#lateral li{
   background-color: #ffffcc;
   margin: 2 2 2 2px;
   padding: 2 2 2 2px;
   font-weight: bold;
}
#lateral a{
   color: #3333cc;
   text-decoration: none;
}
#otrolado{
   width: 120px;
   float: right;
}
#principal{
   margin-left: 170px;
   background-color: #ffffff;
   padding: 4 4 4 4px;
   width: 460px;
}
#pie{
   background-color: #cccccc;
   padding: 3 10 3 10px;
   text-align:right;
   clear: both;
}
```

Tendremos un contenedor, de 770 píxeles (px) de ancho, que es donde vamos a colocar todas

las capas. Nos centramos en explicar la zona del cuerpo, que es donde aparecerán las tres columnas.

Vemos como la capa "lateral" tiene definido un ancho de 160 px, y un float: left; para que flote a la izquierda. Vemos luego como la capa "otrolado" tiene 120 px de ancho y flota a la derecha.

Luego vemos la capa "principal", que tiene un margen a la izquierda de 170 px, para dejar un espacio con respecto a la capa "lateral". 170 px porque la capa "lateral" ocupa 160 px de ancho, más 10 px que es lo que realmente estamos poniendo de margen. En la capa "principal" también hemos definido un ancho de 460 px, para que ocupe justo el espacio que queda en el centro.

Se puede ver el ejemplo en marcha en una página aparte.

Diseño fluido

Los diseños fluidos son los que se ajustan a la anchura que tengamos en la ventana del navegador. En el anterior ejemplo el diseño tenía una anchura fija de 700 píxeles y ahora vamos a hacer un par de modificaciones para que el diseño se ajuste a la ventana del navegador.

Simplemente tendremos que quitar la definición de anchura de la capa "contenedor".

```
#contenedor{
   text-align: left;
   margin: auto;
}
```

Luego, también quitaremos la definición de anchura de la capa "principal" y añadiremos el atributo margin-right: 130px; para que la capa con el contenido central tenga un margen con respecto a la capa que queda a la derecha. 130 px porque la capa de la derecha ocupaba 120 px, más 10 px que es realmente el margen que estamos dejando.

```
#principal{
   margin-left: 170px;
   background-color: #ffffff;
   padding: 4 4 4 4px;
   margin-right: 130px;
}
```

Podemos ver el ejemplo en una página aparte.

*Artículo por **Miguel Angel Alvarez***

Variación de la maquetación con CSS a 3 columnas

Ya vimos en un artículo anterior una manera de realizar una maquetación con CSS a tres columnas. Ahora vamos a realizar unos cambios sencillos para mejorar algo el código del anterior artículo.

Antes que nada habría que explicar por qué estimamos que puede mejorarse el ejercicio de maquetación a tres columnas, tal y como fue planteado anteriormente. Se trata simplemente de mejorar la organización del contenido de la página de una manera más inteligente.

A mi me gusta siempre ver el aspecto que tiene una página web sin la hoja de estilo en cascada, para ver si la información tiene sentido y orden tal y como se ha colocado en el código HTML. Es decir, si está presentada de una forma adecuada, aunque no se estén visualizando los estilos definidos en el CSS.

Cualquier navegador que no tenga soporte a CSS mostrará la página sin ningún estilo y todos los elementos aparecerán uno detrás de otro según se hayan colocado. No es habitual que nos visite una persona que tenga un navegador incompatible con hojas de estilo en cascada, pero para mejorar la accesibilidad de las páginas, también conviene que se lean bien, aunque no se tenga acceso a CSS.

Así se vería la **maquetación CSS a tres columnas sin la declaración de estilos** que habíamos presentado en el anterior ejemplo. Si pulsamos el enlace podremos comprobar que aparece el banner vertical en la parte superior de la página, cuando lo interesante sería que apareciera debajo del contenido, por lo menos en mi opinión.

Variación al ejercicio anterior para maquetar a tres columnas

Básicamente vamos a hacer una maquetación a dos columnas, la izquierda, con la barra de navegación y la de la derecha, con el otro contenido. En la parte de la derecha, a su vez, haremos dos columnas más, una a la izquierda que tendrá el texto de la página y otra a la derecha, con la barra lateral derecha.

En realidad, el ejercicio queda muy similar. Veamos el código HTML:

```
<div id="contenedor">
  <div id="cabecera">
    Cabecera 01
  </div>
  <div id="cuerpo">
    <div id="lateral">
      <ul>
        <li><a href="#">Enlace 1</a>
        <li><a href="#">Vínculo 2</a>
        <li><a href="#">Otro enlace</a>
        <li><a href="#">Link chulo</a>
        <li><a href="#">Más enlaces</a>
        <li><a href="#">Otro último</a>
      </ul>
    </div>
    <div id="derecha">
      <div id="principal">
        Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
        ......
      </div>
      <div id="otrolado">
        <img src="bannerlateral.gif" width="120" height="600" alt="">
      </div>
    </div>
  </div>
  <div id="pie">
    © 2005 DesarrolloWeb.com
  </div>
</div>
```

La capa "derecha" es la que hemos creado nueva, donde colocamos tanto el texto principal como la capa del banner vertical.

El código CSS que hemos variado sólo afecta a la capa "derecha", que no estaba creada

anteriormente y a la capa "principal", que es donde está el texto central.

```
#derecha{
   margin: 0 0 0 170px;
}

#principal{
   background-color: #ffffff;
   padding: 4 4 4 4px;
   width: 460px;
   float: left;
}
```

La capa "derecha" tiene un margen de 170 pixels, para dejar espacio a la barra de enlaces de la izquierda. La capa "principal", que antes tenía ese margen de 170 píxeles, ya no lo necesita, pero en cambio sí que hemos puesto el atributo float: left, para que se flote en la parte de la izquierda.

En definitiva, es una maquetación a dos columnas, en las que una de las columnas tiene a su vez otra maquetación en dos columnas.

Se puede ver el ejemplo en marcha en una página aparte. El código fuente del ejemplo entero no lo vamos a escribir. Pero se puede analizar tanto el HTML como el CSS a través de la opción "ver código fuente" del navegador.

Se puede ver cómo quedaría esta maquetación a tres columnas si nuestro navegador no fuera compatible con CSS.

*Artículo por **Miguel Angel Alvarez***

cambiar el cursor con css del ratón

Nuestro sistema operativo posee cursores por defecto. Como cursores nos referimos al puntero del ratón: la típica flechita o cualquier otro dibujo que puede tener. En este artículo veremos cómo se puede cambiar el cursor CSS de un elemento de la página, esto es, cambiar la imagen del cursor al posicionarse con el ratón sobre elementos de la página que nosotros queramos.

En Windows podemos encontrar varios cursores que se activan cuando pasamos por “ciertas zonas” de nuestra pantalla. Por ejemplo, cuando nos posicionamos sobre un link, la típica flechita (denominada default en CSS) cambia por la manito (pointer). Como en el ejemplo anterior, podemos encontrar decenas de situaciones en donde nuestro puntero cambia de imagen.

Con la ayuda de las hojas de estilo en cascada, podemos forzar a nuestro sistema navegador que no se rija con las normas convencionales de los punteros, logrando así un atractivo diseño web en donde el puntero de nuestro mouse puede llegar a ser hasta una imagen de nuestra propia creación.

> **Nota:** Aunque CSS nos ofrezca la posibilidad de cambiar el aspecto del cursor del ratón, esto es una acción que tenemos que llevar a cabo con bastante cuidado, desde el punto de vista de la usabilidad. Debemos saber que las personas están acostumbradas a visualizar ciertos tipos de cursor en ciertas ocasiones. Por ejemplo, al situarse sobre un enlace aparece siempre la mano, o al ponerse al borde de una ventana el cursor que aparece son la línea con flechas en los lados. Todos estos cursores tienen un significado que las personas conocen y no debemos cambiarlo a la ligera, a no ser que sea para ayudar, porque podemos liar a

los usuarios en el uso con la interfaz de la página web.

A continuación presentamos una la lista de los cursores disponibles por defecto en Windows, y que también tendremos disponibles con CSS. En este listado aparece el nombre del cursor (nombre de la pesudoclase CSS que tenemos que utilizar para cambiar el cursor) y el tipo de cursor que se trata.

- default (flecha)
- crosshair (cruz)
- e-resize (flecha que apunta a la derecha)
- hand (mano)
- help (signo de pregunta)
- move (cruz con flechas en los extremos)
- n-resize (flecha que apunta hacia arriba)
- ne-resize (flecha que apunta al noreste)
- nw-resize (flecha que apunta al noroeste)
- pointer (mano)
- s-resize (flecha que apunta hacia abajo)
- se-resize (flecha que apunta hacia el sudeste)
- sw-resize (flecha que apunta hacia el sudoeste)
- text (I-beam)
- w-resize (flecha que apunta a la izquierda)
- wait (reloj de arena)

Al igual que todas las propiedades del lenguaje CSS, es posible definir el objeto aplicándolo a todo el documento o solo a una parte del mismo.

A todo el documento:

```
<html>
<title>Cambiar el cursor</title>
<head>
<style type="text/css">
<!--
body {cursor: pointer}
-->
</style>
</head>
<body>
</body>
</html>
```

A algunos sectores del documento:

```
<html>
<title>Cambiar el cursor</title>
<head>
</head>
<body>
<h4 style="cursor: default">default</h4>
<h4 style="cursor: crosshair">crosshair</h4>
<h4 style="cursor: pointer">pointer</h4>
<h4 style="cursor: move">move</h4>
<h4 style="cursor: nw-resize">nw-resize</h4>
<h4 style="cursor: ne-resize">ne-resize</h4>
<h4 style="cursor: n-resize">n-resize</h4>
<h4 style="cursor: e-resize">e-resize</h4>
<h4 style="cursor: help">help</h4>
```

```
<h4 style="cursor: text">text</h4>
<h4 style="cursor: wait">wait</h4>
</body>
</html>
```

También es posible utilizar un cursor personalizado:

```
<html>
<title>Cambiar el cursor</title>
<head>
<style type="text/css">
<!--
body {cursor : url("find.cur")}
-->
</style>
</head>
<body>
</body>
</html>
```

Ver ejemplo de cambiar cursor con CSS en marcha

*Artículo por **Federico Elgarte***

Caja CSS para meter contenido

Vamos a hacer un artículo práctico de CSS para mostrar una manera de hacer una caja con CSS para meter contenidos. La caja tendrá un diseño especial que enmarque los contenidos que metamos dentro.

Lo mejor para entender lo que pretendemos hacer es ver el ejemplo en marcha.

El ejemplo que hemos escogido se podría hacer de varias maneras, cada una con sus ventajas e inconvenientes. No obstante, nosotros vamos a explicar la forma de hacerlo que nos ha parecido más útil. Tendría estas ventajas e inconvenientes:

Ventajas: La caja puede crecer hacia abajo todo lo que se desee. Por lo que no será problema que hubiera que colocar más o menos texto, porque la caja se ajustará al tamaño que tengamos.

Inconvenientes: La caja tiene un ancho fijo, que está marcado por el tamaño de las imágenes que hayamos realizado. Si anidamos muchas capas dentro de esta caja, puede dar problemas de maquetación, aunque no hemos experimentado este problema en nuestros ejemplos.

Las imágenes

Para realizar esta caja hemos utilizado un par de imágenes. Una para la parte de arriba del marco y otra para la parte de abajo del marco. Estas imágenes las podemos hacer con cualquier programa de edición de imágenes, ajustando colores y formas al aspecto de nuestra propia página. Las imágenes que hemos utilizado son estas:

El código HTML

El ejercicio lo hemos realizado utilizando dos capas (etiqueta <DIV>), una dentro de otra. Una capa se llama "cajaarriba", que tendrá el estilo de la parte de arriba de la caja, y otra "cajaabajo", donde pondremos los estilos necesarios para el cuerpo de la caja y la parte de abajo. La capa principal es "cajaarriba" y dentro estará la capa "cajaabajo". En "cajaabajo" es donde meteremos el texto a colocar dentro de la caja. la metemos dentro de "cajaarriba" y

El código es el siguiente:

```
<div class="cajaarriba">
  <div class="cajaabajo">
  Lorem ipsum dolor sit amet, consectetuer adipiscing elit
  ....
  </div>
</div>
```

El código CSS

Para definir el aspecto de "cajaarriba" y "cajaabajo" se ha utilizado CSS. Como comentábamos, en "cajaarriba" se definen los aspectos de la parte de arriba de la caja y en "cajaabajo" los de la parte de abajo. Además, como "cajaarriba" contiene a "cajaabajo", todos los estilos que hayamos definido en "cajaarriba", también se heredarán en "cajaabajo".

Este es el código de Hojas de estilo para este ejemplo:

```
.cajaarriba{
  width: 600px;
  background-image: url("arriba.gif");
  background-position: top center;
  background-repeat: no-repeat;
}

.cajaabajo {
  background-image: url("abajo.gif");
  background-position: bottom left;
```

```
  background-repeat: no-repeat;
  padding: 58px 75px 58px 69px;
}
```

El ejercicio es sumamente simple, pero bastante útil. Podemos verlo en marcha en una página aparte.

Hemos publicado en el taller de css otros artículos de interés para mostrar cómo se pueden realizar cajas maquetadas con CSS con otros formatos.

*Artículo por **Miguel Angel Alvarez***

Caja CSS con las esquinas redondeadas

En este taller de CSS continuaremos un ejercicio anterior, en el que realizábamos una caja con CSS para meter contenido. Con el mismo esquema relatado en el artículo anterior, vamos a realizar otro tipo de caja -con esquinas redondeadas-, únicamente cambiando las imágenes utilizadas.

Antes de comenzar la lectura de este artículo habría que leer el artículo precedente.

También es aconsejado ver el ejemplo que vamos a construir.

Caja con esquinas redondeadas

El ejemplo que sigue es para crear un contenedor con esquinas redondeadas. Los redondeados de las esquinas los haremos con imágenes, de manera que se pueda variar el color de la caja con las mismas imágenes.

Para conseguir esto vamos a utilizar las imágenes siguientes.

Estas 2 imágenes son transparentes, menos el redondeado de los lados, que tiene color blanco. Debido a ello, estas cajas con esquinas redondeadas sólo podrá utilizarse sobre fondo blanco. Si queremos hacer una caja para utilizar sobre otro fondo, tenemos que rehacer estas imágenes cambiando el color blanco.

El código HTML sigue siendo el mismo:

```
<div class="cajaarriba">
<div class="cajaabajo">
Lorem ipsum dolor sit amet, consectetuer
...
</div>
</div>
```

Ahora vemos el código CSS. Tiene muy pocas variaciones con respecto al del ejemplo anterior:

```
.cajaarriba{
width: 600px;
background-image: url("arriba.gif");
background-position: top center;
background-repeat: no-repeat;
background-color: #660000;
color: #ffffff;
```

```
}
.cajaabajo {
background-image: url("abajo.gif");
background-position: bottom left;
background-repeat: no-repeat;
padding: 5px 5px 5px 5px;
}
```

Hay que fijarse que hemos definido un color de fondo en la clase cajaarriba. Si queremos que la caja varíe el color, simplemente habría que cambiar ese color de fondo.

El resultado puede verse en una página aparte.

*Artículo por **Miguel Angel Alvarez***

Caja CSS con una línea de borde redondeado

Vamos a realizar un contenedor con CSS, en el que tenemos una línea que hace de borde, todo alrededor de la caja, con las esquinas redondeadas.

Se trata de un ejemplo un poco más sofisticado, que cambiando las imágenes, nos permitirá hacer más variedad de contenedores. Este ejercicio está basado en un artículo precedente que habría que leer antes, llamado caja con CSS para meter contenido.

Antes de empezar, también podemos ver el ejemplo que vamos a realizar.

Para realizar este ejercicio vamos a necesitar tres capas con tres imágenes que vamos a colocar de fondo. Las capas e imágenes serán colocadas arriba, para crear los redondeados superiores, en medio, para crear el borde del medio y la capa de abajo, para crear los redondeados inferiores.

La capa de en medio debe crecer más o menos dependiendo del contenido que hayamos incluido dentro de la caja, a más contenido, la capa se hará mas grande.

Las imágenes que hemos utilizado nosotros son las siguientes:

El código HTML varía un poco con respecto al que habíamos visto en los ejemplos de cajas de artículos anteriores. Como decíamos, ahora participan 3 capas distintas.

```
<div class="caja">
   <div class="cajaarriba">
      <div class="cajaabajo">
         Lorem ipsum dolor sit amet, consectetuer
         ...
      </div>
   </div>
</div>
```

El código CSS para definir la clase de estilo de cada una de las tres capas es el siguiente:

```
.caja {     width: 482px;
   background-image: url("centro.gif");
   background-repeat: repeat-y;
}
```

```
.cajaarriba {
   background-image: url("arriba.gif");
   background-position: top center;
   background-repeat: no-repeat;
}
.cajaabajo {
   background-image: url("abajo.gif");
   background-position: bottom left;
   background-repeat: no-repeat;
   padding: 15px 15px 15px 15px;
}
```

Las tres capas tienen la imagen correspondiente como fondo. Caja es la clase para la capa principal, que tiene el fondo que se debe repetir en un mosaico todo lo que crezca el contenedor.

Se puede ver el ejemplo en marcha en una página aparte.

*Artículo por **Miguel Angel Alvarez***

Estilizando formularios

Una de las preguntas más frecuentes cuando se habla de diseño bajo css es como se pueden estilizar los formularios ya que en muchas ocasiones es la parte de un sitio que no queda de acuerdo al estilo del resto de los elementos. Por esto, en esta, mi primera colaboración para CSS Boulevar decidí que sería buena idea escribir al respecto.

En realidad la estilización de formularios es más simple de lo que parece, sin embargo, se necesitan de un par de trucos para adquirir la apariencia que se quiere la cual aún está limitada por ciertas características que no todos los navegadores actuales soportan.

Para comenzar, lo primero que tenemos que hacer es hacer el diseño de como vamos a querer que se vea nuestro formulario. Desde este punto vamos a tener que tomar en cuenta muchs consideraciones que veremos más adelante y que para este caso no tomé en cuenta a propósito por tratarse de un ejemplo en el que hay que resaltar estas limitaciones.

Aquí podemos ver una imagen del diseño inicial.

El Marcado

El código (X)HTML de este ejemplo no tiene mayor ciencia, unicamente se trata de un formulario con 4 campos, algo típico de un sistema de comentarios, queda algo como lo siguiente:

```
<form name="formulario" id="formulario" method="">
<label for="nombre">Nombre:</label> <input type="text" id="nombre" class="campo" />
<label for="email">E-mail:</label> <input type="text" id="email" class="campo" />
<label for="url">URL:</label> <input type="text" id="url" class="campo" />
<label for="comentario">Comentario:</label> <textarea id="comentario" class="campo"></textarea> <br />
<input type="submit" id="boton_enviar" name="enviar" value="Enviar" /> </form>
```

Lo único que hay que resaltar es la falta de definición del tamaño de los campos de texto, esto lo haremos por medio de CSS asi que no es necesario definirlos por el momento

Estilizando

Aplicar estilos a formularios no es diferente de hacerlo con cualquier otro elemento, para empezar, unicamente vamos a agregar color al fondo de la página para ver como esta nuestro formulario inicialmenteen este primer paso.

```
body { background: #A0CE00; }
```

Como podemos ver nuestro formulario esta desordenado y no se ve nada bien. Lo primero que vamos a hacer es organizaro, para facilitar despues el resto del proceso. Vamos a desplegar las etiquetas como block para que se simulen los cambios de linea además de definir un poco el estilo general del formulario.

```
form { padding: 50px; background: #84AA00; width: 250px; }
label { font-size: 14px; font-family: Arial, Helvetica, sans-serif; color: #FFF; display: block; }
.campo { margin-bottom: 20px; }
```

Asi es como se ve nuestro formulario actualmente, ya está organizado pero se sigue viendo simple y el siguiente paso sera cambiarlo.

La primer parte importante para estilizar las cajas de texto es esconder lo que tenemos por default, esto nos va a permitir mas libertad ya que en realidad el truco nada más se trata de reemplazar las cajas con una imagen. Para que las cajas se escondan, si perder funcionalidad vamos a fundirlas con el color de fondo cambiando tanto los bordes como el fondo.

```
.campo { width: 254px; height: 30px; margin-bottom: 20px; border: 1px Solid #84AA00; background: #84AA00; }
#comentario { width: 294px; height: 193px; }
```

En este tercer ejemplo, nuestro formulario parece no existir, sin embargo si movemos el cursor un poco vemos que sigue estando ahí.

Ahora si llegó la parte divertida, lo siguiente será añadir las imagenes como fondo, un simple background-image bastará para lograr el efecto.

```
.campo { width: 254px; height: 30px; margin-bottom: 20px; border: 1px Solid #84AA00; background: #84AA00;
background-image: url(f1.jpg); background-repeat: no-repeat; padding: 2px; color: #669966; font-family: Verdana,
Arial, Helvetica, sans-serif; font-size: 12px; }
#comentario { width: 254px; height: 193px; background-image: url(f2.jpg); background-repeat: no-repeat; }
```

Añadimos el atributo padding para que el texto no quede totalmente pegado a la imagen además de cambiar el color, tamaño y fuente del texto como detalles.

Lo único que nos falta es el botón de enviar, el proceso es el mismo, lo único extra en este caso es la indentación del texto en caso de que no se quiera que este se vea.

```
#boton_enviar { width: 88px; height: 27px; margin-left: 80px; background: #84AA00; border: 1px Solid #84AA00;
background-image: url(boton.jpg); text-indent: -9999px; }
```

Extras

Basicamente el formulario esta listo, vamos a agregar un pequeño efecto para que se vea aún mejor. Hice un par de imagenes extra sin sombreado que se utilizan cuando el usuario pasa el cursor sobre la caja de texto.

```
.campo:hover { background-image: url(f3.jpg); }
```

```
#comentario:hover { background-image: url(f4.jpg); }
```

Si están utilizando un buen navegador, el efecto se ve sin problema alguno pero en Internet Explorer la situación es diferente ya que como sabemos, no reconoce el uso de hover en elementos que no sean enlaces, sin embargo, no hay ningún problema pues se sigue viendo la imagen inicial.

Aquí está nuestro ejemplo finalizado, pueden combinarlo con otros efectos como la decoración del drop-down menu para lograr formularios que se vean muy bien.

Como mencione al principio, nos encontramos aún con muchos problemas al estilizar formularios, en este ejemplo, podemos ver que las barras de scroll en el area de texto pueden llegar a verse mal y desafortunadamente estas no son estilizables más que en IE. Sin embargo, con un poco de imaginación y un par de sencillos hacks podemos lograr efectos para que los formularios vayan de acuerdo con el estilo del sitio.

*Artículo por **Oscar Alcalá***

Maquetar una galería de fotos con CSS

Hoy en día cada vez es más común contar con una galería fotográfica en nuestra página web. A lo largo de este tutorial vamos a ver dos propuestas para hacer una galería de fotos vistosa y trivial de usar para nuestros visitantes, maquetada íntegramente con código estándar HTML y CSS.

Maquetar la galería utilizando capas

En esta primera versión de nuestra galería fotográfica vamos a emplear capas contenedoras donde insertar cada una de nuestras miniaturas, junto con la información que queramos asociarles (título, descripción, etc.).

Para hacernos una idea más clara del trabajo que vamos a realizar lo mejor va a ser que veamos el ejemplo en funcionamiento.

Como viene siendo habitual trabajando con capas usaremos otra capa contenedora para nuestra galería fotográfica con la intención de organizar de forma lógica nuestro diseño y englobar nuestro contenido en un bloque que podemos manipular con estilos. Podemos limitar el ancho de esta capa contenedora y situarla a nuestro gusto. En este ejemplo no restringiremos ninguna propiedad salvo los márgenes para conseguir que nuestra maquetación se adapte a cualquier resolución. Los estilos para esta capa contenedora principal son:

```
#principal {
  margin:0 auto;
}
```

Ahora que ya tenemos una capa donde insertar las fotos prepararemos el estilo para cada una de las miniaturas de las fotos. Haremos flotar cada una de las miniaturas con un float:left; para que se alineen horizontalmente ajustándose al ancho de la ventana. Dándoles margen las separamos unas de otras. Gracias a que estas capas que contienen a las miniaturas de las fotos flotan, se organizarán según el ancho de la ventana, adaptándose a la resolución del

usuario.

Una primera aproximación al estilo para las miniaturas es:

```
.contenedorfoto {
  float:left;
  margin: 10px;
  padding:5px;
}
```

El código html de nuestra galería tiene el siguiente aspecto:

```
<body>
<div id="principal">
  <h3>GALERÍA CSS</h3>
  <div class="contenedorfoto"><a href="#"><img src="img/1.jpg" border="0" alt=""/></a><br
/><span>Descripción de la imagen</span></div>
  <div class="contenedorfoto"><a href="#"><img src="img/2.jpg" border="0" alt=""/></a><br
/><span>Descripción de la imagen</span></div>
  <div class="contenedorfoto"><a href="#"><img src="img/3.jpg" border="0" alt=""/></a><br
/><span>Descripción de la imagen</span></div>
  <div class="contenedorfoto"><a href="#"><img src="img/4.jpg" border="0" alt=""/></a><br
/><span>Descripción de la imagen</span></div>
  <div class="contenedorfoto"><a href="#"><img src="img/5.jpg" border="0" alt=""/></a><br
/><span>Descripción de la imagen extrañamente larga</span></div>
</div>
</body>
```

Como puede observarse hemos añadido una descripción adicional con un . Podeis ver el aspecto de esta primera aproximación aquí.

Antes de mejorar la apariencia del texto trabajaremos sobre la capa contenedora de la foto para limitar su anchura y altura y así lograr la apariencia de mosaico típica de las galerías de fotos. Centraremos el contenido dentro de esta capa, le añadimos estilo al color de fondo y dos bordes (en nuestro caso inferior y derecho) para dar a cada miniatura apariencia de profundidad:

```
.contenedorfoto {
  float:left;
  width:210px;
  height:180px;
  margin: 10px;
  padding:5px;
  background-color:#f5f7f9;
  border-right: #a5a7aa solid 1px;
  border-bottom: #a5a7aa solid 1px;
  text-align:center;
}
```

Ahora ya tenemos nuestra galería con una apariencia más apropiada, como podeis ver en el ejemplo terminado aquí. Para este ejemplo hemos usado para mostrar las miniaturas unas dimensiones proporcionales a la resolución típica.

Existe otra forma de maquetar una galería de fotos utilizando CSS y aplicando listas. Utiliza la que más te guste.

*Artículo por **Javier Chaure***

Maquetar una galería de fotos con CSS usando listas

Una galería de fotos al fin y al cabo no es más que una lista de elementos (en nuestro caso imágenes) que mostraremos como más nos convenga. Desde el punto de vista de la ordenación lógica de la información en nuestras páginas resulta razonable usar una lista para estructurar nuestros elementos, así que a lo largo de este tutorial veremos el potencial de las listas para maquetar eficientemente casos como este de nuestra galería.

Podemos ver el resultado final de este ejemplo, para hacernos una idea exacta de lo que pretendemos conseguir.

> **Referencia:**este ejemplo continúa otro en el que se explica cómo hacer una galería de fotos utilizando capas contenedoras

Normalmente estamos acostumbrados a ver las listas organizadas verticalmente, pero CSS nos permite jugar con los elementos de una lista para mostrarlos de distintas maneras. En nuestro caso, probablemente si organizáramos la lista verticalmente la apariencia de nuestra galería quedaría extraña comparada con una organización horizontal, tal y como vimos en el ejemplo de la galería fotográfica con capas. Por tanto, organizaremos nuestra lista horizontalmente, aplicando un display:inline y flotando los elementos a la izquierda para que fluyan unos a continuación a otros.

Partimos del siguiente código html:

```
<body>:
<div id="principal">
   <h3>GALERÍA CSS - Listas</h3>
  <ul>
     <li><a href="#"><img src="img/1.jpg" border="0" alt=""/><br /><span>Descripci&oacute;n de la
imagen</span></a></li>
     <li><a href="#"><img src="img/2.jpg" border="0" alt=""/><br /><span>Descripci&oacute;n de la
imagen</span></a></li>
     <li><a href="#"><img src="img/3.jpg" border="0" alt=""/><br /><span>Descripci&oacute;n de la
imagen</span></a></li>
     <li><a href="#"><img src="img/4.jpg" border="0" alt=""/><br /><span>Descripci&oacute;n de la imagen más
larga de lo habitual</span></a></li>
     <li><a href="#"><img src="img/5.jpg" border="0" alt=""/><br /><span>Descripci&oacute;n de la
imagen</span></a></li>
      <li><a href="#"><img src="img/6.jpg" border="0" alt=""/><br /><span>Descripci&oacute;n de la
imagen</span></a></li>
     <li><a href="#"><img src="img/7.jpg" border="0" alt=""/><br /><span>Descripci&oacute;n de la
imagen</span></a></li>
      <li><a href="#"><img src="img/8.jpg" border="0" alt=""/><br /><span>Descripci&oacute;n de la
imagen</span></a></li>
      <li><a href="#"><img src="img/9.jpg" border="0" alt=""/><br /><span>Descripci&oacute;n de la
imagen</span></a></li>
      <li><a href="#"><img src="img/10.jpg" border="0" alt=""/><br /><span>Descripci&oacute;n de la
imagen</span></a></li>
      <li><a href="#"><img src="img/11.jpg" border="0" alt=""/><br /><span>Descripci&oacute;n de la
imagen</span></a></li>

  </ul>
</div>
</body>
```

Como puede apreciarse, hemos insertado dentro de cada <li> (list item) nuestra imagen y un

comentario añadido con un , tal y como hicimos en el tutorial de la galería fotográfica con capas.

Nuestra redefinición de estilos para las listas quedaría de la siguiente manera:

```
#principal li {
  display:inline;
  float:left;
}
```

Nótese que las típicas viñetas que aparecen asociadas a las listas desordenadas desaparecen cuando aplicamos el display:inline, de la misma forma que si hubiéramos unsado un list-style:none. Podeis ver el resultado de aplicar estos estilos aquí . El comportamiento de nuestros elementos de la lista (los list ítems <li>) es similar al de las capas ahora.

Podemos modificar las propiedades que queramos para lograr que la apariencia de cada uno de los elementos de la lista se ajuste a nuestro objetivo. Basándonos en los estilos que aplicamos en el ejemplo de la galería fotográfica con capas, cambiamos el color de fondo, el ancho de cada <li>, les damos margin y padding para presentarlos correctamente y damos estilo a los bordes para lograr apariencia de profundidad.

De este modo la definición de estilos para nuestra lista de fotos queda así:

```
#principal li {
  display:inline;
  float:left;
  width:220px;
  background-color:#f5f7f9;
  padding:5px;
  margin:10px;
  text-align: center;
  border-right: #a5a7aa solid 1px;
  border-bottom: #a5a7aa solid 1px;
}
```

Y como podemos ver en el resultado final aquí, queda perfectamente maquetada, adaptable a distintas resoluciones y con un código limpio y fácil de entender.

*Artículo por **Javier Chaure***

Realizar un rollover sólo con CSS y utilizando imágenes

Se va a mostrar un pequeño truco para crear un efecto de rollover con imágenes, pero usando un simple CSS, sin necesidad de ninguna otra tecnología o lenguaje de programación. El rollover (iluminación o cambio de color al pasar el ratón por encima) es un efecto muy sencillo, pero puede dar ideas para dar mayor dinamismo a nuestro sitio, sin que esto afecte a la complejidad del código.

Además, de paso, vamos a responder una pregunta habitual: ¿cómo hacer para que los enlaces ocupen todo el espacio de la celda o capa donde están situados? O dicho de otra forma: ¿cómo hacer para que el enlace se muestre con todo el espacio disponible dentro del contenedor en el que está?

Lo que deseamos es que el lugar donde colocamos el enlace se comporte como el propio enlace, así, al pasar el ratón por el contenedor donde se haya situado el link (no necesariamente sobre el texto del enlace), se realice el efecto de iluminación.

Lo mejor que podemos hacer es ver el ejemplo en marcha. Merece la pena prestar atención a que el enlace cambia de color al pasar el ratón por encima, pero no hace falta situarse sobre el texto del enlace para que cambie, sino que sirve con posicionarlo sobre la capa donde se ha colocado el enlace.

El código HTML

Para realizar este efecto se ha creado un código HTML como el siguiente:

```
<div class=navegador>
<p class=navenlace><a href="#">Enlace 1</a></p>
<p class=navenlace><a href="#">Link chulo</a></p>
<p class=navenlace><a href="#">Otras cosas</a></p>
</div>
```

Tenemos una capa con estilo "class=navegador" y dentro de esta, tres párrafos con estilo "class=navenlace". La clase navegador contendrá estilos generales para toda la barra de navegación y los párrafos, con clase navenlace, contendrán los estilos de cada línea del navegador.

Hemos visto que el HTML es bastante sencillo. Ahora sólo tenemos que aplicar unos estilos bonitos para realizar el efecto buscado.

Estilos CSS

Empecemos por la capa general, con clase "navegador", donde vamos a utilizar una declaración de estilos como esta:

```
.navegador{
  margin-left: 39%;
  margin-right: 42%;
}
```

Simplemente le estamos indicando unos márgenes, para que se sitúe donde nosotros queremos, que es más o menos en el centro de la ventana del navegador.

Ahora veamos la declaración del estilo de la clase navenlace, que es el estilo que damos a cada línea de enlace de la barra de navegación.

```
.navenlace {
  font-weight: bold;
  border: 1px solid #999999;
  margin:3 0 3 0px;
  padding: 2px;
}
```

Simplemente hemos indicado negrita, un borde de 1 pixel, así como un margen y un padding.

Ahora, para cada enlace hemos definido un par de estilos. Uno para los enlaces en general y otro para los enlaces "hover", que son los que tienen el ratón encima de ellos. De este modo,

los enlaces siempre se verán de un mismo modo y cuando se ponga el ratón encima de uno de ellos se activará el estilo definido en hover.

```
.navenlace a {
  background-image: url("nav-fondo2.gif");
  color: #666666;
  text-decoration: none;
  display: block;
  width: 100%;
}

.navenlace a:hover {
  background-image: url("nav-fondo.gif");
  color: #ffffcc;
}
```

En el primer caso (.navenlace a) se indica el estilo para los enlaces en general. Simplemente hemos definido una imagen de fondo "nav-fondo2.gif" para los enlaces, un color para el texto del link, que no aparezcan subrayados y, lo más importante, que se comporten como un bloque, que se indica con el atributo display: block;

Al comportarse los enlaces como un bloque se produce el efecto que estábamos buscando: que no hace falta posicionarse sobre el texto del enlace para que cambie el estilo, simplemente lo hará con posicionarse en el contenedor donde están situados. Con ello también conseguimos que el estilo de los enlaces se transfiere a todo el contenedor. Además, también hemos definido el ancho de los enlaces a 100%, para asegurarnos que ocupen todo el espacio del contenedor donde están situados.

En el segundo caso (.navenlace a:hover) simplemente definimos un estilo distinto, utilizando una imagen de fondo y un color distinto para los enlaces. La imagen llamada "nav-fondo.gif" se mostrará como fondo del espacio donde está colocado el enlace. Sólo se modificará el estilo del enlace sobre el que hayamos puesto el ratón y no el de los otros enlaces de la barra de navegación.

Con esto ya está. Es así de simple. Está probado en Internet Explorer y en Mozilla Firefox.

Ahora bien, quería comentar que también se puede dar un ancho definido a la capa donde están todos los enlaces, , utilizando el atributo width. Esto da problemas en Internet Explorer a la hora de que los enlaces se comporten como un bloque, con lo que sólo se activan si ponemos el ratón en el texto del link. La solución para Internet Explorer, es justamente asignar el atributo width: 100% a los enlaces, en la clase "navenlace". Este atributo width: 100% para Firefox le resulta indiferente, con lo que el problema queda resuelto para los navegadores más comunes.

Para finalizar, podemos ver el ejemplo en una página aparte.

*Artículo por **Miguel Angel Alvarez***

Esquinas redondeadas con CSS y Javascript, sin imágenes

Hacer cajas con las esquinas redondeadas es una de las típicas preguntas que se pueden hacer sobre CSS. Existen varias soluciones al problema, que hemos visto en algunos artículos de

DesarrolloWeb.com, pero en estas cajas con CSS utilizábamos imágenes, que es lo más directo.

Ahora vamos a ver un método para crear cajas con las esquinas redondeadas que no utiliza imágenes, sino simplemente CSS y un poco de Javascript. El desarrollo de esta técnica no corre por nuestra cuenta. En este caso simplemente vamos a comentar un desarrollo de Alessandro Fulciniti que se llama Nifty Corners Cube que ya está en su tercera versión y se puede ver en la URL: http://www.html.it/articoli/niftycube

Como hemos dicho, el sistema crea cajas o bloques de contenido con esquinas redondeadas o suavizadas, pero sin usar imágenes. En lugar de imágenes utiliza CSS y Javascript compatible con la mayoría de los navegadores. Aunque según se informa en la página de producto, los navegadores Internet Explorer 5, así como los que tengan deshabilitado Javascript, verán cajas cuadradas en lugar de ver las esquinas redondeadas. En la primera versión ya se presentaba como una interesante opción, pero en la segunda y sobretodo en la tercera versión se ha visto mejorado el script y simplificado su manejo, dentro de lo posible.

Lo mejor de todo es que se presenta con licencia GPL, luego cualquiera puede usarlo en sus desarrollos libremente.

Dispone de tres elementos para ponerlo en marcha:

1. Un archivo Javascript
2. Un archivo CSS
3. Llamadas Javascript dentro de las páginas que quieran mostrar cajas con esquinas redondeadas.

En realidad el archivo CSS para hacer esquinas redondeadas lo incluye internamente Javascript, por lo que nosotros simplemente tendremos que incluir un archivo externo .js, que es el código Javascript con las funciones que sirven para hacer las esquinas redondeadas.

```
<script type="text/javascript" src="niftycube.js"></script>
```

Luego será necesario hacer unas llamadas a Javascript para redondear las esquinas de las capas que deseemos. De esta manera:

```
<script type="text/javascript">
window.onload=function(){
Nifty("div#box","big");
}
</script>
```

Como se puede ver, se ha definido una función que se ejecutará con el evento onload (cuando termine de cargarse la página). Esa función invoca a otra función llamada Nifty() que está definida en el Javascript que habíamos incluido como archivo aparte. La función Nifty() recibe dos parámetros. El primero es el selector CSS de la capa que se desea redondear y el segundo sirve para indicar opciones específicas del redondeo.

El primer parámetro, que decíamos es el selector CSS, tiene mucha versatilidad. Permite especificar el redondeo de elementos de la página variados, como todas las apariciones de una etiqueta concreta, una clase de CSS, una etiqueta con un identificador determinado, etc. En la documentación del producto se pueden ver todos los tipos de selectores CSS, pero algunos son estos:

Selector de etiqueta: "p" o bien "h2"
Se mostrarán con los bordes redondeados todas las apariciones de una etiqueta, como los párrafos o los encabezamientos h2.

Selector por identificador "div#capax" o bien "p#parrafoy"
Esto sirve como selector de las siguientes etiquetas con sus identificadores:

Div con su identificador
P con su identificador

Selector de clase "div.nave" o bien "span.fuentepequena".
Cada vez quw apliquemos esa clase, se pondrá con las esquinas redondeadas.

Selector descendente "div#cabecera h1"
Hace referencia a la etiqueta h1 dentro de la capa con id="cabecera".

Hay otros selectores que se pueden ver en la documentación. Además en el primer parámetro se pueden especificar varios selectores a la vez, separados por una coma:

Nifty("div#box,div#prueba,p","big");
Esto afectará a las capas box, prueba y a todas las etiquetas de párrafos.

El segundo parámetro son las opciones de redondeo que se aplicarán a los selectores en cada función. Las distintas opciones se deben escribir separadas por comas. Existe una lista de opciones bastante grande, pero comentamos algunas que parecen más útiles:

tr: redondear sólo la esquina superior derecha.
tl: redondear sólo la esquina superior izquierda.
br: redondear sólo la esquina inferior derecha.
bl: redondear sólo la esquina inferior izquierda.
top: esquinas de arriba
bottom: esquinas de abajo
left: esquinas de la izquierda
right: esquinas de la derecha
all: todas las esquinas (es la opción por defecto)
none: ninguna esquina se redondea

Con estas opciones juntas se pueden definir redondeos de varias esquinas, pero no todas, para que quede alguna sin redondear:

Nifty("p","tl,bl,br");

small: se utilizan esquinas pequeñas, de 2px
normal: se utilizan esquinas normales 5px (opción por defecto)
big: se utilizan esquinas grandes de 10px

Con estas opciones definimos tamaños de las esquinas.

Luego hay otras opciones que son un poco menos claras, que dejamos para que cada uno se las estudie si las llegase a necesitar: transparent, fixed-height, same-height. Parece útil la opción same-height, que hace que todas las capas tengan la misma altura, por si queremos hacer un diseño en distintas columnas donde cada columna tiene la misma altura.

Ejemplo muy básico de esquinas redondeadas sin imágenes

Hemos hecho unas pruebas del sistema y además hemos tratado de conseguir el código más básico con el que probar este script. El resultado es el siguiente ejemplo, en el que se redondean las esquinas de todos los párrafos de la página. Hemos colocado dos párrafos con dos colores distintos de fondo.

```
<html>
<head>
<title>Ejemplo esquinas redondeadas basico</title>
<script type="text/javascript" src="niftycube.js"></script>
<script type="text/javascript">
window.onload=function(){
Nifty("p");
}
</script>
</head>
<body>
<p style="background:#ccccff;padding:5px;">
Esto es una prueba
<br>mola cantidad!!!
</p>
<br>
<p style="background:#cccccc;padding:5px;">
De nuevo, Hola mi amigo!!!
</p>
</body>
</html>
```

Este ejemplo se puede ver en una página aparte.

Haciendo pruebas hemos comprobado que si ponemos el color de fondo de un párrafo con el nombre de un color en lugar de su código RGB, no funciona.

```
<p style="background: red;">
```

Si cambiamos el color de fondo de la página, las esquinas redondeadas también se ven perfectamente.

```
<body style="background: #66ff00;">
```

Pero si ponemos el color de fondo de la página con un nombre en lugar de RGB, tampoco funciona.

Podemos ver multitud de ejemplos y descargar los códigos necesarios para hacer esquinas redondeadas y los ejemplos, en la web del propio sistema:

http://www.html.it/articoli/niftycube/index.html

*Artículo por **Miguel Angel Alvarez***

CSS para campos textarea de formulario

Con las hojas de estilo CSS se puede configurar el aspecto de cada elemento de una página web, de una manera muy detallada. En este taller de CSS vamos a aplicar estilos a los elementos de formulario Textarea, que son cajas de texto de varias líneas. Veremos varios estilos aplicados sobre textarea, comentando sus distintas propiedades CSS.

El objetivo de este taller no es explicar el modo de trabajo con CSS, sino más bien practicar con determinados atributos sobre los textarea. Se puede encontrar información básica para aprender a manejar las hojas de estilo en nuestro Manual de CSS. También disponemos de otros Talleres de CSS, donde aprender a utilizar esta tecnología por la práctica.

Antes que nada, podemos ver la página donde están aplicados los distintos estilos que vamos a comentar sobre elementos textarea.

Un textarea es un elemento de formulario, luego en principio deberíamos colocarlos entre <form> y </form>. El código HTML más básico para un textarea es el siguiente:

```
<textarea class=estilotextarea></textarea>
```

Ya le hemos aplicado una clase de estilos CSS (estilotextarea), que definiremos en la declaración de estilos de la página:

```
.estilotextarea {width:400px;height:100px;border: 2px solid #990000;}
```

En esta declaración de estilos hemos indicado que el ancho de la caja de texto sea de 400 píxeles y que el alto sea de 100 píxeles. También hemos indicado un borde de 2 píxeles de tamaño y de color rojo oscuro.

El efecto es el siguiente:

Ahora veamos otro código HTML para incluir un textarea.

```
<textarea class=estilotextarea2 cols="60" rows="8"></textarea>
```

En este caso, aparte que la clase para definir el estilo del textarea ha cambiado (estilotextarea2), también se está indicando unas filas y unas columnas como tamaño del textarea, con los atributos cols y rows. Como los textarea son líneas de texto de varias líneas, con cols se indica el número de caracteres en horizontal del textarea y con rows el número de filas o líneas.

Ahora el estilo para este textarea sería el siguiente:

```
.estilotextarea2 {width:300px;height:80px;border: 1px dotted #000099;}
```

Nos fijamos que con CSS hemos redefinido el ancho y alto, con los atributos width y height. Ahora bien, entre la definición con HTML de la altura y anchura en caracteres del textarea, y la definición con CSS de la altura y anchura en píxeles, manda lo que hemos definido con CSS. Esto es así generalmente en todos los casos de estilos que se redefinen con CSS, siempre acaban siendo las hojas de estilo las que predominan en el aspecto de los elementos de las webs.

Además, hemos declarado un borde con línea de puntos de 1 pixel de anchura en el textarea, de color rojo oscuro. El aspecto final de este textarea será el siguiente:

Ahora veamos un tercer ejemplo de textarea. Primero su código HTML:

```
<textarea class=estilotextarea3 cols="30" rows="8">Texto de prueba</textarea>
```

Este textarea tiene de particularidad que aparecerá con un texto escrito dentro. Es decir, cuando se visualice en la página web, en lugar de estar vacío, tendrá escrito lo que hemos colocado entre <textarea> y </textarea>, "Texto de prueba".

La clase que define el estilo de este textarea se puede ver a continuación:

```
.estilotextarea3 {font-family: Garamond,verdana;font-size: 18pt;font-weight: bold;letter-spacing: 5px;}
```

Como se puede ver, se han definido en esta ocasión varios estilos para las tipografías que se van a utilizar en el texto del textarea. En esta ocasión se ha definido una fuente garamond, y en su defecto, verdana. Un tamaño del texto de 18 puntos, negrita, y un espaciado entre letras de 5 píxeles.

El resultado se puede ver a continuación.

Texto de prueba

Para acabar, veremos un último textarea al que vamos a poner el fondo transparente y en el que vamos a modificar los colores de las barras de desplazamiento del textarea.

```
<textarea class=estilotextarea4 cols="30" rows="5"></textarea>
```

Para definir el estilo hemos utilizado el siguiente CSS:

```
.estilotextarea4 {background-color: transparent;border: 1px solid #000000;scrollbar-arrow-color: #000066;scrollbar-base-color: #000033;scrollbar-dark-shadow-color: #336699;scrollbar-track-color: #666633;scrollbar-face-color: #cc9933;scrollbar-shadow-color: #DDDDDD;scrollbar-highlight-color: #CCCCCC;}
```

El color de fondo es transparente, por el atributo background-color: transparent; esto quiere decir, que el color del textarea tiene el mismo color que el fondo de la página donde está colocado. Si el textarea lo tenemos colocado sobre un fondo blanco, no observaremos ninguna diferencia con respecto a otros textareas, pero si lo tenemos sobre fondo de otro color, el textarea se verá de ese mismo color. Luego, se ha aplicado un borde de un píxel negro y con los restantes atributos se modifica el color de las barras de desplazamiento del textarea. Atención, que los estilos para barras de desplazamiento sólo funcionan en Internet Explorer.

El resultado se puede ver a continuación:

Con estos ejemplos hemos podido ver unas interesantes declaraciones de estilos para elementos textarea de formulario. Esperemos que os sirvan para hacer vuestros propios formularios con más estilo. Por cierto, que en el taller de CSS tenemos ejercicios para aplicar estilos a otros elementos de formulario, como los artículos Decorar un campo select de formulario con CSS, Estilizando formularios o Estilos en campos de texto.

*Artículo por **Miguel Angel Alvarez***

Creación de gráficas de barras con CSS para la maquetación

Con CSS se puede maquetar una página web, desde la propia estructura de la página hasta elementos más específicos como una gráfica de barras. Como en el caso de este artículo, que utilizando posicionamiento CSS vamos a definir la colocación y el tamaño de elementos que nos pueden ayudar a construir una gráfica de barras. Utilizaremos capas para realizar las gráficas, una para el fondo de la gráfica y dentro de esta, otras capas para cada una de las barras.

Es decir, no vamos a utilizar ni imágenes ni tablas, sino simplemente etiquetas a las que vamos a aplicarles posicionamiento y estilos para crear unas gráficas bastante vistosas.

Para no llevar a confusión, hay que aclarar que en este ejemplo no vamos a crear código especial para hacer un sistema de generación de gráficas de barras para páginas web. En lugar de ello, el objetivo es simplemente explicar el modo en el que podríamos maquetar una gráfica de barras con CSS y capas.

El ejemplo que vamos a construir se puede ver en una página aparte.

Grafica de barras CSS 1

Veamos una primera gráfica, que expresaría las visitas en un día de la semana a un supuesto sitio web. El código HTML sería este:

```
<div id=contenedor1 class=gris>
  <div id=titulo1>Visitas por día</div>
  <div id=grafica1 class=degradadoverde>
    <div id=lunes class=verde>Lunes 390</div>
    <div id=martes class=verde>Martes 423</div>
    <div id=miercoles class=verde>Miércoles 412</div>
    <div id=jueves class=verde>Jueves 459</div>
    <div id=viernes class=verde>Viernes 405</div>
    <div id=sabado class=verde>Sábado 320</div>
    <div id=domingo class=verde>Domingo 302</div>
  </div>
</div>
```

Como se puede observar, tenemos un , o capa con un contenedor, donde se va a situar a su vez otro con un título y otro para la gráfica de las barras. Tenemos dentro de la gráfica otras tantas capas, una para cada día de la semana.

El código CSS será el encargado de aplicar formato a estas capas para que se presenten como una gráfica de barras. Hemos combinado tanto los estilos con clases como los de identificadores. En las clases colocamos los estilos que son comunes y que podríamos repetir en varias capas y en los identificadores especificamos estilos propios exclusivos para la capa. Veamos primero los estilos de los identificadores.

El contenedor simplemente tiene definida posición relativa, una anchura y un espacio de margen con padding.

```
#contenedor1{
  position: relative;
  padding:5px;
  width: 524px;
}
```

Dentro del contenedor tenemos el título, que simplemente define un tipo de letra y un margen.

```
#titulo1{
  font: bold 10pt Verdana, Tahoma, Arial;
  margin:2 0 5 0px;
}
```

Así mismo tenemos una capa con la gráfica. Esta capa es la que tiene anidadas a su vez otras 7 capas con las barras. En los estilos de la gráfica tenemos una anchura, un espaciado arriba y abajo y un formato de texto.

```
#grafica1 {
```

```
width: 500px;
  padding:10 0 10 0px;
  font: bold 8pt Verdana, Tahoma, Arial;
}
```

Ahora veremos cada una de las capas con los días de la semana. Lo importante es ver que cada capa tiene un ancho, que debe de ser proporcional al valor que se desea mostrar en la barra. En este caso hacemos corresponder la anchura de la capa con las visitas que se han obtenido ese día de la semana. El alto es siempre el mismo y el margen arriba y abajo también.

```
#lunes{
  width: 390px;
  height: 18px;
  margin: 5 0 5 0px;
}
#martes{
  width: 423px;
  height: 18px;
  margin: 5 0 5 0px;
}
#miercoles{
  width: 412px;
  height: 18px;
  margin: 5 0 5 0px;
}
#jueves{
  width: 459px;
  height: 18px;
  margin: 5 0 5 0px;
}
#viernes{
  width: 405px;
  height: 18px;
  margin: 5 0 5 0px;
}
#sabado{
  width: 320px;
  height: 18px;
  margin: 5 0 5 0px;
}
#domingo{
  width: 302px;
  height: 18px;
  margin: 5 0 5 0px;
}
```

Ahora veamos los estilos que hemos colocado por medio de clases. Como decía, utilizamos las clases cuando tenemos estilos que podrían ser utilizados en otras capas. En este caso el color de fondo y el borde de las barras es el único estilo que es común a varias capas, porque lo tenemos igual en todas las barras. Sin embargo, el fondo gris del contenedor y el verde de la gráfica también lo sacamos a clases porque en la práctica se podría utilizar para aplicar estilos a otras gráficas.

Los estilos simplemente definen un color de fondo y unos bordes oscuros por la parte de abajo y la derecha y más claros por arriba y la izquierda, así se consigue un efecto de viselado.

```
.gris{
  background-color: #b5b5b5;
  border-bottom: 2px solid #6b6b6b;
  border-right: 2px solid #6b6b6b;
```

```
  border-top: 2px solid #f0f0f0;
  border-left: 2px solid #f0f0f0;
}
.morado{
  background-color: #a05aab;
  border-bottom: 2px solid #672770;
  border-right: 2px solid #672770;
  border-top: 2px solid #d090d9;
  border-left: 2px solid #d090d9;
}
.degradadoverde{
  background-color: #e1e455;
  background-image: url('images/degradado-verde.jpg');
  background-repeat: repeat-x;
  border-bottom: 2px solid #6f722d;
  border-right: 2px solid #6f722d;
  border-top: 2px solid #ece996;
  border-left: 2px solid #ece996;
}
```

Como se ha podido ver, en la clase degradadoverde se ha definido además una imagen de fondo con un degradado, para mejorar un poco el diseño de la gráfica haciéndolo menos plano. Se utiliza el atributo background-repeat: repeat-x; para que el fondo se repita sólo por la coordenada de la x.

Podemos ver el ejemplo en marcha en una página aparte.

Esto es todo. Hemos dejado para un artículo posterior otra gráfica un poco más elaborada, con más colores y más detalles como una leyenda. Seguiremos entonces trabajando la maquetación de gráficas de barras con posicionamiento CSS.

*Artículo por **Miguel Angel Alvarez***

Grafica de barras con CSS, parte 2

El posicionamiento CSS nos sirve para crear elementos complejos en páginas web, como podría ser una gráfica de barras. En el artículo anterior del taller de CSS aprendimos a realizar una maquetación de una gráfica de barras con CSS.

Vamos a utilizar un HTML en el que tenemos unas simples capas y con CSS definiremos sus estilos y posicionamiento, de modo que obtengamos un diseño para realizar una atractiva gráfica de barras, separando por completo el aspecto del contenido.

Veamos el ejemplo en una página aparte para hacernos una idea.

En este artículo veremos cómo realizar la segunda gráfica del ejemplo, la de abajo, que es un poco más elaborada, al contar varios colores para las barras y una leyenda.

El código HTML para la gráfica

```
<div id=contenedor2 class=gris>
      <div id=titulo2>Buscadores referidos</div>
   <div id=grafica2 class=degradadoverde>
      <div id=google class=azul></div>
```

```
   <div id=direct class=verde></div>
     <div id=msn class=rojo></div>
     <div id=yahoo class=morado></div>
  </div>
  <div id=leyenda>
     <ul>
     <li class=azul>Google</li>
     <li class=verde>Directo</li>
     <li class=rojo>MSN</li>
     <li class=morado>Yahoo</li>
     </ul>
  </div>
</div>
```

Como la gráfica del artículo anterior, tenemos un contenedor que tiene dentro todos los elementos de la gráfica: El título, la gráfica con las barras y la leyenda.

El título es una simple capa con texto. La gráfica es otra capa que contiene otras capas con cada una de las barras. Por último, la leyenda, que está dentro de otra capa, tiene una lista con los distintos elementos que se muestran en la grafica.

Veamos los estilos para cada elemento, divididos entre clases e identificadores. Las clases para los estilos comunes a varios elementos y los identificadores con los estilos propios de cada capa. Veamos ahora los estilos para los identificadores.

El contenedor tiene un posicionamiento, un tamaño y un espaciado.

```
#contenedor2{
  position: relative;
  padding:5px;
  width: 280px;
}
```

El título tiene una tipografía y un margen.

```
#titulo2{
  font: bold 10pt Verdana, Tahoma, Arial;
  margin:2 0 5 0px;
}
```

La gráfica tiene dentro unas dimensiones y un espaciado a la izquierda.

```
#grafica2 {
  width: 250px;
  height: 100px;
  padding-left:20px;
  position: relative;
}
```

Luego, para cada una de las barras tenemos un posicionamiento absoluto, para colocarlas dentro de la gráfica. Como esta capa está dentro de la gráfica, la posición absoluta corresponde a la posición que ocupará la barra dentro de la capa de la gráfica.

```
#google{
  width: 190px;
  height: 30px;
  position:absolute;
  left:0px;
  top:10px;
}
```

```
#direct{
  width: 80px;
  height: 30px;
  position:absolute;
  left:0px;
  top:25px;
}
#msn{
  width: 35px;
  height: 30px;
  position:absolute;
  left:0px;
  top:40px;
}
#yahoo{
  width: 25px;
  height: 30px;
  position:absolute;
  left:0px;
  top:55px;
}
```

Como se puede ver, cada barra tiene una anchura que corresponde con el valor que se quiere representar.

Para la leyenda se han de definir estilos tanto para la capa de la leyenda como para la lista que contiene.

```
#leyenda{
  position: relative;
  text-align:center;
}
#leyenda ul{
  margin:10 0 10 0px;
  padding: 0px;
}
#leyenda li{
  display:inline;
  font: bold 8pt Verdana, Tahoma, Arial;
  height: 10pt;
  margin: 2px;
  padding: 2px;
}
```

Lo más destacable de la lista es que tiene el display inline, que hará que se muestren todos los elementos en una línea, en lugar de ocupar un elemento por línea de texto.

Terminamos viendo las clases, que contienen los colores de fondo y borde de las capas. Como cada barra tiene su color, cada una tiene una clase distinta, que utilizamos también como estilo en la leyenda. Además, tanto el contenedor como la gráfica en si tienen un estilo personalizado, que ya utilizamos y comentamos en el ejercicio anterior, con una gráfica de barras más sencilla.

```
.azul{
  background-color: #9190a8;
  border-bottom: 2px solid #535270;
  border-right: 2px solid #535270;
  border-top: 2px solid #cbcbda;
  border-left: 2px solid #cbcbda;
}

.rojo{
```

```
  background-color: #c47965;
  border-bottom: 2px solid #67382c;
  border-right: 2px solid #67382c;
  border-top: 2px solid #e1a494;
  border-left: 2px solid #e1a494;
}
.verde{
  background-color: #5aae4c;
  border-bottom: 2px solid #357d2a;
  border-right: 2px solid #357d2a;
  border-top: 2px solid #aceaa2;
  border-left: 2px solid #aceaa2;
}
.gris{
  background-color: #b5b5b5;
  border-bottom: 2px solid #6b6b6b;
  border-right: 2px solid #6b6b6b;
  border-top: 2px solid #f0f0f0;
  border-left: 2px solid #f0f0f0;
}
.morado{
  background-color: #a05aab;
  border-bottom: 2px solid #672770;
  border-right: 2px solid #672770;
  border-top: 2px solid #d090d9;
  border-left: 2px solid #d090d9;
}
.degradadoverde{
  background-color: #e1e455;
  background-image: url('images/degradado-verde.jpg');
  background-repeat: repeat-x;
  border-bottom: 2px solid #6f722d;
  border-right: 2px solid #6f722d;
  border-top: 2px solid #ece996;
  border-left: 2px solid #ece996;
}
```

Simplemente se debe notar que se utiliza un color de fondo y unos bordes para crear un efecto de viselado. En la clase que da color a la gráfica, llamada degradadoverde, hemos utilizado además una imagen de fondo.

Podemos ver el ejemplo en una página aparte.

Con esto hemos visto como crear gráficas de barras solamente con estilos CSS y unas cuantas capas. Las posibilidades son mucho mayores si se desea emplear un poco de tiempo y creatividad para mejorar los ejemplos presentes y hacerlos más variados.

*Artículo por **Miguel Angel Alvarez***

CSS para imprimir páginas web

A veces necesitamos que nuestra página se imprima en una impresora de manera distinta a como se visualiza en la página web. Por ejemplo, si en una página se muestra un informe con datos que se desea guardar impreso en papel, probablemente deseemos que en la impresora se muestre con una fuente más pequeña, para que se pueda comprimir todo el contenido de manera que quepa en un folio. También es posible que en los informes deseemos que aparezca el logo de la compañía centrado en la cabecera del informe.

Todo esto se puede hacer con CSS. Las Hojas de Estilo en Cascada sirven para definir el aspecto de la página, y estos estilos se pueden declarar de manera distinta a la hora de imprimir un documento y a la hora de verlo en el navegador.

Con CSS se puede definir estilos en un documento externo, de esta manera:

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

Con esto suponemos que tenemos un archivo llamado estilos.css, que está en el mismo directorio de la página, donde se definen los estilos del documento.

> **Referencia:** Las diferentes maneras de incluir estilos en una web están comentadas en nuestro Manual de CSS.

De modo parecido, podemos asignar una hoja de estilos externa para definir el aspecto cuando un usuario imprime la página web:

```
<link href="estilos_impresion.css" rel="stylesheet" type="text/css" media="print">
```

Lo único que cambia es el atributo media="print", que indica que esta hoja de estilos es sólo para cuando se va a imprimir la web.

Ejemplo de dos hojas css distintas para impresión y visualización

Ahora veamos un ejemplo de página web que tiene dos hojas de estilo distintas, una para cuando se está en el navegador y otra cuando se va a imprimir.

El ejemplo se puede ver en marcha en una página aparte.

Tenemos un HTML que incluye dos hojas de estilos y dispone varias capas, que luego maquetaremos o posicionaremos con CSS.

```
<html>
<head>
  <title>informe superpuzzles</title>
  <link rel="STYLESHEET" type="text/css" href="estilo.css">
  <link rel="STYLESHEET" type="text/css" href="estilo_imprimir.css" media="print">
</head>

<body>

<div id="contenedor">
   <div id="cabecera">
     Superpuzzles
   </div>
  <div id="logo">
      <img src="logo.gif">
  </div>
  <div id="cuerpo">
   <div id="lateral">
   <ul>
      <li><a href="#">Enlace 1</a>
      <li><a href="#">Vínculo 2</a>
   </ul>
   </div>
   <div id="derecha">
      <div id="principal">
```

```
Contenido de un posible informe
      </div>
    </div>
   </div>
    <div id="pie">
    © 2005 DesarrolloWeb.com
    </div>
</div>

</body>
</html>
```

Como se ha podido ver en el HTML anterior, se han incluido dos archivos CSS con estilos. El primero es estilo.css, que es el estilo que se utilizará al visualizar la página en el navegador. El segundo link con una hoja de estilos CSS es estilo_imprimir.css, que definirá el aspecto de la página al imprimirla (fijarse en el atributo media="print" de la etiqueta).

Los códigos CSS son muy parecidos, simplemente hemos hecho un par de cambios para ilustrar lo que venimos diciendo. En la visualización de la página no se mostrará la capa con id="logo". Por su parte, al imprimir la página no se mostrará la barra de navegación de la izquierda y los contenidos centrales se mostrarán en todo el ancho del espacio de impresión. Tampoco se mostrará la capa id="cabecera".

El código CSS de visualización en navegador

```
BODY {
font: 8pt Verdana, Geneva, Arial, Helvetica, sans-serif;
margin: 10 0 10 0px;
text-align: center;
background-color: #ffffff;
}
#contenedor{
text-align: left;
width: 770px;
margin: auto;
}
#cabecera{
background-color: #d0d0ff;
color: #333300;
font-size:12pt;
font-weight: bold;
padding: 3 3 3 10px;
}

#logo{
visibility:hidden;
display: none;
}

#cuerpo{
margin: 10 0 10 0px;
}
#lateral{
width: 160px;
background-color: #d0d0ff;
float:left;
}
#lateral ul{
margin : 0 0 0 0px;
padding: 0 0 0 0px;
list-style: none;
}
```

```
#lateral li{
background-color: #ffffff;
margin: 2 2 2 2px;
padding: 2 2 2 2px;
font-weight: bold;
}
#lateral a{
color: #3333cc;
text-decoration: none;
}

#pie{
background-color: #cccccc;
padding: 3 10 3 10px;
text-align:right;
clear: both;
}

#principal{
background-color: #ffffff;
padding: 0 0 0 20px;
width: 580px;
float: left;
}

#principal table{
background-color: #ffffff;
width: 580px;
border: 2px solid #cccccc;
font-size:10pt;
}
```

El código CSS que se utilizará para la impresión de la página

```
BODY {
  font: 8pt Verdana, Geneva, Arial, Helvetica, sans-serif;
  margin: 10 0 10 0px;
  text-align: center;
   background-color: #ffffff;
}
#contenedor{
   text-align: left;
   width: 600px;
   margin: auto;
}
#cabecera{
  visibility:hidden;
  display: none;
}

#logo{
  visibility:visible;
  display: block;
  margin-left: 20px;
}

#cuerpo{
   margin: 10 0 10 0px;
}
#lateral{
  visibility:hidden;
  display: none;
}

#pie{
```

```
    background-color: #cccccc;
    padding: 3 10 3 10px;
    text-align:right;
   clear: both;
}

#principal{
    background-color: #ffffff;
    padding: 0 0 0 20px;
    width: 600px;
    float: left;
}

#principal table{
   background-color: #ffffff;
    width: 600px;
   border: 2px solid #cccccc;
   font-size:10pt;
}
```

Nuevamente, ponemos el enlace para que se pueda ver la página del ejemplo de estilos de impresión CSS.

> **Nota:** Si queremos ver cómo se imprimiría la página, pero sin necesidad de utilizar la impresora (para no gastar papel ni tinta o si no tenemos impresora), podemos acceder al menú Archivo - Vista preliminar de nuestro navegador.

Esperamos que este taller de css para imprimir haya sido de utilidad. Queremos daros un enlace a otro taller en el que se explica cómo evitar, con CSS, que una página se imprima.

*Artículo por **Ana Alvarez***

Generar Columnas con CSS de una lista sin tablas

Es un truco sencillo, pero ingenioso para hacer que las listas se muestren en varias columnas. Nosotros hemos puesto tres columnas, pero se podrían haber creado las que se hubieran deseado.

Se trata simplemente de crear una lista y definir con estilos CSS la anchura de sus elementos. Definiremos la anchura de los elementos <LI> como un 30% del total de anchura de la lista.

Los estilos CSS que colocaremos son los siguientes. Primero veremos el estilo para lo que es la lista <UL> y luego los de los elementos de la lista <LI>:

```
UL.col3
{
PADDING-RIGHT: 0px;
PADDING-LEFT: 0px;
FLOAT: left;
PADDING-BOTTOM: 0px;
MARGIN: 15px 0px;
WIDTH: 100%;
PADDING-TOP: 0px;
LIST-STYLE-TYPE: none
}

UL.col3 LI
{
PADDING-RIGHT: 2px;
```

```
DISPLAY: inline;
PADDING-LEFT: 2px;
FLOAT: left;
PADDING-BOTTOM: 2px;
WIDTH: 30%;
PADDING-TOP: 2px
}
```

Para crear varias columnas en CSS se define en el estilo UL.col3 LI, donde el width es el tamaño de cada columna. Así automáticamente se irán creando las columnas. Como hemos puesto un 30%, el espacio donde esté la lista se dividirá en tres columnas, sin necesidad de utilizar tablas.

El html quedaría así para la lista:

```
<ul class="col3">
<li>Lo que sea</li>
<li>Segunda opcion</li>
<li>Tercera cosa</li>
<li>Otra cosa, que aparecerá en la primera columna</li>
<li>Otra cosa para la segunda columna</li>
<li>Y esto para la tercera columna</li>
</ul>
```

*Artículo por **Pol Salvat***

Marcos para fotos con CSS

En este taller de CSS vamos a crear una serie de 4 marcos para fotos, que podemos utilizar en una página web para mejorar la presencia, pero sin complicarnos la vida. Una vez elegido el marco que más nos guste, podemos utilizarlo repetidas veces en la página para que todas las fotos se vean de manera parecida y gane también un poco de personalidad el diseño de la web.

Para hacer esta serie de marcos hemos evitado el uso de imágenes adicionales, sólo las fotografías, lo que hace que el diseño sea más fácil de aplicar, sólo definiendo los estilos CSS.

Podemos ver el resultado conseguido en una página aparte.

Primer marco CSS, que es sencillo y aplica estilos tanto al contenedor de la foto como a la imagen misma.

```
.marco1 {
  padding:8px;
  background-color: #f5f5f5;
  width: 200px;
  border: 1px solid #999999;
}
.marco1 IMG{
  border: 1px solid #000000;
}

<div class="marco1">
```

```
<img src="im1.jpg" width="200" height="150" alt="">
</div>
```

En este caso hemos definido un espacio entre el marco y la foto, un color de fondo y un borde. Con la segunda declaración estamos definiendo también un borde de color negro para la imagen, para que quede más resaltada.

Veamos ahora el segundo marco, que hemos querido hacer un borde como con relieve.

```
.marco2 {
  padding:8px;
  background-color: #f5f5f5;
  width: 200px;
  border-bottom: 1px solid #999999;
  border-right: 1px solid #999999;
}

<div class="marco2">
<img src="im2.jpg" width="200" height="150" alt="">
</div>
```

Hemos definido estilos CSS para un espacio entre la foto y el borde del contenedor, un color de fondo y los mencionados bordes, que sólo se aplican abajo y a la derecha.

Ahora hemos definido un marco muy sencillo, pero que recuerda a las fotos instantáneas de Polaroid.

```
.marco3 {
  padding:8px 8px 20px 8px;
  background-color: #ffffff;
  width: 200px;
  border: 1px solid #999999;
}

<div class="marco3">
<img src="im3.jpg" width="200" height="150" alt="">
</div>
```

Simplemente hemos definido unos espacios entre la foto y el borde del elemento, donde el espacio de abajo es mayor para emular el marco de las Polaroid, que era más ancho abajo.

Luego hemos puesto un borde al propio contenedor para que se distinga con el fondo de la página, que también es blanco.

Por último hemos creado un marco con sombra. Este marco con sombra sin utilizar imágenes se nos complica bastante en el código HTML, pero todavía más en el código CSS. Este efecto de sombra con CSS ya fue explicado en otro artículo anterior en DesarrolloWeb.com, por lo que no voy a dar más explicaciones, sino la referencia al artículo: Efecto de sombra con CSS

```
.blur{
  background-color: #ccc; /*shadow color*/
  color: inherit;
  margin-left: 4px;
  margin-top: 4px;
  width: 224px;
}
.shadow, .content{
  position: relative;
  bottom: 2px;
```

```
  right: 2px;
}
.shadow{
  background-color: #666; /*shadow color*/
  color: inherit;
}
.content{
  background-color: #fff; /*background color of content*/
  color: #000; /*text color of content*/
  border: 1px solid #000; /*border color*/
  padding: .5em 2ex;
}
.content IMG{
  border: 1px solid #000000;
}

<div class="blur">
<div class="shadow">
<div class="content"><img src="im4.jpg" width="200" height="150" alt=""></div>
</div>
</div>
```

Como vemos en el código HTML, se utilizan tres contenedores distintos, para emular el efecto de degradado de la sombra, que es más oscura cerca del objeto y más clara a medida que la sombra se aleja del objeto.

Tendremos que definir estilos para cada uno de los tres contenedores. Lo bueno es que, como los estilos se definen por clases CSS, sólo los tendremos que definir una vez y los podremos utilizar en todas las imágenes que se deseen.

Podemos ver de nuevo los distintos marcos realizados con CSS en una página aparte.

*Artículo por **Miguel Angel Alvarez***

Crear una barra de navegación lateral con listas y CSS

Con CSS podemos hacer mucho con muy poco esfuerzo y muy poco código HTML. Con una simple lista y un par de etiquetas `<div>` podemos crear una barra de navegación muy configurable a nuestro aspecto preferido.

En este artículo veremos como un mismo código HTML, con distintas declaraciones de estilos CSS, permite obtener barras de navegación de modelos variados. Todos los modelos son de barras de navegación verticales, que vienen muy bien para los laterales de las páginas. Aunque si lo deseamos, también hemos creado en otro artículo unos ejemplo con barra de navegación vertical.

Partimos, como decimos, de un HTML bien simple:

```
<div id="nav">
  <div class="titulonav">
  Secciones
  </div>

  <div class="cuerporec">
  <ul>
```

```
    <li><a href="#">Portada</a></li>
    <li><a href="#">Grandes rutas</a></li>
    <li><a href="#">Nuevas rutas</a></li>
    <li><a href="#">Ciudades</a></li>
    <li><a href="#">Pueblos</a></li>
  </ul>
  </div>
</div>
```

Tenemos varias capas a las que hemos asignado distintas clases e identificadores: una capa principal llamada "nav", donde meteremos toda la barra de navegación. Otra llamada "titulonav" con el título de la barra. Por último otra "cuerporec" con el cuerpo de las distintas opciones de la barra de navegación.

Ahora veremos distintas definiciones de estilos para adaptar a este mismo HTML. Cada una con más dificultad que la anterior.

Primer estilo CSS

Veamos un primer estilo CSS que es muy sencillo.

Podemos verlo en una página aparte.

En este ejemplo cada enlace viene con un borde. Con unos fondos en colores planos bien simples, pero medianamente vistosos.

```
body{
  font-family: verdana, helvetica;
  font-weight: bold;
}

#nav{
  width: 155px;
}

.titulonav{
  border: 1px solid #d7d7d7;
  font-size: 8pt;
  font-weight: bold;
  background-color: #e0e0e0;
  color: #000;
  padding: 4px;
}

.cuerporec{
  margin-top: 0px;
  margin-right: 0pt;
  margin-bottom: 10px;
  margin-left: 0pt;
}

.cuerporec ul{
  margin-top: 0px;
  margin-right: 0px;
  margin-bottom: 0px;
  margin-left: 0px;
  padding-top: 0pt;
  padding-right: 0pt;
  padding-bottom: 0pt;
  padding-left: 0px;
  list-style-type: none;
```

```
  list-style-image: none;
  list-style-position: outside;
}

.cuerporec li{
  border-bottom: 1px solid #d7d7d7;
  border-left: 1px solid #d7d7d7;
  border-right: 1px solid #d7d7d7;
  padding: 1px 0 2px 4px;
  font-size: 8pt;
  margin-top: 0px;
  margin-right: 0pt;
  margin-bottom: 0px;
  margin-left: 0pt;
  background-color: #ffffcc;
}

.cuerporec a{
  text-decoration: none;
}
```

Segundo ejemplo de estilos CSS

El segundo ejemplo hace uso de la lista para mostrar un estilo más habitual, de elementos de lista, pero de manera personalizada.

Podemos verlo en una página aparte.

```
body{
  font-family: verdana, helvetica;
  font-weight: bold;
}

#nav{
  width: 155px;
}

.titulonav{
  border-right-width: 1px;
  border-right-style: solid;
  border-right-color: #d7d7d7;
  border-bottom-width: 1px;
  border-bottom-style: solid;
  border-bottom-color: #d7d7d7;
  padding-top: 5px;
  padding-right: 0pt;
  padding-bottom: 5px;
  padding-left: 4px;
  font-size: 8pt;
  font-weight: bold;
  background-color: #e8e8e8;
}

.cuerporec{
  border-width: 1px;
  border-style: solid;
  border-color: #d7d7d7;
  margin-top: 4px;
  margin-right: 0pt;
  margin-bottom: 10px;
  margin-left: 0pt;
}
```

```
.cuerporec ul{
  margin-top: 5px;
  margin-right: 10px;
  margin-bottom: 20px;
  margin-left: 0px;
  padding-top: 0pt;
  padding-right: 0pt;
  padding-bottom: 0pt;
  padding-left: 1px;
  list-style-type: none;
  list-style-image: none;
  list-style-position: outside;
}

.cuerporec li{
  padding-left: 12px;
  font-size: 8pt;
  padding-bottom: 2px;
  margin-top: 1px;
  margin-right: 0pt;
  margin-bottom: 1px;
  margin-left: 0pt;
  background: transparent url("cuadrado-amarillo.gif") 0 2px no-repeat;
}

.cuerporec a{
  text-decoration: none;
}
```

Tercer ejemplo de estilos CSS para el navegador

Un tercer ejemplo que no reviste mucha mayor dificultad que el anterior, pero que altera sensiblemente el aspecto visual.

En este caso vamos a separar un poco cada una de las cajitas donde vienen los enlaces y vamos a poner un fondo con un atractivo degradado en cada elemento de la lista.

Podemos verlo en funcionamiento en una página.

```
body{
  font-family: verdana, helvetica;
  font-weight: bold;
}

#nav{
  width: 155px;
}

.titulonav{
  border-width: 1px;
  border-style: solid;
  border-color: #000000;
  font-size: 8pt;
  font-weight: bold;
  background-color: #999999;
  color: #ffffff;
  padding: 4px;
}

.cuerporec{
  margin-top: 4px;
  margin-right: 0pt;
  margin-bottom: 10px;
```

```
  margin-left: 0pt;
}

.cuerporec ul{
  margin-top: 5px;
  margin-right: 0px;
  margin-bottom: 20px;
  margin-left: 0px;
  padding-top: 0pt;
  padding-right: 0pt;
  padding-bottom: 0pt;
  padding-left: 1px;
  list-style-type: none;
  list-style-image: none;
  list-style-position: outside;
}

.cuerporec li{
  border-width: 1px;
  border-style: solid;
  border-color: #d7d7d7;
  padding: 1px 0 2px 2px;
  font-size: 8pt;
  padding-bottom: 2px;
  margin-top: 2px;
  margin-right: 0pt;
  margin-bottom: 2px;
  margin-left: 0pt;
  background-image: url(fondo_gris.gif);
}

.cuerporec a{
  text-decoration: none;
}
```

Espero que estos distintos ejemplos de estilos CSS para hacer una barra de navegación hayan servido de inspiración a los lectores.

*Artículo por **Artemio Artigas***

Efecto zoom de imagen utilizando solo Css

La información de estilo puede ser adjuntada tanto como un documento separado o en el mismo documento HTML.

En este css podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo "style".

a X

Para ver el resultado de este sólo tienes que colocar el puntero justo debajo de la letra M en la etiqueta ZOOM que hay encima de la imagen:

```
<head></p>
<style type="text/css">
#info {position:relative;}
#magnifier {display:inline; position:relative; width:240px; height:210px; z-index:100; float:left; margin:0 20px 10px 0;
background:url(graphics/enlarge.gif) no-repeat;}
```

```
#cover1 {position:absolute; top:0; left:0; width:24px; height:30px; background:transparent;}
#cover2 {position:absolute; top:0; left:48px; width:240px; height:30px; background:transparent;}

#magnifier img {position:absolute; width:240px; height:180px; top:30px; left:-240px;}

#magnifier ul {padding:0; border:0; margin:0; list-style-type:none;}


#magnifier li {display:block; width:24px; height:24px; position:absolute; left:24px; top:0; background:transparent;}

#magnifier li.one {left:0;}

#magnifier li:hover {width:24px; height:24px; left:0;}
#magnifier li.ten:hover {width:24px; height:24px; left:24px;}

#magnifier li.one:hover img {width:280px; height:210px; left:-240px;}
#magnifier li.two:hover img {width:320px; height:240px; left:-216px;}
#magnifier li.three:hover img {width:360px; height:270px; left:-192px;}
#magnifier li.four:hover img {width:400px; height:300px; left:-168px;}
#magnifier li.five:hover img {width:440px; height:330px; left:-144px;}
#magnifier li.six:hover img {width:480px; height:360px; left:-120px;}
#magnifier li.seven:hover img {width:520px; height:390px; left:-96px;}
#magnifier li.eight:hover img {width:560px; height:420px; left:-72px;}
#magnifier li.nine:hover img {width:600px; height:450px; left:-48px;}
#magnifier li.ten:hover img {width:640px; height:480px; left:-48px;}
#magnifier li.eleven:hover img {width:240px; height:180px; left:-24px;}

#magnifier table {border:0; padding:0; margin:0; border-collapse:collapse;}
</style><p><!--[if lte IE 6]></p>
<style type="text/css">

#magnifier li {position:static;}
#magnifier img {left:-264px;}
#magnifier li {height:20px;}
#magnifier a {display:block; width:24px; height:24px; left:24px; top:0; position:absolute; background:transparent;}

#magnifier a:hover {border:0;width:24px; height:24px; left:0;}

#magnifier a.aten:hover {border:0;width:24px; height:20px; left:24px;}

#magnifier a.aone:hover img {width:280px; height:210px; left:-240px;}
#magnifier a.atwo:hover img {width:320px; height:240px; left:-216px;}
#magnifier a.athree:hover img {width:360px; height:270px; left:-192px;}
#magnifier a.afour:hover img {width:400px; height:300px; left:-168px;}
#magnifier a.afive:hover img {width:440px; height:330px; left:-144px;}
#magnifier a.asix:hover img {width:480px; height:360px; left:-120px;}
#magnifier a.aseven:hover img {width:520px; height:390px; left:-96px;}
#magnifier a.aeight:hover img {width:560px; height:420px; left:-72px;}
#magnifier a.anine:hover img {width:600px; height:450px; left:-48px;}
#magnifier a.aten:hover img {width:640px; height:480px; left:-48px;}
#magnifier a.aeleven:hover img {width:240px; height:180px; left:-24px;}

</style><p><![endif]--></p>
<p></head></p>
<p><body></p>
<div id="wrapper">
<div id="info">
<div id="magnifier"><H5>Sitúa X aqui tu puntero para zoom</H5></p>
<ul>
<li class="one"><a class="aone" href="#nogo"><!--[if IE 7]><!--></a><!--<![endif]--><br />
<table>
<tr>
<td>
<ul>
```

```
<li class="two"><a class="atwo" href="#nogo"><!--[if IE 7]><!--></a><!--<![endif]--><br />
<table>
<tr>
<td>
<ul>
<li class="three"><a class="athree" href="#nogo"><!--[if IE 7]><!--></a><!--<![endif]--><br />
<table>
<tr>
<td>
<ul>
<li class="four"><a class="afour" href="#nogo"><!--[if IE 7]><!--></a><!--<![endif]--><br />
<table>
<tr>
<td>
<ul>
<li class="five"><a class="afive" href="#nogo"><!--[if IE 7]><!--></a><!--<![endif]--><br />
<table>
<tr>
<td>
<ul>
<li class="six"><a class="asix" href="#nogo"><!--[if IE 7]><!--></a><!--<![endif]--><br />
<table>
<tr>
<td>
<ul>
<li class="seven"><a class="aseven" href="#nogo"><!--[if IE 7]><!--></a><!--<![endif]--><br />
<table>
<tr>
<td>
<ul>
<li class="eight"><a class="aeight" href="#nogo"><!--[if IE 7]><!--></a><!--<![endif]--><br />
<table>
<tr>
<td>
<ul>
<li class="nine"><a class="anine" href="#nogo9"><!--[if IE 7]><!--></a><!--<![endif]--><br />
<table>
<tr>
<td>
<ul>
<li class="ten"><a class="aten" href="#nogo"><!--[if IE 7]><!--></a><!--<![endif]--><br />
<table>
<tr>
<td>
<ul>
<li class="eleven"><a class="aeleven" href="#nogo"><!--[if IE 7]><!--></a><!--<![endif]--><br />
<table>
<tr>
<td>
<p>                        <img src="http://www.tufuncion.com/images/tufuncionblanco.jpg" alt="Sparrow" title="Sparrow" /></p
</td>
</tr>
</table>
<p><!--[if lte IE 6]></a><![endif]--></li>
</ul>
</td>
</tr>
</table>
<p><!--[if lte IE 6]></a><![endif]--></li>
</ul>
</td>
</tr>
</table>
<p><!--[if lte IE 6]></a><![endif]--></li>
</ul>
```

```
</td>
</tr>
</table>
<p><!--[if lte IE 6]></a><![endif]--></li>
</ul>
</td>
</tr>
</table>
<p><!--[if lte IE 6]></a><![endif]--></li>
</ul>
</td>
</tr>
</table>
<p><!--[if lte IE 6]></a><![endif]--></li>
</ul>
</td>
</tr>
</table>
<p><!--[if lte IE 6]></a><![endif]--></li>
</ul>
</td>
</tr>
</table>
<p><!--[if lte IE 6]></a><![endif]--></li>
</ul>
</td>
</tr>
</table>
<p><!--[if lte IE 6]></a><![endif]--></li>
</ul>
</td>
</tr>
</table>
<p><!--[if lte IE 6]></a><![endif]--></li>
</ul>
<div id="cover1"></div>
<div id="cover2"></div>
</div>
<p>
```

*Artículo por **Manu Gutierrez***

Maquetar una página con un mapa de Google, usando CSS

La maquetación con mapas de Google, que creamos con el API de Google Maps, es muy sencilla, ya que el mapa ocupa el espacio disponible en la capa donde esté contenido. Así pues, al incorporar un mapa a una página web, no tenemos que especificar el tamaño del mapa por Javascript ni nada parecido, simplemente el mapa de Google tomará el los atributos de ancho y alto de la capa (el <div>) donde está contenido. Esto es dinámico, es decir, si cambia el espacio disponible de la capa donde está el mapa -ya sea porque el usuario ha cambiado el tamaño de la ventana o porque se han alterando los atributos width y height de la capa con Javascript o por cualquier modo- cambiará automáticamente el tamaño del mapa para ajustarse dinámicamente al espacio disponible.

Dicho esto, puede resultar poco revelador este artículo, porque no vamos a ver nada de Javascript ni de creación de mapas de Google, sino que vamos a mostrar como maquetar con CSS, teniendo en cuenta que en una de las capas hay un mapa de Google. No obstante, seguro que algunas de las informaciones proporcionadas aquí no van a ser triviales para los lectores.

En DesarrolloWeb.com ofrecemos un manual en el que puedes aprender a trabajar con el API de Google Maps.

También recomiendo la lectura del material que ofrecemos en DesarrolloWeb.com sobre la maquetación CSS, pues voy a dar por sabidos los aspectos sobre maquetación con hojas de estilos y por tanto, no los voy a explicar.

Los atributos de altura y anchura de un mapa son la capa que lo contiene

El ejemplo más sencillo que podemos encontrarnos para maquetar un mapa de google es que tenga dimensiones en ancho y alto fijas. Es como hemos colocado los mapas en el manual hasta este momento:

```
<div id="map" style="width: 765px; height: 388px"></div>
```

En este ejemplo el mapa ocupa 765 píxel de ancho y 388 de alto. Esto no tiene ningún secreto.

¿Pero qué pasa si queremos que un mapa de Google Maps tenga el ancho de toda la ventana del navegador? Porque sabemos que el navegador puede tener distintos tamaños, dependiendo de la definición de pantalla del visitante y si la ventana está maximizada o dimensionada de cualquier otra forma.

Entonces es muy sencillo, podemos colocar width:100% en la declaración de estilos css de la capa donde está el mapa.

```
<div id="map" style="width: 100%; height: 388px"></div>
```

Este mapa ocuparía todo el área disponible de la ventana del navegador, o bien de la capa donde estuviera contenido.

El ejemplo se complica si queremos que el mapa de Google ocupe además todo el alto disponible en el navegador, que también, sabemos, puede ser variable dependiendo de las características de pantalla o del estado de la ventana del browser. Lo normal es que probásemos algo como esto:

```
<div id="map" style="height: 100%; width:100%;"></div>
```

Pero esto tal cual, sin hacer ninguna otra cosa, tiene un problema y es que misteriosamente el mapa aparece vacío o con height = 0. Esto es porque los navegadores tienen problemas al maquetar con height=100%. La idea para solucionarlo es colocar a todas las capas que contengan al mapa height:100%, así como a las etiquetas <BODY> y <HTML> que también deberían tener el height de 100%. Entonces nos debería funcionar y el mapa ocuparía todo el alto y ancho de la ventana.

Referencia: Podemos ver una FAQ con explicaciones sobre utilización del height:100% en CSS.

Ahora veamos el ejemplo de un mapa de Google Maps que ocupa todo el ancho y alto del espacio en una página aparte.

Maquetar un mapa de Google con width y height 100%, pero mezclado con otros elementos

Para acabar veamos un ejemplo de maquetación de un Google Maps que ocupa el 100% del espacio, pero que tiene una cabecera y una barra lateral. La cabecera y el lateral ocupan unos espacios fijos y el mapa de google ocupará todo el sitio que dejan libre otros elementos de la maquetación.

Para explicarlo lo más sencillo es ver directamente el ejemplo en una página aparte.

El ejercicio es simple, pero necesitaremos conocimientos de CSS sobre maquetación y posicionamiento. El código es el siguiente:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:v="urn:schemas-microsoft-com:vml">
<head>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html;charset=ISO-8859-1">
<title>Mapa de Google</title>
<script src="http://maps.google.com/maps?file=api&v=2&key=coloca-tu-llave"
type="text/javascript"></script>
<script type="text/javascript">
//<![CDATA[
function load() {
  if (GBrowserIsCompatible()) {
    var map = new GMap2(document.getElementById("map"));
    map.setCenter(new GLatLng(33,0),3);
    map.addControl(new GLargeMapControl());
    map.setMapType(G_NORMAL_MAP);
  }
}

window.onload=load
//]]>
</script>
<style type="text/css">
html,body{
margin:0px;
height:100%;
}
</style>
</head>

<body>
<div id="contenedor" style="width:100%; height:100%;">
  <div id="cabecera" style="background-color: #ffcc99; font-weight:bold; font-size: 110%; height: 23px; padding:
3px; margin-bottom:10px;">Esto es la cabecera!!</div>
  <div id="lateral" style="float:right; width:200px; height:100%; background-color:#eeff99;">
    <div id="contenidolateral" style="padding: 30px 10px 0 10px;">
    Por <a href="http://www.guiarte.com">guiarte.com</a>
    <br />
    <br />
    Esto son contenidos que colocamos en el lateral izquierdo. Resulta muy fácil maquetar con un mapa de Google en
una capa que toma el tamaño del contendor donde se encuentra.
    </div>
  </div>
  <div id="map" style="height: 100%; margin-right:210px;"></div>
```

```
</div>
</body>
</html>
```

*Artículo por **Miguel Angel Alvarez***

Hacks CSS

Los Hacks de CSS son triquiñuelas para conseguir que una misma declaración de estilos CSS actúe de manera distinta en distintos navegadores. Bajo este concepto se engloban diversas técnicas, muchas veces poco ortodoxas, para lidiar con las distintas interpretaciones de CSS que tienen los distintos navegadores.

Los hacks CSS surgen porque distintos navegadores a menudo interpretan de manera distinta el mismo estilo CSS. Entonces los desarrolladores y diseñadores, que se ven en la necesidad de hacer páginas que se muestren igual independientemente del navegador, tienen que utilizar códigos y trucos complejos para conseguir que la página se visualice de la misma forma en distintos navegadores.

La cuestión es que debería haber una sola interpretación del estándar CSS, pero como cada navegador hace la guerra por su cuenta, al final ocurre que no es así y nos queda a nosotros el problema de hacer que todo cuadre. Es algo a lo que ya estamos acostumbrados con HTML y Javascript, pero que también tenemos que aprender a lidiar con CSS.

Los Hack CSS muchas veces juegan con distintas interpretaciones a errores de sintaxis, cometidos a propósito en el código CSS. Por ello en principio conviene evitarlos o bien es recomendable utilizarlos con cuidado.

Hack CSS de los comentarios de dos barras (//)

En este artículo mostraremos un ejemplo sobre cómo funcionan los Hack CSS, con un típico caso que utiliza los comentarios formados por dos barras (//), que no son tratados como comentarios en Internet Explorer.

Es decir, como hemos visto en otros lenguajes de programación, si una línea de código tiene dos barras al comienzo significa que está comentada y no se tiene en cuenta. En CSS los comentarios se tienen que hacer así:

```
/* esto es un comentario */
```

Pero Firefox también interpreta los comentarios formados por //

```
// esto es un comentario
```

Internet Explorer simplemente ignora las dos barras y trata la línea como si fuera un código normal.

Por ejemplo tenemos este código:

```
h1{
  font-size: 12px;
```

```
  color: #009900;
  //font-size: 16px;
  //color: #000099;
}
```

Aquí hemos definido un par de estilos para la etiqueta H1.
Los dos primeros estilos son font-size:12px y color:#009900. Todos los navegadores leerán estos estilos.

Pero luego tiene otros dos estilos, que en realidad son los mismos que los anteriores, que comienzan por dos barras. Firefox y Opera, que interpretan esas dos barras como comentarios, no leerán los estilos. Internet Explorer que no tiene en cuenta las dos barras como comentarios, leerá los dos últimos estilos.

Por lo tanto, el resultado final será que

- Todos los navegadores mostrarán los H1 con tamaño de fuente 12px y color verde.
- Internet Explorer mostrará los H1 con tamaño de 16px y color azul, dado que no tiene en cuenta las dos barras como si fueran comentarios.

Podemos ver el ejemplo en marcha en la URL:
http://www.desarrolloweb.com/articulos/ejemplos/tallercss/css-hack1.html

Tendríamos que verlo con Internet Explorer y con otros navegadores para comprobar que cambian los estilos con los que se muestra el encabezamiento H1.

Conclusión sobre los Hacks CSS

Lo cierto es que los Hacks CSS están ahí y muchas personas los utilizan habitualmente. Hay varios tipos de Hacks, con técnicas más o menos conflictivas u ortodoxas. No debemos abusar de ellos, pero en algunas ocasiones no queda otra solución que utilizarlos para que una página se vea de la misma manera en los navegadores más comunes.

Vimos también un tipo de Hack CSS que sirve para ocultar ciertos estilos en Firefox u otros navegadores y que sólo funcione en Internet Explorer.

*Artículo por **Miguel Angel Alvarez***

Maquetación de un recuadro con pestañas con CSS

Vamos a hacer con CSS una maquetación de un recuadro que tiene una serie de pestañas o solapas, que habremos visto en numerosas ocasiones en interfaces de usuario diversas y en páginas web. Estos recuadros permiten hacer clic en cada una de las solapas o pestañas para mostrar contenidos en un área. Pulsando una pestaña distinta se muestran otros contenidos en el mismo espacio donde estaban los anteriores, por lo que esta interfaz de usuario nos puede servir para mostrar distintos contenidos, bajo demanda de usuario, en un mismo espacio.

Este artículo sólo va a especificar cómo se podría maquetar con HTML y CSS este sistema de etiquetas, sin hacer los scripts necesarios para ponerlo en marcha. En otros artículos explicaremos cómo hacer que esta interfaz responda a las acciones del usuario, para que pulsando cada pestaña se cambien los contenidos del recuadro para mostrar los relacionados

con esta solapa. Como la creación de esta maqueta ya reviste una relativa complejidad por si sola, la vamos a explicar en este taller de CSS.

Para entender bien de qué estamos hablando, lo mejor es ver uno de los ejemplos que hemos creado.

Código HTML de la interfaz

Hemos creado un sencillo código HTML que sea suficientemente versátil para soportar varios tipos de diseño, simplemente alterando la hoja de estilos en cascada CSS.

```
<div class="interfazpestanas">
  <div class="pestanas">
    <ul>
    <li class="pestanaseleccionada">Opción 1</li>
    <li>Opción 2</li>
    <li>Opción 3</li>
    </ul>
    <div class="cuerpopestanas">
      Esto es el cuerpo de las pestañas?
    </div>
  </div>
</div>
```

Como se puede ver, ha quedado un HTML bastante sencillo. Está claro que se podría haber complicado algo más para poder alcanzar un grado mayor de detalle, pero con lo que hemos hecho obtenemos un resultado bastante versátil y potente de cara a darle un aspecto atractivo con CSS.

Como se puede ver, tenemos una capa de class "pestanas" donde vamos a colocar el conjunto de solapas o pestañas y el cuerpo donde se mostrará toda la información de cada pestaña.

Dentro hemos colocado una lista UL donde cada uno de sus elementos LI serán cada una de las pestañas del conjunto. Además, vemos que la pestaña que está seleccionada en un momento dado tiene una clase class="pestanaseleccionada", para poder asignarle un estilo distinto.

Por último, para definir el lugar donde se muestra el cuerpo de cada pestaña hemos utilizado otra capa con class="cuerpopestanas".

Bueno… resulta bastante sencillo y esquemático. Ahora veremos la parte más interesante, que es donde damos estilos CSS al diseño HTML.

Estilos CSS para definir el aspecto de la interfaz de pestañas

Veamos las hojas de estilos utilizadas para definir el aspecto de la interfaz de pestañas.

```
.pestanas{
  width: 350px;
}
.pestanas ul{
  margin:0px;
  padding:0px;
  list-style: none;
}
.pestanas li{
  float:left;
```

```
  margin:0px 6px 0px 0px;
  padding:4px 6px 4px 6px;
  background-color: #999999;
  color: #eeeeee;
  font-size:8pt;
}
li.pestanaseleccionada{
  background-color: #333366;
  color: #ffffff;
  font-weight: bold;
}
.cuerpopestanas{
  background-color: #333366;
  color: ffffff;
  clear:both;
  margin:0px;
  padding:4px;
  height: 300px;
  overflow: auto;
}
```

Como vemos, a la interfaz le hemos dado un espacio de 300 pixel.

La lista la hemos definido para que no tenga márgenes ni bullets (esos puntitos que se colocan antes de cada elemento) de ningún tipo. Luego, los elementos de la lista tienen ya un margen y un padding y, lo más importante, es que tienen el float:left; para que se pongan uno detrás de otro en lugar de uno debajo de otro. Como habíamos marcado en HTML, la pestaña seleccionada tiene un estilo particular que hemos definido con la clase li.pestanaseleccionada, que simplemente tiene un cambio de color de fondo y del texto.

Para acabar, tenemos la clase cuerpopestanas, que es donde definimos la altura de la interfaz. Tenemos también un clear:both; para que el cuerpo no tenga en cuenta el float de los elementos de la lista donde hemos colocado las pestañas. Pero lo más importante es el overflow: auto para que salgan unas barras de desplazamiento verticales, en caso que el contenido del texto, o lo que haya dentro, ocupe más que el alto del cuerpo de las pestañas.

Con esta declaración de estilos hemos conseguido este efecto, que podemos ver en la URL: http://www.desarrolloweb.com/articulos/ejemplos/tallercss/pestanas/pestanas.html

Pero hemos creado otras declaraciones de estilos CSS para hacer este mismo esquema pero con otros aspectos:

Ejemplo 2:
Ver ejemplo
Con esta hoja de estilos

Ejemplo 3:
Ver ejemplo
Con esta hoja de estilos

*Artículo por **Miguel Angel Alvarez***

Crear una barra de navegación horizontal con listas y CSS

Ya hemos estudiado la aplicación de estilos CSS a listas en diversos artículos de DesarrolloWeb.com, en el taller de CSS y en los distintos manuales que tenemos sobre la materia. En concreto, hay un artículo que vamos a completar con este otro taller de CSS, en el que vimos cómo Crear una barra de navegación vertical con listas y CSS. En este caso veremos como crear una barra de navegación, pero en horizontal. Además veremos como alterar los estilos cuando se pasa el ratón por encima de los enlaces.

Básicamente, para crear una barra de navegación horizontal por listas, tenemos que conseguir que los elementos de la lista se pongan uno detrás de otro en la misma línea (el comportamiento normal de las listas es que aparezcan los elementos uno debajo del otro en distintas líneas). Además, cada elemento será un enlace y al pasar el ratón sobre ellos, debe resaltarse de alguna manera el elemento, lo que aportará el dinamismo a la barra de navegación. Todo esto se puede hacer con CSS. Veremos cómo a lo largo de este artículo.

Sería ideal ver el resultado final del ejemplo en marcha antes de continuar.

Código HTML de la lista - Barra de navegación

El código HTML de la lista para hacer la barra de navegación es sencillo. La complejidad la reservaremos para las declaraciones de estilos.

```
<div id="navegador">
<ul>
<li><a href="#">Elemento 1</a></li>
<li><a href="#">Elemento 2</a></li>
<li><a href="#">Elemento 3</a></li>
<li><a href="#">Elemento 4</a></li>
</ul>
</div>
```

Como se puede comprobar, tenemos una capa llamada "navegador", que tiene el código de la lista, que en principio no necesita nada especial. Simplemente se colocarán los elementos LI deseados y un enlace dentro de cada uno.

Código CSS de la lista

Primero vamos a definir el estilo de la lista (etiqueta UL):

```
#navegador ul{
  list-style-type: none;
  text-align: center;
}
```

Simplemente indicamos, con list-style-type: none;, que no se van a utilizar bolitas o marcas de ningún tipo al lado de los elementos de la lista.

Luego, para centrarla le asignamos text-align: center;

Ahora los estilos para cada uno de los elementos de la lista (etiqueta LI):

```
#navegador li{
  display: inline;
  text-align: center;
```

```
  margin: 0 10px 0 0;
}
```

Esto indica varias cosas. Primero le damos un display: inline; para conseguir que los elementos se dispongan uno detrás de otro en la misma línea. Luego con text-align: center; volvemos a indicar que cada uno de los elementos tiene el texto centrado y por último con margin indicamos un margen entre los elementos.

Los estilos para los enlaces serían como estos:

```
#navegador li a {
  padding: 2px 7px 2px 7px;
  color: #666;
  background-color: #eeeeee;
  border: 1px solid #ccc;
  text-decoration: none;
}
```

Primero hacemos un padding: 2px 7px 2px 7px; para que el enlace tenga un poco de espacio a los lados. Esto nos vendrá también bien para que el área que se puede pinchar del enlace sea mayor. Luego marcamos el color del enlace con color: #666;, un fondo background-color: #eeeeee;, el borde border: 1px solid #ccc; y por último le quitamos el subrayado a los enlaces con text-decoration: none;.

Para acabar, utilizamos la pseudoclase hover para definir un estilo distinto cuando el ratón pasa por encima de un enlace.

```
#navegador li a:hover{
  background-color: #333333;
  color: #ffffff;
}
```

Eso es todo! acabamos de terminar nuestra barra de navegación horizontal con listas y CSS.

Este es el código completo del ejemplo:

```
<html>
<head>
<title>Barra de navegación horizontal con listas y estilos CSS</title>

<style type="text/css">
#navegador ul{
  list-style-type: none;
  text-align: center;
}
#navegador li{
  display: inline;
  text-align: center;
  margin: 0 10px 0 0;
}
#navegador li a {
  padding: 2px 7px 2px 7px;
  color: #666;
  background-color: #eeeeee;
  border: 1px solid #ccc;
  text-decoration: none;
}
#navegador li a:hover{
  background-color: #333333;
  color: #ffffff;
```

```
}
  </style>
</head>

<body>


<div id="navegador">
<ul>
<li><a href="#">Elemento 1</a></li>
<li><a href="#">Elemento 2</a></li>
<li><a href="#">Elemento 3</a></li>
<li><a href="#">Elemento 4</a></li>
</ul>
</div>

</body>
</html>
```

Se puede Ver el ejemplo en marcha en una página aparte.

*Artículo por **Miguel Angel Alvarez***

Esquinas redondeadas con CSS

Existen muchas formas de conseguir un efecto de esquinas redondeadas con CSS. En nuestro taller de CSS ya vimos algunas maneras de hacer ese efecto. En este artículo vamos a ver otras maneras interesantes porque nos permiten construir cajas de anchura y color variable. Lo malo de este método es que no funciona con Internet Explorer, pues utiliza los pseudo-elementos after o before, que por el momento no tienen soporte en el navegador de Microsoft.

Este método de conseguir las esquinas redondeadas con CSS lo he visto en el artículo Rounded corners in CSS, yo simplemente me he dedicado a adaptarlo a otro diseño. Podemos ver el resultado final del ejercicio en una página aparte.

Para el ejemplo utilizo varias imágenes que he puesto en un archivo para descarga. Como veréis, hemos utilizado imágenes en formato png, porque se adaptan mejor a nuestras necesidades, por el adecuado tratamiento que da el formato png a las transparencias.

Como decía, lo bueno de este sistema es que soporta cajas de cualquier anchura o altura. Pero no sólo eso, sino que también permite cajas de cualquier color de fondo que se quiera emplear, utilizando las mismas imágenes para crear el efecto de esquinas redondeadas.

La técnica que se utiliza en este caso es la utilización de pseudo-elementos after o before, para insertar antes y después del recuadro las imágenes con borde redondeado en las cuatro esquinas. Para ello se han creado las siguientes definiciones de estilos CSS.

```
.redondeado:before {
  background: transparent url(arr-der.png) scroll no-repeat top right;
  margin-bottom: -10px;
  height: 14px;
  display: block;
  border: none;
  content: url(arr-izq.png);
```

```
  padding: 0;
  line-height: 0.1;
  font-size: 1px;
}

.redondeado:after {
  display: block;
  line-height: 0.1;
  font-size: 1px;
  content: url(aba-izq.png);
  margin: 2px 0 0 0;
  height: 14px;
  background: transparent url(aba-der.png) scroll no-repeat bottom right;
  padding: 0;
}
```

Como se puede ver, se ha creado una clase "redondeado" utilizado las cuatro imágenes de las esquinas, insertándolas como contenido y fondo de los elementos de antes y después del elemento HTML que deseamos que aparezca redondeado.

La definición de la clase redondeado se completa con otras declaraciones de estilos:

```
.redondeado * {
  padding-left: 25px;
  padding-right: 25px;
}

.redondeado {
  padding: 0;
  background: #993333;
  color: white;
  margin-right: -1px;
}
```

La primera sirve para indicar que cualquier etiqueta o elemento que se coloque dentro de la caja con class=redondeado tenga un espaciado a la derecha y la izquierda.

En la segunda declaración se añaden nuevos estilos a la clase redondeado, entre los que se incluye el color de fondo.

El código HTML para hacer una capa con esquinas redondeadas sería el siguiente:

```
<div class="redondeado">
<p>
Texto de la caja con esquinas redondas?
</p>
</div>
```

Se puede ver que el código HTML queda extremadamente limpio.

Podemos experimentar con diversos colores de fondo para el elemento o la caja con las esquinas redondeadas, para adaptarlas a nuestras necesidades. Dejo aquí dos enlaces al mismo ejercicio simplemente cambiando el color de fondo:

- Ejemplo con el color de fondo en rojo.
- Ejemplo con una textura o imagen de fondo.

Se puede jugar también con otras imágenes, pero si cambiamos el tamaño de las imágenes tendremos que cambiar algunos valores de márgenes de los pseudos elementos after y before.

Lo mejor es que veáis el código fuente de los resultados finales para entenderlos y adaptarlos a vuestras necesidades.

Esta técnica tiene un problema, lamentablemente bastante gordo, que no funciona en Internet Explorer porque no tiene en cuenta los pseudo-elements after y before. En Internet Explorer 7 no funciona y no lo he probado todavía en Internet Explorer 8, que ya está en fase beta. Pero esperemos que el navegador de Microsoft se adapte pronto a las características avanzadas de CSS2.

*Artículo por **Miguel Angel Alvarez***

Alineación vertical con CSS

Para los que estábamos acostumbrados a utilizar tablas para maquetar páginas, nos resulta extraño que con CSS no se haya pensado una manera de alinear verticalmente un contenido en un contenedor. Cuando teníamos un contenido metido en una celda de una tabla utilizábamos el atributo valign=middle para alinearlo al centro vertical de la celda.

Pero en las versiones actuales de CSS (vamos por la CSS2), no se ha incorporado una forma de definir la alineación vertical de los elementos. Me figuro que habrá una razón especial por la que se ha decidido no incorporar la alineación vertical en etiquetas DIV, pero lo cierto es que los diseñadores o maquetadores necesitamos ese atributo, por lo menos en algunas ocasiones, para realizar nuestro trabajo.

Dentro de poco es de prever que sacarán CSS3, igual entonces hay una manera de alinear verticalmente un contenido en una capa, pero mientras tanto debemos utilizar algún truco para conseguirlo.

Ahora presentaré un hack CSS para alineación vertical de elementos en las páginas. Los llamados hack CSS son como trucos para definir estilos que dan problemas en distintos navegadores o no existen maneras de definirlos con las herramientas actuales de CSS. Explicamos y dimos ejemplos de hacks CSS en un artículo anterior.

Hack CSS para alineación vertical por medio de una imagen

Vamos a mostrar un truco sencillo para alinear verticalmente con CSS utilizando una imagen. Aprovecharemos que la imagen tiene un atributo vertical-align:middle; que sirve para que el texto que hay después de la imagen esté alineado a su mitad vertical.

Entonces haremos algo como esto.

Definimos los estilos para una imagen:

```
<style type="text/css">
img.valign {
  height: 100%;
  vertical-align: middle;
  width: 0px;
}
</style>
```

En los estilos de la imagen definimos que tenga height 100% para adaptarse a la altura del contenedor donde la hayamos metido. vertical-align:middle sirve para que el texto que haya antes o después de la imagen. El atributo width: 0px nos sirve simplemente para que la imagen no tenga anchura, puesto que no queremos mostrar ninguna imagen ni que esta tome espacio en la página, sólo queremos alinear verticalmente.

Ahora podremos crear una capa con un contenido alineado en la vertical, de esta manera:

```
<div style='height:330px; background-color: #ccccff;'>
Contenido alineado verticalmente. <img class="valign" />
</div>
```

La etiqueta DIV puede tener la altura que deseemos. El color de fondo simplemente lo he colocado para que se vean sus límites.

Podemos ver la imagen que se ha colocado dentro de la capa, que tiene class definida en el css anterior. No tiene src ni nada, pues no necesitamos colocarlo porque no queremos mostrar ningún archivo como imagen.

Podemos ver el ejemplo en marcha en una página aparte.

El ejemplo funciona perfectamente en Internet Explorer, Firefox y Opera, con lo que es un hack css perfectamente usable y compatible con la mayoría de sistemas.

Lo malo es que esta alineación vertical sólo funcionará en caso que el contenido de la capa no supere una línea. En el momento que el texto de la capa tenga varias líneas, sólo se alineará verticalmente una de ellas, la primera o la última, dependiendo de donde se haya colocado la imagen, delante o detrás del texto.

*Artículo por **Miguel Angel Alvarez***

Propiedades de impresión CSS page-break-after y page-break-before

Existen propiedades de CSS como page-break-after y page-break-before que sirven para alterar el aspecto de las páginas web al ser impresas en papel. Estos atributos sirven en concreto para forzar saltos de página antes o después de elementos de la página. Con page-break-after podemos definir si deseamos un salto de página después de un elemento, mientras que con page-break-before definimos la posibilidad de incorporar un salto de página antes de un elemento.

Recordemos que con CSS se pueden definir tanto los estilos que deseamos aplicar a la hora de ver la página en el monitor como a la hora de imprimirla con una impresora. Eso ya lo explicamos en el artículo CSS para imprimir páginas web. Pero estos dos atributos son un poco distintos, porque los podemos utilizar libremente en la especificación de estilos en global de la web, siendo que, tanto page-break-after como page-break-before, no tienen representación en el momento de visualización de la página en la pantalla del ordenador, pero sí a la hora de imprimirla en la impresora.

Atributo CSS page-break-after

Sirve para forzar, o no, un salto de línea después de un elemento de la página. Se puede colocar en elementos de bloque, como tablas o capas DIV. Un uso típico sería:

```
<div style="page-break-after: always;">
Contenido del div
...
</div>
```

Entonces, a la hora de imprimir la web, se insertará un salto de línea después del bloque DIV.

Atributo CSS page-break-before

De manera similar al anterior atributo, page-break-before sirve para decir si queremos, o no, forzar un salto de página antes de un elemento. Si lo colocamos con el valor always, haremos que siempre se cambie de página antes de imprimir dicho elemento.

Esto puede venir muy bien para, por ejemplo, tablas u otros elementos que no queremos que se rompan en dos partes, si coincide un salto de página en medio de la impresión de la página. Con ello hacemos que se cambie de página, con un salto de página forzado y así en la nueva página es más fácil que tengamos espacio suficiente para que quepa la impresión de ese elemento.

Por ejemplo, podría darse este uso:

```
<table border="1" style="page-break-before: always;">
Otras etiquetas de la tabla
...
</table>
```

Otros posibles valores de los atributos

Tanto page-break-after como page-break-before permiten otros tipos de valores, aparte del always que hemos visto en los dos ejemplos anteriores.

- **auto**: es el valor por defecto. Sólo coloca el salto de página si es necesario..
- **always**: coloca siempre un salto de línea, después o antes del elemento.
- **avoid**: evita colocar un salto de línea antes o después.
- **left**: inserta uno o dos saltos de página, de modo que se pueda asegurar que la siguiente página es una página izquierda (page-break-after), o para asegurar que la página donde se empieza el elemento es una página izquerda (page-break-before). Imaginemos un libro abierto, que tiene páginas a la izquierda y a la derecha para saber a lo que se refiere una página izquierda.
- **right**: inserta uno o dos saltos de página, para asegurar que se pueda insertar el elemento al principio de una página de derecha (page-break-before) o para asegurar que después del elemento comience una página derecha (page-break-after).

Compatibilidad de page-break-before y page-break-after con distintos navegadores

En principio, tanto Internet Explorer como Firefox y otros navegadores como Opera, soportan este atributo. El problema es que en Internet Explorer, al menos en la versión 7, funciona unas veces y otras no. Esto es una pena, porque no podemos estar seguros si la impresión va a ser tal como hemos definido con CSS para el documento.

Para solucionarlo a mi me ha ido bien colocando un DIV vacío (pero con un espacio en blanco) antes o después del elemento al que queremos asignar los saltos de página. Luego poniendo el page-break-after o page-break-before a ese DIV vacío. Además de separar por un <p> el DIV vacío del elemento. El código sería algo parecido a esto:

```
[Contenido de la página]
<div style="page-break-before: always;"> </div>
<p>
<table cellspacing="2" cellpadding="6" border="1" >
<tr>
  <td>Esta tabla</td>
  <td>Aparecerá en</td>
  <td>una nueva página</td>
</tr>
</table>
[Más contenido de la página]
```

Como se puede deducir del ejemplo, la página tendrá un contenido. Luego se forzará un salto de página antes del DIV que tiene el , de modo que, al imprimir, la tabla siguiente aparecerá siempre al principio de una página nueva.

He visto por ahí otros hack CSS para solucionar la compatibilidad de Explorer con estos atributos, pero complican un poco la solución. Así a mi me ha funcionado correctamente siempre el page-break-after o page-break-before en todos los navegadores.

*Artículo por **Miguel Angel Alvarez***

Columnas de altura 100% con CSS

Con CSS podemos hacer muchas cosas de una manera rápida y sencilla. Pero muchas veces tenemos que emplear técnicas, algunas sencillas y otras no tanto, para conseguir efectos que de otra manera son imposibles. Es el caso que nos ocupa en este artículo, conseguir que las columnas en una maquetación CSS ocupen el 100% de la altura disponible. Es decir, tener una página maquetada a varias columnas, donde todas las columnas llegan hasta la parte de abajo de la página.

Quizás hayas llegado a este artículo porque no consigas un height: 100% en una columna al diseñar tu página con CSS y no necesites más explicaciones de lo que pretendemos conseguir. Pero si no es el caso y deseas ver un ejemplo de cómo se haría una maquetación CSS a dos columnas accede al artículo de DesarrolloWeb.com Maquetación CSS a dos columnas. Luego echa un vistazo a la página del ejemplo que se desarrolla en el artículo anterior.

Verás que la columna que tiene una especie de barra de navegación lateral sólo ocupa un área limitada del espacio que hay en vertical. Esto es porque en CSS las capas DIV crecen verticalmente justo lo necesario, atendiendo a los contenidos que se tienen que colocar dentro. Por mucho que intentemos colocar un estilo CSS height:100% a la capa de la columna pequeña, en principio, no conseguiremos que ocupe el mismo espacio que la columna grande. Pero claro, hay una solución a todo esto y podemos estar tranquilos porque es muy simple.

Antes de continuar, puedes ver el ejemplo de maquetación CSS que vamos a explicar en este artículo.

Referencia: Te dejamos aquí un enlace a un artículo del taller de CSS que tiene relación con este y que tal vez te interese también leer para encontrar otros trucos relacionados con capas de height: 100%.

El pequeño truco de maquetación consiste en simular las columnas con un fondo. En realidad vamos a tener columnas normales, con capas DIV transparentes, pero las vamos a situar sobre un fondo de color que será el que ofrezca el acabado en columna.

El fondo para simular las columnas

El fondo lo crearemos con el espacio que nosotros deseemos, es decir, definiremos en el fondo la distribución en columnas que necesitemos para nuestro proyecto. Haremos un fondo como este:

Nota: la imagen del fondo está distorsionada en tamaño, para que quepa en esta página y se pueda leer bien el artículo. Nuestro fondo original mide 1280x50 pixel y está creado con espacios optimizados para definiciones de pantalla de 1024 pixels de ancho o superior.

Como se puede ver en la imagen, tenemos un listón que tiene varias columnas, al repetirse en la vertical hará una columna tan alta como los contenidos de la página requieran.

Para poner la imagen de fondo, en el body o en el contenedor donde queramos simular las columnas, haremos un CSS como este:

```
background: #f0f0f0 url(fondo.gif) repeat-y 50%;
```

El atributo css tiene un valor position, que en este caso es 50%, con lo que conseguimos que el fondo se centre en la página.

Creamos las columnas con capas flotando a izquerda y derecha

Luego ya sólo sería crear un contenedor que aparezca en el centro de la página y un par de columnas izquierda y derecha, donde estén flotando cada una hacia su lado. En otros artículos de desarrollo web .com hemos visto cómo hacer estas cosas con CSS.

El códido HTML quedaría así:

```
<div id="container">
  <div id="izquierda">
    Columna de la izquierda
  </div>
  <div id="derecha">
    Esta columna de la derecha
  </div>
</div>
```

El código CSS del ejemplo sería como este:

```
body {
  background: #f0f0f0 url(fondo.gif) repeat-y 50%;
```

```
  text-align: center;
}
#container{
  margin:auto;
  text-align:left;
  width: 1000px;
}
#izquierda{
  width: 670px;
  float: left;
  margin: 10px 0px 10px 10px;
}
#derecha{
  width: 290px;
  float: right;
  margin: 10px 5px 10px 0px;
}
```

No tiene mucho misterio, si es que ya hemos seguido otros talleres publicados en desarrolloweb .com sobre la cómo maquetar webs con CSS. Os aconsejamos seguir la línea de artículos que comienza por el artículo Maquetación CSS.

Finalmente, os pasamos el enlace para ver el ejemplo en funcionamiento.

*Artículo por **Miguel Angel Alvarez***

Opacidad CSS

En este artículo vamos a entrar en detalles sobre la opacidad en CSS. Veremos cómo alterar la transparencia de los elementos de la página a través de atributos CSS.

Aunque parezca una contradicción, la opacidad sirve para definir la transparencia de los elementos de la página. Esto es porque todos los elementos de una página son 100% opacos por defecto. Es decir, no se puede ver a través de ellos y por tanto al sobreponerse tapan totalmente lo que haya debajo. Así pues, a través de la opacidad podremos especificar cuanto de opaco será un elemento, o lo que es lo mismo, cuan transparente sea.

La opacidad se trabaja de dos maneras distintas en los navegadores más comunes. Por un lado tenemos a Internet Explorer, y por otro lado a todos los demás navegadores, es decir, Firefox, Opera o Chrome. Por ello, para especificar la opacidad o transparencia tendremos que hacerlo con dos atributos distintos. En DesarrolloWeb.com ya habíamos tratado este tema con anterioridad, pero ahora vamos a ofrecer explicaciones más completas.

Opacidad en Internet Explorer

En Explorer tenemos que indicar el grado de opacidad como un filtro de CSS. Dichos filtros se pueden aplicar con el atributo filter de hojas de estilos. Existen varios filtros disponibles para hacer distintos efectos sobre elementos de la página, en concreto el filtro opacity es el que nos interesa para definir transparencias. Se especifica de la siguiente manera:

```
.claseopacidad{
  filter:alpha(opacity=25);
}
```

Con esta declaración de estilos conseguiríamos que los elementos de la clase claseopacidad tuvieran un 25% de opaco, o lo que es lo mismo, un 75% de transparente.

Ahora bien, en Explorer no sé por qué, podemos encontrar un problema al utilizar el filtro opacity. Simplemente en algunos casos no funcionará y no veremos la transparencia que hayamos declarado. Esto se soluciona aplicando algún estilo adicional, aparte del propio filtro CSS. Por ejemplo, ponerle un ancho a la capa bastaría, o también colocarle un float:letf. Es importante saber esto, puesto que es el típico bug que si no lo sabes puedes acabar con un dolor de cabeza antes de dar con él.

Quedaría por ejemplo así:

```
.claseopacidad {
  filter:alpha(opacity=25);
  width: 120px;
}
```

Opacidad en Firefox, Opera, Chrome...

En estos navegadores se define la transparencia de una manera más simple. Lo haremos a través del atributo opacity, de esta manera:

```
. claseopacidad {opacity: .5}
```

Con esto conseguiremos que la clase sea 50% transparente, o 50% opaca, como prefiramos decir.

> **Nota:** Está bien comentar que en versiones antiguas de Firefox, y productos derivados de Mozilla, este atributo no funcionaba. Hace años que ya está integrado en el navegador y funciona sin problemas, pero durante un tiempo había que utilizar otro atributo propio de Mozilla.

```
-moz-opacity: .25
```

Con esto indicamos una transparencia del 75%. Este atributo hoy no hace falta utilizarlo, pero si lo ponemos no es mala idea para aumentar la compatibilidad de la página con versiones anteriores del navegador.

Transparencia para todos los navegadores

Finalizamos este artículo de desarrollo web .com, poniendo todos los conocimientos adquiridos a la vez. Para declarar un estilo de transparencia que sirva para todos los navegadores simplemente tenemos que poner las distintas declaraciones de estilos todas juntas. Cada browser interpretará la que conoce.

```
.transparencia {
  opacity: .25;
  -moz-opacity: .25;
  filter:alpha(opacity=25);
  float: left;
}
```

Podemos ver un ejemplo sobre cómo quedaría la opacidad declarada en una página aparte.

*Artículo por **Miguel Angel Alvarez***

Estilos de borde CSS

CSS permite crear varios tipos de bordes en elementos de la página. El borde más habitual es el de una línea lisa, pero también hay otros tipos de bordes que podemos implementar como una línea de puntos, bordes redondeados, etc. En este artículo de DesarrolloWeb.com veremos las distintas posibilidades de creación de bordes en CSS.

Lo cierto es que en DesarrolloWeb.com ya hemos hablado varias veces de bordes con CSS, sobre todo para explicar modos de hacer los bordes redondeados, que suelen quedar muy atractivos en las páginas web. De todos modos, vamos a ver cómo se especifica un borde con CSS.

```
DIV {
  border: 1px solid #d0d0d0;
}
```

Así conseguiremos que todas las etiquetas DIV de la página tengan un borde de 1 píxel, sólido (de una línea sólida) y con color grisáceo (#d0d0d0).

Lo que vamos a ver en este artículo es los distintos estilos de borde, aparte del solid que ya conocemos. Para ilustrar el ejercicio, he creado una página con los distintos ejemplos de bordes. Pero ya que estamos hablando de bordes, sobre su declaración en CSS, cabe decir que podríamos utilizar otra notación distinta, en la que especificamos por separado los tres atributos del borde, como sigue:

```
.bordesolido{
  border-color: #aaaaaa;
  border-width: 1px;
  border-style: solid;
}
```

También podemos especificar los bordes por separado de cada uno de los lados del elemento, de esta manera:

```
.bordeporlados{
  border-top: 1px solid #ff9999;
  border-right: 2px dotted #99ff99;
  border-bottom: 2px dashed #9999ff;
  border-left: 4px double #666666;
}
```

Todos estas indicaciones para la construcción de bordes las hemos visto en anteriores artículos de CSS de desarrollo web .com, como el manual de css o el taller de css.

Estilos de borde

Con el atributo border-style podemos definir varios estilos de borde, entre una gama relativamente amplia de posibilidades. No obstante, tenemos que admitir que muchos de los estilos hacen bordes un poco feos, que deberíamos utilizar con bastante cuidado.

Los posibles estilos de borde son:

- solid
- dotted
- double
- dashed
- groove
- ridge
- inset
- outset

Podemos ver todos estos estilos de borde en una página aparte.

Ahora vamos a comentar estos distintos estilos y poner ejemplos de cada uno.

Borde solid:
Este borde es de una línea sólida.

```
.bordesolido{
  border:-color: #aaaaaa;
  border-width: 1px;
  border-style: solid;
}
```

Borde dotted (punteado):
Este borde está compuesto a base de una línea de puntos.

```
.bordepunteado{
  border: 1px dotted #ff8833;
}
```

Borde double (Doble):
El borde double está compuesto por una doble línea sólida. Hay que saber que para que este borde se vea como una doble línea, tenemos que especificar un ancho de al menos 3 pixel. Así se vería un borde con dos líneas de un píxel más un espacio de separación del borde de otro pixel. A medida que especificamos una anchura mayor del borde doble, las líneas van aumentando de grosor, así como el espacio que las separa.

```
.bordedoble{
  border: 9px double #990000;
}
```

Borde dashed (línea discontinuada):
El borde dashed es muy similar al borde punteado, pero en lugar de puntos, lo que tenemos es una línea partida.

```
.bordedashed{
  border: 1px dashed #660000;
}
```

Borde groove (ranura):
Es un borde especial, con una especie de relieve difícil de describir, creado con la combinación de colores claros y oscuros de borde.

```
.bordegroove{
  border: 5px groove #66cc66;
```

```
}
```

Borde ridge (cresta):
Este borde también es un poco raro. Parecido al groove, pero con los colores oscuros y claros invertidos.

```
div.borderidge{
  border: 10px ridge #6666cc;
}
```

Borde inset (relieve hacia dentro):
Crea un borde con una imitación de biselado, como si estuviera hundido. El efecto de este borde se ve mejor si el elemento que está bordeando tiene un color de fondo parecido al color de borde.

```
div.bordeinset{
  border: 10px inset #3333ff;
  background-color: #0000ff;
}
```

Borde outset (con relieve hacia fuera):
Este último estilo de borde imita un biselado, como si fuera un botón que está levantado. Es igual que el estilo de borde inset, pero con los colores claros y oscuros invertidos. Veremos también este efecto mejor si el elemento tiene un color de fondo.

```
div.bordeoutset{
  border: 10px outset #cccccc;
  background-color: #cccccc;
}
```

De nuevo, pongo el enlace al ejemplo donde aparecen todos estos estilos de borde.

Borde redondeado con CSS

El borde redondeado, o con las esquinas en curva, se puede especificar a partir de CSS3. Esto lo hemos explicado ya en el artículo: Bordes redondeados en CSS 3

También podemos hacerlo por medio de distintas técnicas, que no requieren de CSS3, que aun no está soportado por todos los navegadores. Podéis consultar otros artículos de DesarrolloWeb.com que ofrecen esas técnicas: Caja con borde y esquinas redondeadas.

*Artículo por **Miguel Angel Alvarez***

Atributo gradiente de colores en borde con CSS y Firefox

Investigando un poco con algunas de las propiedades nuevas que va a traer CSS 3, para completar la información del artículo Introducción a CSS 3 , he dado con un atributo para cambiar el color del borde de los elementos de la página, que nos permite definir el color con una sucesión de distintos valores de colores. En la actualidad, con CSS podemos definir por separado los colores del borde de un elemento, de arriba, derecha, abajo e izquierda. Pero no nos referimos a poder dar un color por separado para cada parte del borde, sino aplicar varios colores distintos a una parte, por ejemplo a la parte de arriba del borde.

Esta propiedad la he visto comentada en algún lugar como de CSS 3, pero según veo en el borrador de la especificación de CSS 3 para bordes y fondos, publicado por el W3C, no aparece esta nueva característica como parte del nuevo estándar.

Por ello, habría que matizar que esta propiedad no es de CSS 3, sino que se trata de un atributo no estándar de CSS, creado por Mozilla y que, por tanto, se puede ver en su navegador más conocido: Firefox. Dicho de otra forma, es una extensión de CSS realizada por Mozilla. En la página de Mozilla podemos encontrar más información sobre esta extensión de CSS: https://developer.mozilla.org/en/CSS/-moz-border-bottom-colors

Los atributos a los que nos referimos, que permiten definir varios colores para cada una de las partes del borde, son los siguientes:

```
-moz-border-top-colors
-moz-border-right-colors
-moz-border-bottom-colors
-moz-border-left-colors
```

En cada uno de los atributos se define el color, pudiendo especificar una lista de colores, separados por espacios, que se aplicarán a cada una de las partes del borde, de dentro hacia fuera. Claro que el borde tiene que tener una anchura mayor que 1 para que se vean los distintos colores. Con una anchura de 2 pixel se podrán ver 2 colores distintos, con una anchura de 3 podremos definir 3 colores y así sucesivamente.

Para ver una de las posibilidades que daría el uso de este atributo podemos ver un ejemplo en una página aparte. (Pero tener en cuenta que sólo lo veréis correctamente en Firefox).

El código para crear ese gradiente de colores sería el siguiente:

```
<style type="text/css">
.coloresborde{
  border-style: solid;
  border-width: 10px;
  -moz-border-top-colors: #ffcc99 #ffbb88 #ffaa77 #ff9966 #ff8855 #ff7744 #ff6633 #ff5522 #ff4411 #ff3300;
  -moz-border-right-colors: #ffcc99 #ffbb88 #ffaa77 #ff9966 #ff8855 #ff7744 #ff6633 #ff5522 #ff4411 #ff3300;
  -moz-border-bottom-colors: #ffcc99 #ffbb88 #ffaa77 #ff9966 #ff8855 #ff7744 #ff6633 #ff5522 #ff4411 #ff3300;
  -moz-border-left-colors: #ffcc99 #ffbb88 #ffaa77 #ff9966 #ff8855 #ff7744 #ff6633 #ff5522 #ff4411 #ff3300;
}
</style>
```

Es una pena que sea un atributo únicamente desarrollado por Mozilla, aunque si el soporte a estos atributos se lleva a cabo por más navegadores y la W3C lo tiene a bien, quizás en algún momento se convierta en un estándar de CSS.

Por el momento no vale para mucho más que una mera curiosidad y posibilidad de personalización especial para los usuarios de Firefox y otros navegadores basados en Mozilla.

*Artículo por **Miguel Angel Alvarez***

Múltiples imágenes de fondo con CSS

Con el atributo background-image podemos conseguir que un elemento de la página tenga un fondo de imagen. Esto debemos saberlo, puesto que es algo básico de las hojas de estilo en cascada (CSS). Pero en este artículo de DesarrolloWeb.com vamos a mostrar cómo podríamos conseguir que un elemento de la web tuvisese varias imágenes de fondo al mismo tiempo. Colocar varias imágenes de fondo a un elemento en principio no se puede con CSS, por lo que mostraremos cómo hacerlo con una sencilla técnica alternativa. Pero también veremos que en CSS 3 existe la posibilidad de configurar varios fondos al mismo tiempo en un elemento, con una declaración de background-image y sin necesidad de realizar ningún tipo de técnica alternativa.

> **Referencia:** Para conocer un poco de la reciente especificación de CSS3 os vendrá bien leer el artículo Introducción a CSS 3.

En CSS normal (CSS 1), como hemos dicho, podemos colocar un único fondo a un elemento de la página. Esto es algo soportado por todos los navegadores. Como no podemos poner más de 1 fondo por elemento, para colocar varios fondos al mismo tiempo, tenemos que utilizar varios elementos. A cada elemento le colocaremos un único fondo, pero al mostrarse los elementos en el mismo espacio, conseguiremos el efecto deseado de tener varios fondos de imagen a la vez.

En nuestro caso, vamos a utilizar varias capas DIV anidadas y cada una tendrá su fondo de imagen.

El código HTML que utilizaríamos para anidar varias capas DIV sería como sigue:

```
<div id="fondo1">
  <div id="fondo2">
    <div id="fondo3">
      contenido del elemento que va a tener 3 fondos de imagen distintos
      ...
    </div>
  </div>
</div>
```

Como se ha visto, podemos anidar varias capas DIV y cada una tendrá un identificador, o si lo preferimos una clase, para asignar estilos por CSS. Al estar anidados, todos los elementos DIV se mostrarán uno encima del otro.

Ahora veamos el código CSS para darle fondos a cada uno de estos elementos DIV:

```
<style type="text/css">
#fondo1{
  background-image: url(fondo1.gif);
  width: 300px;
}
#fondo2{
  background-image: url(fondo2.png);
  background-repeat: no-repeat;
  background-position: bottom right;
}
#fondo3{
  background-image: url(fondo3.png);
  background-repeat: no-repeat;
  background-position: center;
}
</style>
```

Los fondos se superpondrán unos a otros, siendo el fondo1 el que se vea más abajo y el

fondo3 el que se verá más arriba.

El resultado de este ejemplo se puede ver en una página aparte.

En principio todos los navegadores visualizarán perfectamente los fondos, pero como he utilizado imágenes transparentes en formato PNG e Internet Explorer 6 tiene problemas con la transparencia de los archivos PNG, podemos encontrar algún defecto, pero los fondos de los elementos DIV se verán.

Colocar varios fondos de imagen a un elemento con CSS 3

Una de las nuevas características de CSS 3 consiste en la posibilidad de declarar varios fondos de imagen a un elemento de la página. Lo que antes hemos visto que es posible, creando varios elementos anidados y colocando un fondo en cada uno, se puede hacer en CSS 3 con un solo elemento, al que aplicaremos varios fondos distintos.

El HTML del ejemplo de varias imágenes de fondo sería el siguiente:

```
<div id="fondos">
  texto de un único elemento
  ...
</div>
```

Ahora veamos el CSS 3 válido para este ejemplo:

```
<style type="text/css">
#fondos{
    background: url(fondo3.png) bottom right no-repeat,
    url(fondo2.png) center no-repeat,
    url(fondo1.gif) center repeat;
    width: 300px;
}
</style>
```

Sólo cabe comentar que las distintas imágenes de fondo se tienen que escribir en la declaración CSS separadas por comas. Además, las imágenes que declaramos se van colocando de modo que la primera aparece sobre las siguientes. Así pues, en esta declaración, fondo1.gif, que está colocada como último fondo, es la que aparece detrás del todo.

De momento, la posibilidad de incluir varios fondos de imagen a un elemento sólo está disponible en el navegador Safari, pero esperemos que pronto otros navegadores vayan incorporando esta funcionalidad de CSS 3.

Este ejemplo se puede ver en una página aparte.

*Artículo por **Miguel Angel Alvarez***