Introducción a CSS3 (Parte 2)

1. INCRUSTACIÓN DE FUENTES

Una de las limitaciones más importantes a la que se enfrenta un diseñador de páginas web la encuentra en el uso de **fuentes** o tipos de letra especiales para los elementos de texto.

Si el diseñador se limita al uso de las fuentes "típicas", es decir, aquellas que casi con total seguridad dispondrá cualquier usuario en su ordenador, esto no es un problema; pero si utiliza fuentes distintas y no tan habituales, se encontrará con dificultades.

Aquí puedes encontrar un listado de las llamadas "*fuentes web seguras*", es decir, de las fuentes que normalmente no presentan problemas, ya que la mayoría de los usuarios las tienen instaladas en sus equipos:

http://fluidwebtype.info/web-safe-fonts

Y es que si el usuario no dispone de las fuentes que ha utilizado el diseñador, el aspecto final del texto puede variar considerablemente, sobre todo cuando se utiliza la fuente predeterminada del navegador.

CSS3 soluciona este problema mediante la **incrustación de fuentes**. Esto quiere decir que el usuario puede descargarse en su equipo las fuentes que se han utilizado en la página web, de forma similar a lo que ocurre con las imágenes.

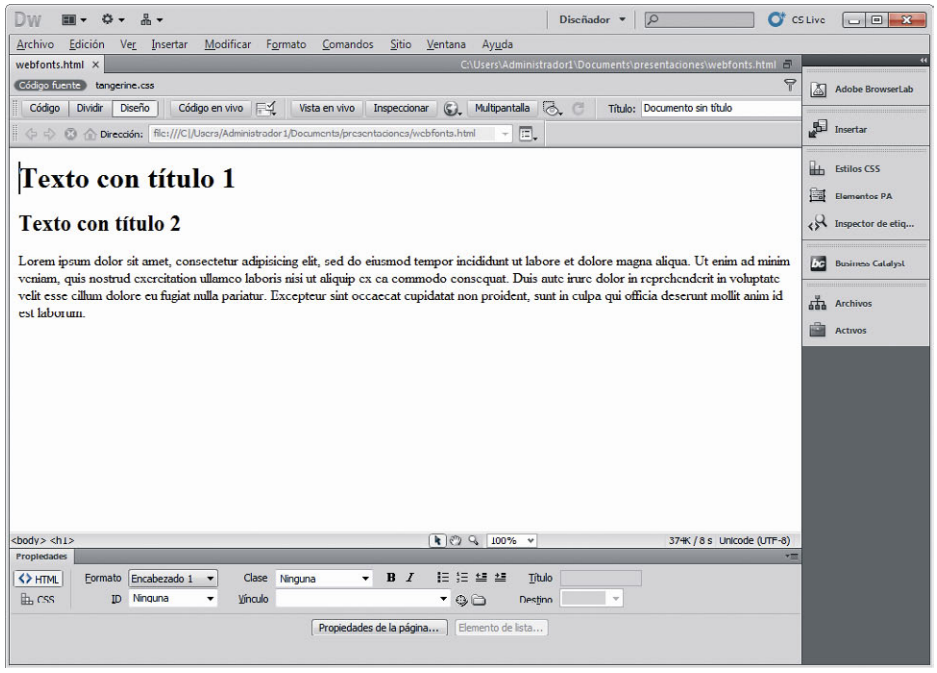
Esto tiene algunas consideraciones añadidas:

- La primera es que el usuario se descargará el archivo correspondiente a la fuente utilizada y que no dispone en su equipo. Por lo tanto, debemos tener en cuenta el **tamaño** de esos archivos.

- Por otra parte, la mayoría de las fuentes tendrán limitaciones de **licencia** en cuanto a su distribución.

Olvidando por el momento estas dos consideraciones, vamos a ver cómo podemos incrustar fuentes mediante CSS3. Para ello, se utiliza la regla **@font-face**.

En la figura de la página siguiente tenemos una página web con cierto texto. Por ahora, el tipo de letra es el predeterminado para los títulos **h1** y **h2**, así como para los párrafos **p**.



Ahora definimos esta regla **@font-face** para incrustar una fuente determinada:

```
@font-face {
 font-family: 'TangerineRegular';
 src: url('../fonts/Tangerine_Regular-webfont.eot');
 src: local('TangerineRegular'),
   url('../fonts/Tangerine_Regular-webfont.eot?#iefix')
   format('embedded-opentype'),
   url('../fonts/Tangerine_Regular-webfont.woff')
   format('woff'),
   url('../fonts/Tangerine_Regular-webfont.ttf')
   format('truetype'),
   url('../fonts/Tangerine_Regular-webfont.svg#TangerineRegular')
   format('svg');
 font-weight: normal;
 font-style: normal;
}
```

Esta regla define la fuente incrustada. Para ello, se utiliza:

- La propiedad **font-family**: en este caso no tiene mucha importancia, ya que simplemente es el nombre con el que será conocida la fuente que vamos a utilizar. Puedes utilizar el nombre de la misma fuente o cualquier otro nombre que te ayude a identificarla después.

- La propiedad **src**: se utiliza para indicar la ubicación del archivo de la fuente.

Como puedes ver, esta se encuentre en la subcarpeta **fonts** del sitio web. Es habitual guardar las fuentes en una carpeta específica, como ocurre con las imágenes, hojas de estilo o *scripts*.

Aquí viene otro problema y es que no todos los navegadores y dispositivos admiten los mismos tipos de fuentes:

- Internet Explorer solo admite el tipo de fuente **EOT**.
- La mayoría de los otros navegadores admiten los tipos **TTF** y **OTF**.
- Algunos navegadores para dispositivos móviles aceptan únicamente el tipo **SVG**.
- Se está desarrollando otro formato para unificar todo esto, que es el tipo de fuente **WOFF**.

Esto significa que tendrás que proporcionar la misma fuente en distintos formatos si deseas ser lo más compatible posible. Al menos en formato **EOT** (para Internet Explorer) y **TTF/OFT** (para la mayoría de los otros navegadores).

Esto es lo que se hace en esta regla. Como puedes ver, existen dos propiedades **src**. La primera hace referencia al formato admitido por Internet Explorer, mientras que en la segunda se indica la ubicación del resto de archivos.

Con Internet Explorer 9 o posterior este truco ya no es necesario, pero no está de más ponerlo.

Como Internet Explorer solo es compatible con los archivos **.eot**, este debe ser el primero que aparezca. Además, tenemos que evitar que intente descargar cualquiera de los otros archivos.

Para ello, se utiliza un pequeño truco y es incluir el operador **local**.

Como Internet Explorer no entiende este operador, no hace caso de la segunda propiedad **src**. Sin embargo, el resto de navegadores cogerán el archivo correspondiente al formato más adecuado para ellos.

El nombre que se utiliza entre los paréntesis del operador **local** no tiene mucha importancia, por lo que puedes poner el mismo nombre que habías utilizado con **font-family**.

Fíjate que aunque está dispuesta en varias líneas, realmente estamos incluyendo una única segunda propiedad **src**. Se separan con comas los distintos archivos y se indica el formato de cada uno de ellos.

Dentro de la misma regla **@font-face** podemos establecer otros detalles, como **font-weight** y **font-style**. Por ejemplo, para especificar los realces negrita o cursiva.

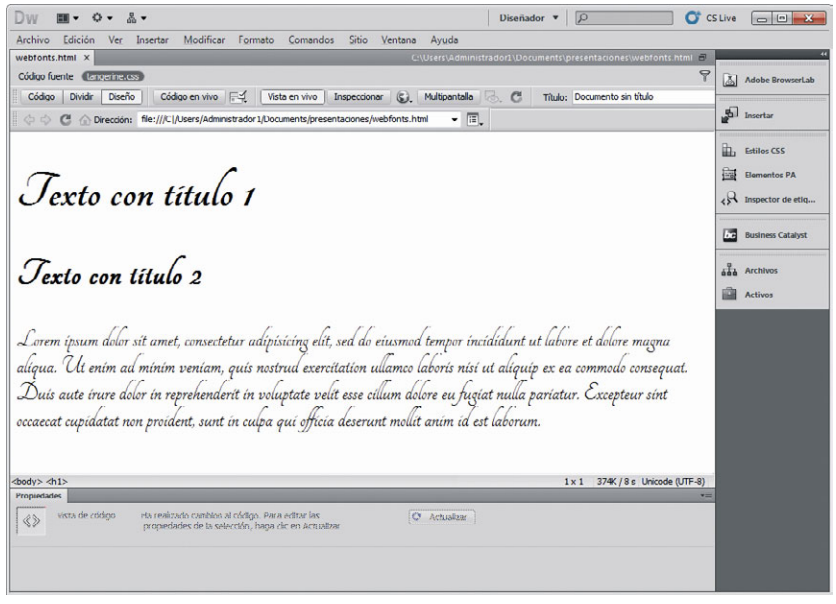
Aunque hemos visto la regla @font-face muy completa, normalmente incluirás solo las fuentes .eot y .ttf, ya que estos dos formatos son los más utilizados.

Además, es posible que tengamos más de una versión de la misma fuente. Por ejemplo, en el caso de la fuente que hemos utilizado, también la tenemos en negrita:

```
@font-face {
 font-family: 'TangerineBold';
 src: url('../fonts/Tangerine_Bold-webfont.eot');
 src: local('TangerineBold'),
   url('../fonts/Tangerine_Bold-webfont.eot?#iefix')
   format('embedded-opentype'),
   url('../fonts/Tangerine_Bold-webfont.woff')
   format('woff'),
   url('../fonts/Tangerine_Bold-webfont.ttf')
   format('truetype'),
   url('../fonts/Tangerine_Bold-webfont.svg#TangerineBold')
   format('svg');
 font-weight: normal;
 font-style: normal;
}
```

Una vez has escrito la regla o reglas **@font-face**, ya puedes utilizar la nueva fuente exactamente igual que lo harías con cualquier otra:

```
h1 {
   font-family: "TangerineBold", serif;
   font-size: 4em;
}
h2 {
   font-family: "TangerineRegular", serif;
   font-size: 3em;
}
p {
   font-family: "TangerineRegular", serif;
   font-size: 2em;
}
```



Ahora puedes ver el aspecto de las fuentes que se han utilizado. Si el usuario no dispone de esa fuente, primero se descargará en el equipo.

Como has comprobado, es sencillo utilizar la regla **@font-face** para incrustar fuentes que has utilizado en la página web. Veamos ahora dónde encontrar esas fuentes y qué limitaciones tenemos.

2. DÓNDE OBTENER FUENTES

El sistema operativo, ya sea Windows, Mac o Linux, viene con un determinado conjunto de fuentes. Además, a medida que instalamos aplicaciones, estas también pueden añadir fuentes a nuestra disposición.

Bien, el problema es que normalmente esas fuentes tendrán limitaciones en cuanto a su distribución. Incluso si las has comprado directamente, debes comprobar lo que se indica al respecto en la licencia de la fuente.

Esto puede ser un inconveniente, por lo que han aparecido algunos proyectos para recopilar fuentes de libre uso y distribución.

Entre esos proyectos debemos destacar dos: **Font Squirrel** (*www.fontsquirrel.com*) y **Google Font Directory** (*www.google.com/webfonts*).

Accediendo a cualquiera de estos dos sitios web podrás descargar los archivos de fuentes libres e incluso el código necesario para utilizarlas.

Adicionalmente existen sitios web donde conseguir fuentes libres y de pago, como:

TypeKit (*www.typekit.com*)

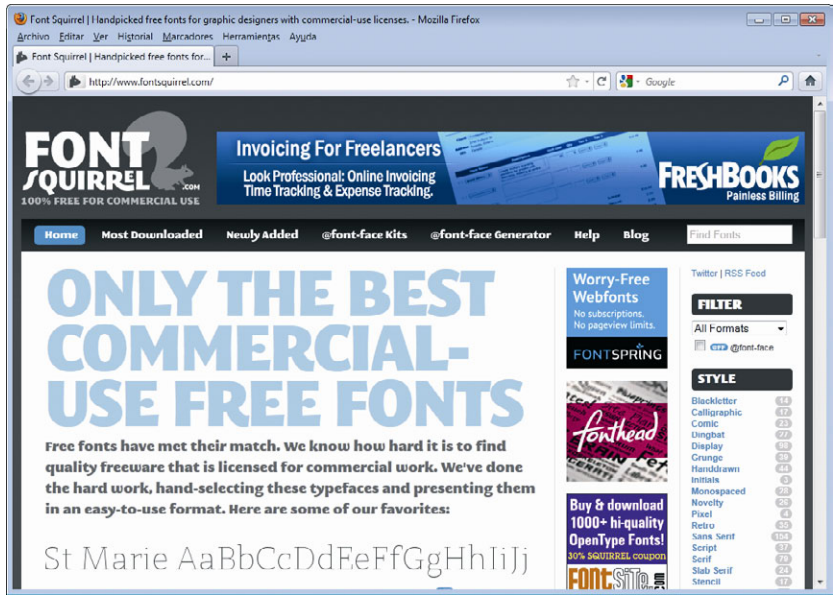
Kernest (*www.kernest.com*)

Ascender (*www.ascender.com*)

Typotheque (*www.typotheque.com*)

Fontdeck (*www.fontdeck.com*)

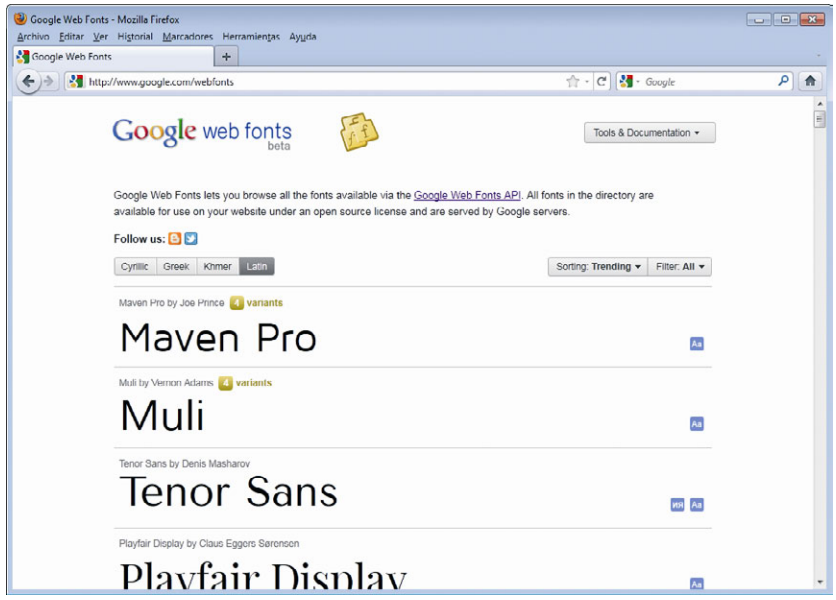
En la figura siguiente se muestra el aspecto del sitio web **Font Squirrel**, donde puedes encontrar un gran número de fuentes de libre uso y distribución.



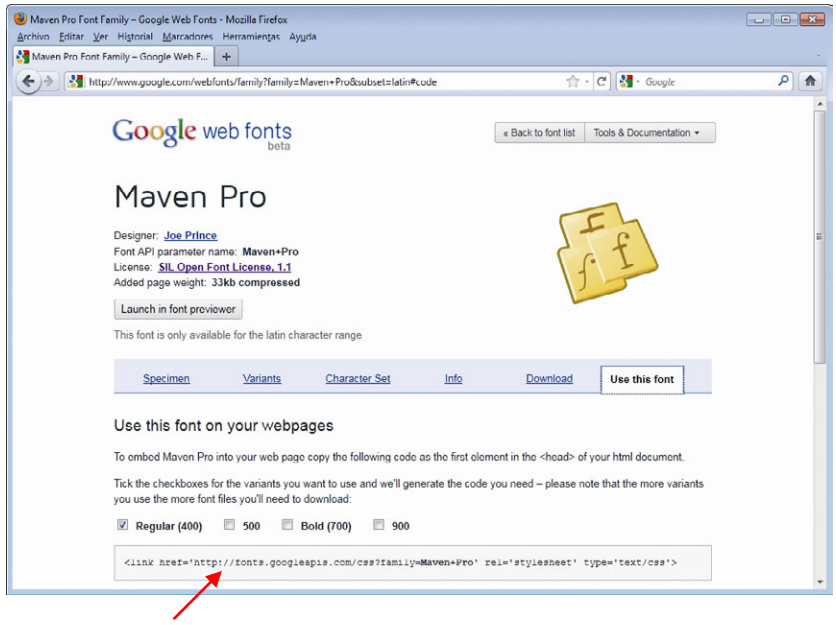
Aquí puedes descargar las fuentes, que normalmente están en formato **OTF** o **TTF**. Pero eso no es problema, ya que este mismo sitio web permite convertirlas a los otros formatos si lo necesitas. Para ello, accede a la sección **@font-face Generator**.

Pero es que además, podemos descargar completos kits de fuentes, con todo lo que necesitamos ya preparado. Para ello, accede a la sección **@font-face Kits**. Cualquiera de esos kits incluye todos los archivos correspondientes a los distintos formatos de fuente, una hoja de estilo CSS con el código necesario para su utilización y un archivo .html de prueba.

Por su parte, con **Google Font Directory** no necesitamos preocuparnos del código CSS.



Podemos descargar el archivo o archivos correspondientes a una fuente si queremos incluirlos en nuestro sitio web, aunque esto no es necesario, ya que podemos utilizar los servidores de Google en su lugar. Para ello, utiliza el vínculo *Use this font*, dentro de la página particular de una de las fuentes.



Para ello, lo único que tendríamos que hacer es incluir la línea de código que aparece aquí al principio de nuestra página web. Debes hacerlo en la sección de cabecera.

Fíjate que la URL que se utiliza corresponde a un servidor de Google. Deberías actualizarla si has descargado las fuentes y dispones de ellas en tu servidor web. Después simplemente utilizarías dicha fuente como ya conoces.

De esta forma no es necesario utilizar la regla **@font-face**, ya que la hoja de estilo con esas reglas se genera dinámicamente en función del navegador del usuario.

En resumen debemos recordar que la **incrustación de fuentes** es posible ahora con CSS3.

Deberemos disponer de los archivos de fuente en los formatos adecuados y de los correspondientes derechos de distribución.

De esta forma, nos aseguramos de que el aspecto del texto que visualiza el navegador del usuario es el que nosotros esperamos aunque no tenga instalada la fuente utilizada durante el diseño de la página web.

3. MÚLTIPLES IMÁGENES DE FONDO

Otra novedad de la especificación CSS3 la encontramos cuando comprobamos que podemos aplicar más de una imagen para el fondo de un elemento.

Hasta ahora esto no era posible, por lo que el diseñador se veía en la obligación de insertar elementos **div** unos dentro de otros para conseguir el mismo efecto.

Cada una de las imágenes de fondo tiene un determinado índice **Z**, lo que determina qué imagen se muestra encima de la otra en el resultado final. Veamos un ejemplo.

```
<div id="container">
  <h1>Texto con título 1</h1>
  <h2>Texto con título 2</h2>
  <p>Lorem ipsum dolor sit amet, consectetur adipisicing
elit, sed do eiusmod tempor incididunt ut labore et
dolore magna aliqua. Ut enim ad minim veniam, quis
nostrud exercitation ullamco laboris nisi ut aliquip ex
ea commodo consequat. Duis aute irure dolor in
reprehenderit in voluptate velit esse cillum dolore eu
fugiat nulla pariatur. Excepteur sint occaecat cupidatat
non proident, sunt in culpa qui officia deserunt mollit
anim id est laborum.</p>
</div>
```

Una vez tenemos el contenido de la página en el interior de un elemento **div**, vamos a aplicar varias imágenes de fondo a este último.

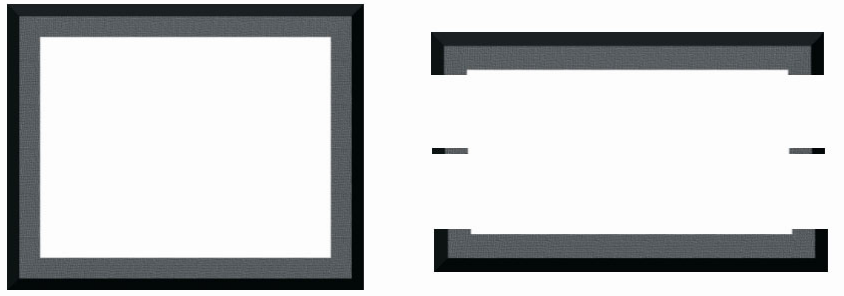
La idea es que el texto aparezca en el interior de un marco, que coincide con la imagen de la derecha.

Pero tenemos un problema y es que la imagen de fondo debería adaptarse al espacio ocupado por el texto. Esto solo sería posible conseguirlo si conocemos de antemano dicho texto y preparamos la imagen explícitamente para esas dimensiones, pero esto no siempre será así.

Pues para ese problema tenemos una solución: utilizar varias imágenes de fondo en lugar de una sola.

Así pues, podríamos tener una imagen con la parte superior del marco, otra con la parte inferior y una tercera con un trozo que se repita para la parte central del marco.

Esta es la idea: descomponer la imagen completa del marco en tres más pequeñas, repitiendo la central para adaptar el fondo al texto de la página web.



Como tenemos la oportunidad de establecer varias imágenes para el fondo del elemento **div**, podemos conseguir el marco completo a partir de estas tres imágenes más pequeñas:

```
#container {
   width: 960px;
   margin: 0 auto;
   background-color: gray;
   background-image: url("../images/top.jpg"),
                     url("../images/repeat.jpg"),
                     url("../images/bottom.jpg");
   background-repeat: no-repeat, repeat-y, no-repeat;
   background-position: top, 0, bottom;
}
```

Primero se ha establecido un color de fondo, que siempre es importante por si no se pueden cargar las imágenes de fondo, y después las imágenes de fondo.

En este caso vemos que se ha indicado más de una imagen de fondo en la propiedad **background-image**.

Cada una de ellas separada por una coma. El orden en que se establecen las imágenes en esta propiedad es el orden **Z**.

Así pues, la primera imagen es la que se situará encima de las otras dos, mientras que la segunda se situará encima de la tercera.

Adicionalmente podemos indicar otros detalles referentes a cada una de estas imágenes, como la posición, el tamaño, si deben repetirse, etc.

La propiedad **background-repeat** se establece para cada una de las imágenes de fondo. Así pues, la primera y tercera imagen no se tienen que repetir, mientras que la imagen correspondiente a la zona central del marco, la repetimos en vertical.

Si solo se indica uno de los dos valores x o y, entonces la otra coordenada queda centrada.

Por su parte, la propiedad **background-position** establece la posición de cada imagen respecto del borde superior izquierdo del elemento contenedor.

Podemos utilizar valores numéricos, porcentajes o palabras clave. Si utilizamos valores numéricos, podemos indicar la posición **x** e **y** separándolas por un espacio en blanco.

Las palabras clave **top**, **bottom**, **left** y **right** indican la posición relativa a dicho elemento contenedor.

Por lo tanto, lo que hemos indicado es que la primera imagen se incluya en la parte superior del elemento div (**top**); para la segunda imagen solo indicamos la posición x (**0**) y para la tercera imagen, indicamos que se sitúe en la parte inferior del elemento div (**bottom**).

Sin embargo, si probamos el resultado, nos daremos cuenta de que la imagen de la parte inferior no se muestra, quedando el marco abierto por debajo.

¿A qué se debe esto? Pues al orden que hemos utilizado al establecer las imágenes en la propiedad **background-image**.

Al indicar la imagen que se repite en la zona central antes que la imagen inferior, entonces realmente queda oculta esta última. Será fácil arreglarlo (en negrita se indican los cambios):

```
#container {
   width: 960px;
   margin: 0 auto;
   background-color: gray;
   background-image: url("../images/top.jpg"),
                     url("../images/bottom.jpg"),
                     url("../images/repeat.jpg");
   background-repeat: no-repeat, no-repeat, repeat-y;
   background-position: top, bottom, 0;
}
```

Ahora la imagen **bottom.jpg** se mostrará por encima de la imagen **repeat.jpg**, que es lo que necesitamos. También he cambiado el orden en las propiedades **background-repeat** y **background-position**.

En la figura siguiente puedes ver el resultado obtenido. Se ha utilizado la propiedad **padding** en los bloques de texto para centrarlos en el marco de la imagen de fondo.

Los navegadores que no soporten esta característica de CSS3 simplemente mostrarán el color de fondo establecido. El usuario no dispondrá del mejor resultado, pero seguirá siendo válida la página web.

4. TRANSICIONES

Continuamos con otra funcionalidad muy interesante que presenta la especificación CSS3 por primera vez como parte de las hojas de estilo en cascada. Se trata de las **transiciones**.

Las **transiciones** CSS permiten que el cambio que se efectúan en algunas propiedades de un elemento de la página se realice suavemente.

Se pueden aplicar transiciones con muchas propiedades CSS, pero habitualmente las utilizaremos para cambios de color, fondo o posición. Para ello, se utiliza el conjunto de propiedades que empieza con la palabra **transition**.

Con un ejemplo todo esto quedará más claro.

Vamos a modificar la regla que afecta a la etiqueta **h2** para realizar un cambio en el color y tamaño del texto de este tipo.

La idea es sencilla, ya que cambiaremos esas propiedades cuando el usuario sitúe el puntero del ratón por encima de dicho texto. Es decir, en su estado **:hover**.

```
h2:hover {
   color: red;
   font-size: 6em;
}
```

Si pruebas el resultado, comprobarás que al situar el puntero del ratón por encima de ese texto, cambia su color y tamaño, ya que se aplican las propiedades establecidas para el estado **hover**.

Además, también podrás comprobar que el tamaño de la imagen de fondo, es decir, del marco, se hace más grande, como habíamos anticipado en el capítulo anterior.

Bien, este cambio es bastante brusco. El aspecto cambia de un estado a otro presentando las propiedades tal como se han establecido.

¿Qué te parece si en lugar de este cambio tan brusco pudiéramos hacer que el cambio fuera más suave? Pues para eso sirven las **transiciones**.

Para utilizar transiciones entre estados se emplea el conjunto de propiedades siguiente:

- **transition-property**: se indica la propiedad o propiedades que se verán afectadas por la transición.
- **transition-duration**: determina el tiempo total de la transición. Se mide en segundos.
- **transition-timing-function**: establece cómo se realiza la transición.
- **transition-delay**: permite establecer un retraso inicial para la transición, también en segundos.

Estos son los valores que se pueden utilizar para la propiedad **transition-timing-function**. Determinan cuál es la velocidad durante la transición:

- **ease** (desaceleración gradual)
- **linear** (velocidad constante)
- **ease-in** (aumentando la velocidad)
- **ease-out** (disminuyendo la velocidad)
- **ease-in-out** (al principio aumenta la velocidad y después disminuye)
- **cubic-bezier** (permite definir una función personalizada)

Normalmente se utiliza **ease** o **linear**. Lo mejor es probar los distintos valores para comprobar las diferencias entre ellos, aunque si la transición dura poco tiempo, prácticamente no notarás ninguna diferencia.

Si utilizas la palabra clave all en lugar del nombre de una propiedad, entonces la transición afectará a cualquier propiedad que cambie entre un estado y otro del elemento.

Podemos emplear estas propiedades por separado o hacerlo con la propiedad abreviada **transition**. En este último caso, el orden que se utiliza es el mismo que se ha utilizado para describir estas propiedades. Además, si queremos que la transición afecte a más de una propiedad, podremos indicarlo repitiendo la regla y separándolas con comas:

```
h2 {
 font-family: "TangerineRegular";
 font-size: 3em;
 transition: color 1s ease 0.5s, font-size 1s ease 0.5s;
}
```

Con esta regla estamos indicando que se realice una transición del **color** y tamaño del texto durante **1** segundo, utilizando la función **ease** y con un retardo inicial de **medio** segundo.

Es importante darse cuenta de que estamos definiendo la transición en el estado inicial del título **h2** y no en la regla del estado **hover**.

Esto se hace así por comodidad, ya que esto permite definir la misma transición para otros estados. Por ejemplo, si cambiaran algunas de esas propiedades en el estado **focus** también se aplicaría la transición.

El problema es que esta es la forma estándar de definir una transición, pero los navegadores no la utilizan todavía, sino que emplean sus correspondientes prefijos.

Así que habrá que añadir la misma propiedad pero con prefijos:

```
h2 {
  font-family: "TangerineRegular";
  font-size: 3em;
  -webkit-transition: color 1s ease 0.5s, font-size 1s ease 0.5s;
  -moz-transition: color 1s ease 0.5s, font-size 1s ease 0.5s;
  -o-transition: color 1s ease 0.5s, font-size 1s ease 0.5s;
  transition: color 1s ease 0.5s, font-size 1s ease 0.5s;
}
```

Recuerda escribir la forma estándar; esto es, sin prefijos, en último lugar. Así, cuando los navegadores la implementen, será esa la que realmente se utilice.

Con la transición, el efecto es mucho más agradable que antes, pero si el navegador no es compatible, tampoco pasa nada, ya que el usuario simplemente obtendrá el resultado que hemos visto inicialmente, sin transición.

Con las transiciones podemos conseguir efectos espectaculares que hasta ahora solo eran posibles con tecnologías como Flash y JavaScript.

Recuerda que puedes utilizar esta característica con un gran número de propiedades, incluyendo algunas de las novedades de CSS3.

Lista de algunas de las propiedades que se pueden utilizar con las transiciones:

background-color	background-image (solo gradientes)
background-position	border-spacing
border-width	bottom
color	crop
font-size	font-weight
height	left
letter-spacing	line-height
margin	opacity
outline	padding
rigth	text-indent
text-shadow	top
transform	vertical-align
visibility	width
word-spacing	z-index
zoom	

5. TRANSFORMACIONES

Finalizaremos la lección estudiando otra novedad CSS3 que permite resultados espectaculares. Se trata de la posibilidad de **transformar** un elemento.

Cuando hablamos de transformaciones nos referimos a cambiar el tamaño, rotar, inclinar o mover un elemento. Para ello se utilizan los operadores **scale**, **rotate**, **skew** y **translate**, respectivamente, junto a la propiedad **transform**.

Estudia el código siguiente:

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
 charset=utf-8">
<title>Transformaciones</title>
<style type="text/css">
</style>
<link href="css/transform.css" rel="stylesheet"
type="text/css">
</head>

<body>
<ul id="galeria">
  <li><img src="images/menorca1.jpg" width="200"
       height="150" alt="Cala en Menorca"></li>
  <li><img src="images/menorca2.jpg" width="200"
       height="150" alt="Acantilado en Menorca"></li>
  <li><img src="images/menorca3.jpg" width="200"
  height="150" alt="Barcas de pescadores en Menorca"></li>
</ul>
</body>
</html>
```

En esta página hemos insertado tres imágenes en una lista sin numerar. Además, hemos aplicado algunas reglas CSS para mostrarlas como ahora verás en la hoja **transform.css**:

```
@charset "utf-8";
#galeria {
     margin-top: 200px;
     margin-left: 100px;
}
#galeria li {
     float: left;
     margin-top: 0px;
     margin-right: 35px;
     margin-bottom: 0px;
     margin-left: 15px;
     list-style-type: none;
}
```

```
#galeria li:hover {
     -webkit-transform: scale(2) rotate(-15deg);
     -moz-transform: scale(2) rotate(-15deg);
     -o-transform: scale(2) rotate(-15deg);
     -ms-transform: scale(2) rotate(-15deg);
     transform: scale(2) rotate(-15deg);
}
```

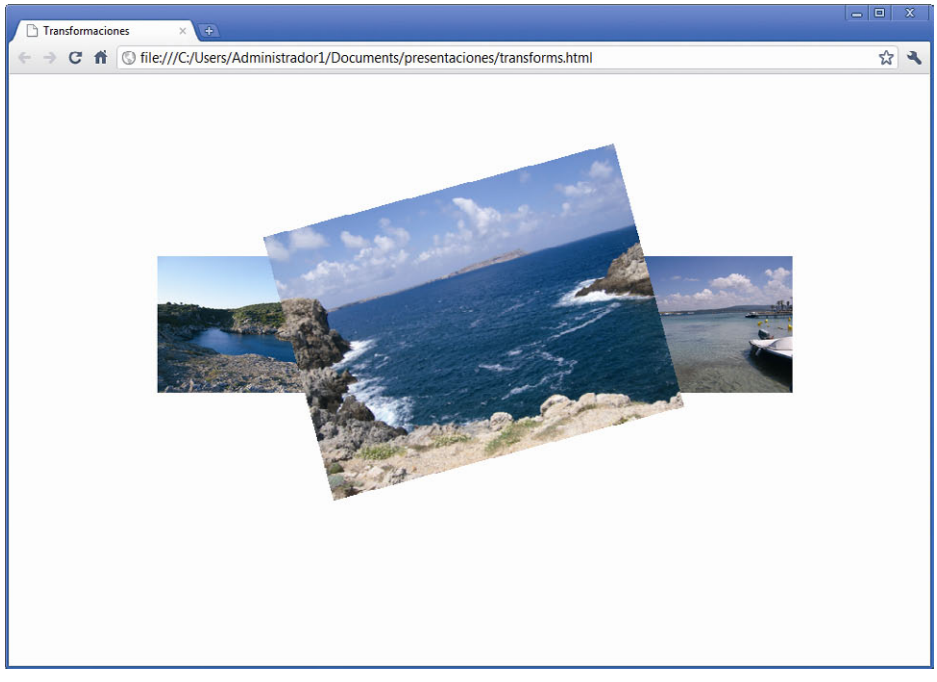
Pero lo que realmente me interesa es que estudies la regla correspondiente a **li:hover**, es decir, al estado en el que se sitúa el puntero del ratón por encima de uno de los elementos de la lista.

En ese estado se aplica una transformación que consiste primero en la aplicación del operador **scale** para cambiar el tamaño de la imagen y después de **rotate** para rotar la imagen.

Todos estos operadores no pueden utilizarse por sí mismos, sino siempre dentro de la propiedad **transform**.

Así pues, estamos indicando que primero se escale la imagen al **doble** del tamaño con el que se muestra en la página web y después que rote **15 grados** hacia la izquierda.

Como puedes ver, se requiere del uso de los prefijos para cada navegador, ya que la propiedad estándar "**transform**" todavía no está implementada por estos. Observa el resultado.



Al situar el puntero del ratón se realiza la transformación. En este caso la imagen original tiene realmente el tamaño que conseguimos con dicha transformación, por lo que no se ha deformado.

Ten en cuenta que estamos aplicando la transformación en el estado **hover** para que veas la diferencia, pero realmente esto no es necesario, ya que podrías aplicarla en estado normal, haciendo que las imágenes se mostraran como quieres desde el principio.

Además, podemos añadir una transición a la vez para que el efecto sea más agradable:

```
#galeria li {
     float: left;
     margin-top: 0px;
     margin-right: 35px;
     margin-bottom: 0px;
     margin-left: 15px;
     list-style-type: none;
     -webkit-transition: -webkit-transform 1s ease;
     -moz-transition: -moz-transform 1s ease;
     -o-transition: -o-transform 1s ease;
     transition: transform 1s ease;
}
```

Añadimos la transición indicando que la propiedad que se verá afectada es justamente **transform**. Como ambas propiedades requieren de sintaxis particular para cada navegador, el código es un poco extenso pero no es complicado.

La especificación **CSS3** está en continua evolución, no se trata de un trabajo finalizado.

Sin embargo, hemos podido comprobar que es posible utilizar características avanzadas sin ningún problema, ya que debe hacerse siempre en la capa de presentación de una página web.

De esta forma, si el usuario dispone de un navegador compatible con esas características, obtendrá una experiencia mucho más rica que si lo hace con un navegador antiguo.

Con un navegador antiguo, el usuario seguirá pudiendo acceder al contenido de la página web, aunque obteniendo una experiencia más limitada.