

Unidad 6

Más propiedades introducidas en CSS3

“Supongo que ustedes no están preparados para esta música, pero a sus hijos les encantará”.

Marty McFly

Son muchas las propiedades añadidas en CSS3 y muy irregular la implantación en los diferentes navegadores hasta el punto de que cada uno de ellos en muchos casos ha desarrollado por su cuenta algunas de las implementaciones añadiendo un prefijo distintivo que es -o- para ópera -moz- para firefox, -webkit- para chrome, safari y otros que usen el Webkit y -ms- para Internet Explorer. Veremos ejemplos de esto en varios de los apartados siguientes.

Gradientes y una nueva paleta de colores

CSS3 incorpora la posibilidad de realizar degradados. Existen dos tipos de gradientes: lineales y radiales. Veamos un ejemplo de cada uno:

```
background-image: linear-gradient(bottom, rgb(107,168,104) 13%,  
rgb(204,137,180) 57%, rgb(164,219,245) 83%);
```

```
background: radial-gradient(center, circle, white, #7DA5CD);
```

Desglosemos las particularidades de estas reglas. En primer lugar, cualquiera de ellas puede ir en la propiedad background o background-image. En segundo, el valor de la propiedad debería de empezar por

linear-gradient o radial-gradient según el tipo de gradiente deseado. En tercer lugar, ambas deben de llevar un número de parámetros separados por comas que serán un mínimo de tres en el caso del lineal y de cuatro en el caso del radial. Veamos estos parámetros.

En el caso del lineal, el primer parámetro define la dirección del degradado definiendo su inicio y pudiendo usar para ello una o dos de las palabras clave top, bottom, right o left. A continuación se enumeran los colores del degradado (dos, al menos pero pudiendo ser más). Cada color se define de cualquiera de las formas admitidas habitualmente en CSS (con un valor hexadecimal, una palabra distintiva o una función rgb) y pudiendo ir acompañado de un porcentaje que define la suavidad de la transición entre uno y otro color.

En el caso del radial, el primer parámetro también define la dirección pero además de las palabras claves anteriores tenemos otra, center, que en caso de usarse debería de ir sola. El segundo parámetro define la forma del degradado y puede ser circle o ellipse. A continuación irían los colores del degradado que siguen el mismo formato explicado anteriormente y deberían de ser al menos dos.

Se trata esta de una de las propiedades de que hablábamos cuyo estándar no se encuentra generalizado, así que si queremos que los principales navegadores lo entiendan debemos repetir la regla con los diferentes prefijos distintivos. Por ejemplo así:

```
p.degradadoradial{
  background: radial-gradient(center, circle, white, #7DA5CD);
  background: -o-radial-gradient(center, circle, white, #7DA5CD);
  background: -moz-radial-gradient(center, circle, white, #7DA5CD);
  background: -webkit-radial-gradient(center, circle, white, #7DA5CD);
  background: -ms-radial-gradient(center, circle, white, #7DA5CD);
}
```

En el caso de los gradientes lineales, tenemos una herramienta web muy útil que nos permite generar las reglas de forma automática aquí:

<http://gradients.glrzad.com/>

Fondos múltiples

CSS3 incorpora la posibilidad de incluir fondos múltiples en una misma caja sin necesidad de tener que superponer una caja por cada imagen. La posición, y características de cada imagen (repetición, etc.) se define también de forma independiente para cada una de ellas. Veamos un par de ejemplos. Ahora definir imágenes independientes para personalizar las esquinas de una caja es tan sencillo como esto:

```
p.caja2{
  padding: 60px;
  border: 1px solid black;
  background: url(esquina1.png) top left no-repeat,
             url(esquina2.png) top right no-repeat,
             url(esquina3.png) bottom right no-repeat,
             url(esquina4.png) bottom left no-repeat;
  background-color: white;
}
```

Y si queremos solapar dos imágenes de fondo transparente en la misma caja nos basta con hacer lo siguiente:

```
p.caja3{
  width: 300px;
  height: 400px;
  background: url(imagen1.gif) bottom no-repeat,
             url(imagen2.gif) bottom no-repeat;
}
```

Esquinas redondeadas y bordes decoradas

CSS3 incluye, por fin, la posibilidad de redondear bordes sin necesidad de complicadas maniobras como hemos tenido que hacer hasta el momento. La propiedad para hacer esto se llama border-radius. Su sintaxis es bien sencilla:

```
border-radius: 10px;
```

El valor indica el radio del arco de circunferencia que delimita el arco. Se puede usar cualquier unidad de medida válida en CSS (pulgadas, em,

etc.) y definir de forma independiente cada una de las cuatro esquinas de la caja con la siguiente sintaxis:

```
border-top-right-radius: 5px;  
border-bottom-right-radius: 2em;  
border-bottom-left-radius: 3em;  
border-top-left-radius: 10px;
```

Con el formato abreviado podemos poner dos valores diferentes (el primero para las esquina superior izquierda e inferior derecha y el otro para las otras dos) o cuatro valores (uno para cada esquina empezando por la superior izquierda y continuando en el sentido de las agujas del reloj):

```
border-radius: 10px 5px;  
border-radius: 10px 5px 20px 25px;
```

Podemos usar también esquinas irregulares especificando dos radios diferentes: uno para el eje X y otro para el eje Y. El formato cuando todas las esquinas son iguales es este:

```
border-radius: 10px / 50px;
```

Cuando especificamos las esquinas una a una no necesitamos la barra:

```
border-top-right-radius: 5px 10px;
```

Y cuando las esquinas son diferentes pero las especificamos en grupo:

```
border-radius: 10px 10px / 20px 30px;  
border-radius: 5px 10px 5px 10px / 10px 5px 10px 5px;
```

El redondeado de esquinas mediante esta propiedad se puede aplicar, además de a una caja, también directamente a una tabla o a una imagen:

```
table {  
  width: 200px;  
  background-color: rgb(185,205,225);  
  text-align: center;  
  margin: 30px;  
  border: 1px solid gray;
```

```

border-radius: 20px;
}

#imgredondo {
width: 600px;
height: 150px;
margin: 0 auto;
border-radius: 1em;
border: 1px solid gray;
background-image: url(header.jpg);
}

```

Sombras

CSS3 incorpora dos propiedades que sirven para dar sombras a textos y cajas que se llaman, respectivamente, `text-shadow` y `box-shadow`. El formato básico de ambas es similar. Veamos un par de ejemplos:

```

text-shadow: 15px 5px 10px orange;
box-shadow: -5px -10px 5em #000;

```

Los dos primeros parámetros son el desplazamiento vertical y horizontal de la sombra. Valores positivos desplazan esta hacia abajo y la derecha. Valores negativos hacia arriba y la izquierda. El tercer parámetro marca la intensidad y/o difuminación de la sombra (valores más pequeños definen una sombra más intensa y marcada, mientras que valores grandes crean una sombra muy difuminada y poco definida). No se puede usar un porcentaje como ninguno de estos tres parámetros. El cuarto parámetro es el color de la sombra y puede expresarse de cualquiera de las formas válida vistas en CSS.

`box-shadow` permite, además, dos parámetros opcionales: uno de ellos iría delante del color y define, mediante una medida, la distancia al objeto del foco que provoca la sombra, de forma que valores positivos acercan el foco provocando una sombra mayor y valores negativos alejan el foco provocando una sombra menor. El siguiente parámetro opcional es la palabra clave `inset` que iría al final de la declaración y provocaría una sombra interior a la caja.

```
box-shadow: -5px -10px 5em 15px #000 inset;
```

Opacidad

La propiedad `opacity` permite definir elementos parcial o totalmente transparentes. El valor de esta propiedad es un número decimal entre 0 y 1, de forma que 1 define la total opacidad (cuerpo sólido y no translucido) y el 0 la total transparencia.

```
opacity: .3;
```

Más control de la tipografía

En CSS3 se normaliza el uso de tipografías externas (como las de Google Fonts que vimos hace ya tiempo) de forma que podamos usar cualquier tipo de fuente disponible en la red en nuestras páginas. La incorporación y descarga de fuentes se hace mediante la instrucción `@font-face` que tiene el siguiente formato:

```
@font-face {  
    font-family: "fantasia";  
    url("fantasia.ttf");  
}
```

Si quisiéramos importar más de una fuente en nuestro proyecto necesitamos una instrucción `@font-face` por cada una de ellas:

```
@font-face { font-family: vera; src: url("VeraSeBd.ttf"); }  
@font-face { font-family: fantasia; src: url("fantasia.ttf"); }
```

Inexplicablemente, aunque las fuentes ttf se han popularizado gracias a Microsoft, estas no son soportadas por ninguna versión de Explorer. Microsoft prefiere usar las tipografías eot. Para que los usuarios con Explorer vean la fuentes necesitamos convertir las ttf a este formato y usar esta sintaxis que contempla ambos tipos:

```
@font-face {  
    font-family: fantasia;
```

```
src: url("fantasia.ttf");  
src: url("fantasia.eot");  
}
```

Existen diferentes programas capaces de convertir entre diferentes formatos. También puedes usar este recurso web:

<http://www.kirsle.net/wizards/ttf2eot.cgi>

La nueva propiedad `font-size-adjust` permite ajustar la altura de fuentes diferentes mediante el uso de un coeficiente de ajuste. Sólo está implementada en Firefox:

```
font-size-adjust: .448;
```

Aunque puede aproximarse “a ojo” por el método de prueba y de error, existe un recurso web que nos lista las fuentes disponibles en nuestro sistema y nos calcula los coeficientes de ajuste:

<http://www.brunildo.org/test/xheight.pl>

La propiedad `text-overflow` decide lo que se hará con un texto que no cabe completamente dentro de la caja en la que se encuentra. Tiene dos posibles valores: `clip` (lo corta sin más) o `ellipsis` (lo corta pero antes coloca tres puntos suspensivos).

La propiedad `font-stretch` permitirá ajustar el ancho de los caracteres, pero por el momento ningún navegador la implementa. Sus valores posibles son, de más ancha a más estrecha, `wider`, `narrower`, `ultra-condensed`, `extra-condensed`, `condensed`, `semi-condensed`, `normal`, `semi-expanded`, `expanded`, `extra-expanded` y `ultra-expanded`.

La propiedad `text-wrap` define como cortar las líneas de un texto y tampoco está implementada por el momento. El valor `normal`, por defecto, indica que sólo se pueden romper en los espacios y puntos. `none`

indica que no pueden romperse, `unrestricted` que pueden romperse entre dos caracteres cualesquiera y `suppress` indica que no sólo no pueden romperse sino que se suprimen incluso los saltos de línea manuales que hayamos colocado.

Columnas más fáciles

CSS3 dispone de siete propiedades que nos van a permitir mostrar el texto mediante columnas de forma muy fácil. Son `columns`, `column-gap`, `column-fill`, `column-span` y `column-rule`. Al igual que ocurre con las propiedades relacionadas con los gradientes, esta colección de propiedades precisa llevar el prefijo del navegador (`-moz-`, `-webkit-`) para funcionar en Firefox y Chrome pero no así en Opera. Explorer no las implementa aún. Los ejemplos a continuación se verán con la sintaxis necesaria para Mozilla salvo que se indique lo contrario.

`columns` define el número de columnas y el ancho mínimo esperado para cada una de ellas. Puede “separarse” en dos propiedades independientes: `column-width` y `column-count`. El funcionamiento de ambos parámetros está íntimamente relacionado pero la interpretación que hacen de ellos los diferentes navegadores no es unánime. Mozilla, por ejemplo, da preferencia al número de columnas mientras que Opera lo hace con el ancho de estas.

```
/* -moz-columns: 200px 3; */  
-moz-column-width: 200px;  
-moz-column-count: 3;
```

La propiedad `column-rule` define un separador entre las columnas y también puede dividirse en tres propiedades distintas: `column-rule-width`, `column-rule-style` y `column-rule-color`. Los estilos permitidos son los mismos que los de un borde.

```
/* -moz-column-rule: 1px solid gray; */  
-moz-column-rule-width: 1px;  
-moz-column-rule-style: solid;  
-moz-column-rule-color: gray;
```


column-gap define el espacio de separación entre columnas:

```
-moz-column-gap: 50px;
```

column-span nos permite marcar algún elemento para que no se propague entre diferentes columnas sino que esté completo en una de ellas. Imaginemos, por ejemplo, un texto de entradilla o un titular. Tiene dos valores válidos: 1 y auto según que deba limitarse a la primera columna o expandirse por las que precise. Sólo lo implementan Opera y Chrome:

```
column-span: 1;  
-webkit-column-span: 1
```

column-fill define como se “llenaran” las columnas. Por defecto su valor es balance, que hace que el navegador divida el contenido de la caja en tantas partes iguales como columnas para que estas sean lo más uniformes y parejas posible. Si su valor es auto, el navegador llenará primero la primera columna y no pasará a la segunda hasta que no quepa nada en ella y así sucesivamente. Esta propiedad aún no está implementada en ningún navegador.

Transformaciones y transiciones

La propiedad transform permite realizar diferentes transformaciones en 2D o 3D sobre cualquier elemento (textos, cajas...) de nuestra web. Se trata de una de las propiedades sobre las que aún se está trabajando (sobre todo en el terreno de 3D), implementada en todos los navegadores excepto el explorer, que hay que usar con los prefijos propios de cada navegador (-moz-, -webkit- y -o-) y para la que sólo veremos algunos ejemplos básicos.

Las transformaciones disponibles son rotación cambio de escala, deformación oblicua y desplazamiento.

La rotación se puede expresar en grados (deg), radianes (rad), aplicarse a los ejes X e Y por separado (rotateX y rotateY), a ambos por

igual (rotate), y especificar un número de grados positivo (en el sentido de las agujas del reloj) o negativo (sentido contrario). Un ejemplo:

```
transform: rotate(-20deg);  
-moz-transform: rotate(-20deg);  
-webkit-transform: rotate(-20deg);  
-o-transform: rotate(-20deg);
```

Los cambios de escala se especifican mediante un número entero o decimal que puede ser mayor que cero (ampliación). Menor (reducción) o negativo (reflexión). También podemos aplicar un factor de escala diferente al eje X y al Y (scaleX y scaleY). Otro ejemplo:

```
-moz-transform: scale(2.5);  
-webkit-transform: scale(2.5);  
-o-transform: scale(2.5);  
transform: scale(2.5);
```

La deformación oblicua usa el parámetro shew y nos permite especificar un ángulo en grados, radianes o gradientes (grad), positivos, negativos (para el sentido, igual que en la rotación) y especificar una deformación común o separada para cada eje:

```
-moz-transform: skew(-30deg);  
-webkit-transform: skew(-30deg);  
-o-transform: skew(-30deg);  
transform: skew(-30deg);
```

Por último, translate nos permite definir un desplazamiento indicando este en una medida concreta o en tanto por ciento, con valores positivos (abajo y hacia la derecha) o negativos (arriba y hacia la izquierda). En este caso el valor de la propiedad puede llevar uno o dos valores aplicando en el primer caso a ambos ejes y en el segundo distinguiendo entre ambos:

```
-moz-transform: translate(100px, -50px);  
-webkit-transform: translate(100px, -50px);  
-o-transform: translate(100px, -50px);  
transform: translate(100px, -50px);
```

Otras transformaciones en diferentes estados de normalización son: matrix, matrix3d, translate3d, translateZ, scale3d, scaleZ, rotate3d, rotateZ y perspective.

La propiedad transition, por último, nos va a permitir realizar pequeños efectos de animación sin necesidad de recurrir a javascript. Usamos para ello cuatro propiedades: transition-property, transition-duration, transition-timing-function y transition-delay (o transition, sin más). Tampoco está disponible para Explorer. Veamos primero un ejemplo y luego explicaremos los diferentes parámetros:

```
div.trancision {
    margin: 10px auto;
    padding: 10px;
    text-align: center;
    width: 200px;
    height: 100px;
    background-color: pink;
    border: 5px solid blue;

    -webkit-transition: border 5s ease-out ;
    -moz-transition: border 5s ease-out;
    -o-transition: all 5s ease-out;
    transition: border 5s ease-out;
}

div.trancision:hover{
    border-right: 200px solid #9cf;
}
```

En la primera regla especificamos que la trancisión afectará al borde (trancision-property), que durará 5 segundos (trancision-timing-function) y que será rápida al principio y más lenta al final (ease-out). En la segunda regla, que se activará al colocar el ratón encima, hacemos un cambio en el borde del elemento (tanto de tamaño como de color) y la trancisión realizará la transformación entre el estado original y el final con los datos especificados. Al retirar el ratón la animación se invertirá.

La propiedad puede ser el ancho (width), el alto (height), el color, una lista de propiedades diferentes separadas por coma o all (todas las propiedades que cambien entre uno y otro estado). Los tiempos y retrasos se especifican en segundos (s) o milisegundos (ms). Por último, el timing-function puede ser constante (linear), de lento a rápido (ease-in), de rápido a lento (ease o ease-out) o lento al principio y al final y más rápido en el centro (ease-in-out).

Se pueden especificar también diferentes parámetros para diferentes propiedades:

```
transition: width 2s ease, height 3s linear;
```