

1. Herencia

Una de las características principales de CSS es la herencia de los estilos definidos para los elementos. Cuando se establece el valor de una propiedad CSS en un elemento, sus elementos descendientes heredan de forma automática el valor de esa propiedad. Si se considera el siguiente ejemplo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo de herencia de estilos</title>
<style type="text/css">
body { color: blue; }
</style>
</head>

<body>
<h1>Titular de la página</h1>
<p>Un párrafo de texto no muy largo.</p>
</body>
</html>
```

En el ejemplo anterior, el selector `body` solamente establece el color de la letra para el elemento `<body>`. No obstante, la propiedad `color` es una de las que se heredan de forma automática, por lo que todos los elementos descendientes de `<body>` muestran ese mismo color de letra. Por tanto, establecer el color de la letra en el elemento `<body>` de la página implica cambiar el color de letra de todos los elementos de la página.

Aunque la herencia de estilos se aplica automáticamente, se puede anular su efecto estableciendo de forma explícita otro valor para la propiedad que se hereda, como se muestra en el siguiente ejemplo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo de herencia de estilos</title>
<style type="text/css">
body { font-family: Arial; color: black; }
  h1 { font-family: Verdana; }
  p { color: red; }
</style>
</head>

<body>
<h1>Titular de la página</h1>
<p>Un párrafo de texto no muy largo.</p>
</body>
</html>
```

En el ejemplo anterior, se establece en primer lugar el color y tipo de letra del elemento `<body>`, por lo que todos los elementos de la página se mostrarían con ese mismo color y tipo de letra. No obstante, las otras reglas CSS modifican alguno de los estilos heredados.

De esta forma, los elementos `<h1>` de la página se muestran con el tipo de letra Verdana establecido por el selector `h1` y se muestran de color negro que es el valor heredado del elemento `<body>`. Igualmente, los elementos `<p>` de la página se muestran del color rojo establecido por el selector `p` y con un tipo de letra Arial heredado del elemento `<body>`.

La herencia de estilos funciona automáticamente en la mayoría de las propiedades CSS. Más adelante se explica el mecanismo definido por CSS para forzar a que el valor de una propiedad se herede.

Por último, aunque la herencia automática de estilos puede parecer complicada, simplifica en gran medida la creación de hojas de estilos complejas. Como se ha visto en los ejemplos anteriores, si se quiere establecer por ejemplo la tipografía base de la página, simplemente se debe establecer en el elemento `<body>` de la página y el resto de elementos la heredarán de forma automática.

1.1 Colisiones de estilos

En las hojas de estilos complejas, es habitual que varias reglas CSS se apliquen a un mismo elemento HTML. El problema de estas reglas múltiples es que se pueden dar colisiones como la del siguiente ejemplo:

```
p { color: red; }  
p { color: blue; }
```

```
<p>...</p>
```

¿De qué color se muestra el párrafo anterior? CSS tiene un mecanismo de resolución de colisiones muy complejo y que tiene en cuenta el tipo de hoja de estilo que se trate (de navegador, de usuario o de diseñador), la importancia de cada regla y lo específico que sea el selector.

Hasta que no se expliquen más adelante los conceptos de tipo de hoja de estilo y la prioridad, el mecanismo simplificado que se puede aplicar es el siguiente:

1. Cuanto más específico sea un selector, más importancia tiene su regla asociada.
2. A igual *especificidad*, se considera la última regla indicada.

Como en el ejemplo anterior los dos selectores son idénticos, las dos reglas tienen la misma prioridad y prevalece la que se indicó en último lugar, por lo que el párrafo se muestra de color azul.

En el siguiente ejemplo, la regla CSS que prevalece se decide por lo específico que es cada selector:

```
p { color: red; }
p#especial{ color: green; }
* { color: blue; }

<p id="especial">...</p>
```

Al elemento <p> se le aplican las tres declaraciones. Como su origen y su importancia es la misma, decide la especificidad del selector. El selector * es el menos específico, ya que se refiere a "todos los elementos de la página". El selector p es poco específico porque se refiere a "todos los párrafos de la página". Por último, el selector p#especial sólo hace referencia a "el párrafo de la página cuyo atributo id sea igual a especial". Como el selector p#especial es el más específico, su declaración es la que se tiene en cuenta y por tanto el párrafo se muestra de color verde.

Ejercicio

A partir del código HTML proporcionado, añadir las reglas CSS necesarias para que la página resultante tenga el mismo aspecto que el de la siguiente imagen:

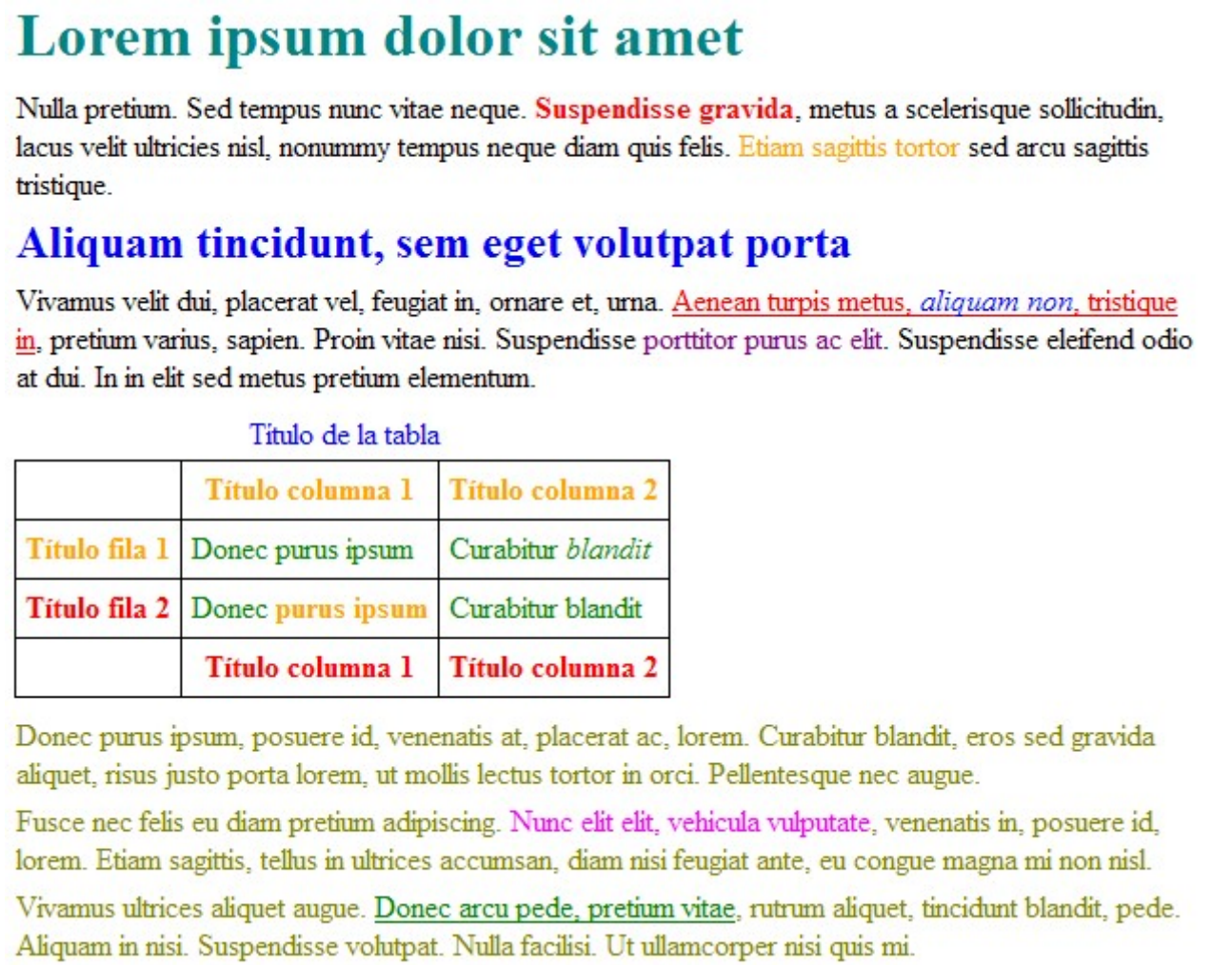


Figura 15.1. Aspecto final de la página

A continuación se muestra el código HTML de la página sin estilos:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Ejercicio de selectores</title>
</head>

<body>
<h1 id="titulo">Loremipsum dolor sit amet</h1>

<p>Nullapretium. Sed tempus nunc vitae neque.
<strong>Suspendissegravida</strong>, metus a scelerisquesollicitudin,
lacus velit
ultriciesnisl, nonummytempusnequediamquisfelis.
<spanclass="destacado">Etiamsagittis tortor</span> sed
arcusagittistristique.</p>

<h2 id="subtitulo">Aliquamtincidunt, sem egetvolutpat porta</h2>

<p>Vivamusvelit dui, placeratvel, feugiat in, ornare et, urna. <a
href="#">Aeneanturpismetus, <em>aliquam non</em>, tristique in</a>,
pretiumvarius, sapien. Proin vitae nisi. Suspendisse<span
class="especial">porttitorpurus ac elit</span>.
Suspendisseeleifendodio at dui. In in
elitsedmetuspretiumelementum.</p>

<tablesummary="Descripción de la tabla y su contenido">
<caption>Título de la tabla</caption>
<thead>
<tr>
<th scope="col"></th>
<th scope="col" class="especial">Títulocolumna 1</th>
<th scope="col" class="especial">Títulocolumna 2</th>
</tr>
</thead>

<tfoot>
<tr>
<th scope="col"></th>
<th scope="col">Títulocolumna 1</th>
<th scope="col">Títulocolumna 2</th>
</tr>
</tfoot>

<tbody>
<tr>
<th scope="row" class="especial">Títulofila 1</th>
<td>Donecpurusipsum</td>
<td>Curabitur<em>blandit</em></td>
</tr>
<tr>
<th scope="row">Títulofila 2</th>
<td>Donec<strong>purusipsum</strong></td>
<td>Curabiturblandit</td>
</tr>
```

```

</tbody>
</table>

<div id="adicional">
<p>Donecpurusipsum, posuere id, venenatis at, <span>placerat ac,
lorem</span>. Curabiturblandit, erossgedgravidaaliquet, risusjusto
portalorem, utmollislectustortor in orci. Pellentesquenecaugue.</p>

<p>Fuscenecfeliseudiampretiumadipiscing. <span id="especial">Nunc
elitelit, vehicula vulputate</span>, venenatis in,
posuere id, lorem. Etiamsagittis, tellus in ultrices accumsan,
diamnisifeugiat ante, eucongue magna mi non nisl.</p>

<p>Vivamusultricesaliquetaugue. <a href="#">Donecarcupede, pretium
vitae</a>, rutrumaliquet, tinciduntblandit, pede.
Aliquam in nisi. Suspendissevolutpat. Nullafacilisi. Ut
ullamcorpernisiquismi.</p>
</div>

</body>
</html>

```

Aunque la propiedad que modifica el color del texto se explica detalladamente en los próximos capítulos, en este ejercicio solamente es preciso conocer que la propiedad se llama color y que como valor se puede indicar directamente el nombre del color.

Los nombres de los colores también están estandarizados y se corresponden con el nombre en inglés de cada color. En este ejercicio, se deben utilizar los colores: teal, red, blue, orange, purple, olive, fuchsia y green.

3. Unidades de medida y colores

Muchas de las propiedades de CSS que se ven en los próximos capítulos permiten indicar medidas y colores en sus valores. Además, CSS es tan flexible que permite indicar las medidas y colores de muchas formas diferentes.

Por este motivo, se presentan a continuación todas las alternativas disponibles en CSS para indicar las medidas y los colores. En los siguientes capítulos, cuando una propiedad pueda tomar como valor una medida o un color, no se volverán a explicar todas estas alternativas.

3.1 Unidades de medida

Las medidas en CSS se emplean, entre otras, para definir la altura, anchura y márgenes de los elementos y para establecer el tamaño de letra del texto. Todas las medidas se indican como un valor numérico entero o decimal seguido de una unidad de medida (sin ningún espacio en blanco entre el número y la unidad de medida).

CSS divide las unidades de medida en dos grupos: **absolutas y relativas**. Las medidas relativas definen su valor en relación con otra medida, por lo que para obtener su valor real, se debe realizar alguna operación con el valor indicado. Las unidades absolutas

establecen de forma completa el valor de una medida, por lo que su valor real es directamente el valor indicado.

Si el valor es 0, la unidad de medida es opcional. Si el valor es distinto a 0 y no se indica ninguna unidad, la medida se ignora completamente, lo que suele ser una fuente habitual de errores para los diseñadores que empiezan con CSS. Algunas propiedades permiten indicar medidas negativas, aunque habitualmente sus valores son positivos.

3.1.1. Unidades relativas

Las unidades relativas son más flexibles que las unidades absolutas porque se adaptan más fácilmente a los diferentes medios. A continuación se muestra la lista de unidades de medida relativas y la referencia que se toma para determinar su valor real:

- em, (no confundir con la etiqueta de HTML) relativa respecto del tamaño de letra empleado. Aunque no es una definición exacta, el valor de 1em se puede aproximar por la anchura de la letra M ("*eme mayúscula*") del tipo y tamaño de letra que se esté utilizando
- ex, relativa respecto de la altura de la letra x ("*equis minúscula*") del tipo y tamaño de letra que se esté utilizando
- px, (píxel) relativa respecto de la resolución de la pantalla del usuario

Las unidades em y ex no han sido creadas por CSS, sino que llevan décadas utilizándose en el campo de la tipografía. La unidad em hace referencia al tamaño en puntos de la letra que se está utilizando. Si se utiliza una tipografía de 12 puntos, 1em equivale a 12 puntos. El valor de 1ex se puede aproximar por 0.5 em.

En el siguiente ejemplo, se indica que el tamaño de letra del texto de la página debe ser el 90% del tamaño por defecto (que depende de cada navegador, aunque es muy similar entre ellos):

```
body{font-size: 0.9em;}
```

Como em es una unidad relativa, el valor 0.9 indicado sólo tiene sentido cuando se tiene en consideración su referencia. Para la unidad em, la referencia es el tamaño de letra por defecto del sistema (ordenador, dispositivo móvil, etc.) del usuario.

Por lo tanto, 0.9em significa que se debe multiplicar 0.9 por el tamaño de letra por defecto, lo que en la práctica significa que la medida indicada es igual al 90% del tamaño de letra por defecto. Si este tamaño por defecto es 12pt, el valor 0.9em sería igual a $0.9 \times 12\text{pt} = 10.8\text{pt}$.

Cuando el valor decimal de una medida es inferior a 1, se puede omitir el 0 de la izquierda, por lo que el código anterior es equivalente al código siguiente:

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

El siguiente ejemplo muestra el uso de la unidad em para establecer el tamaño de la letra de diferentes párrafos:

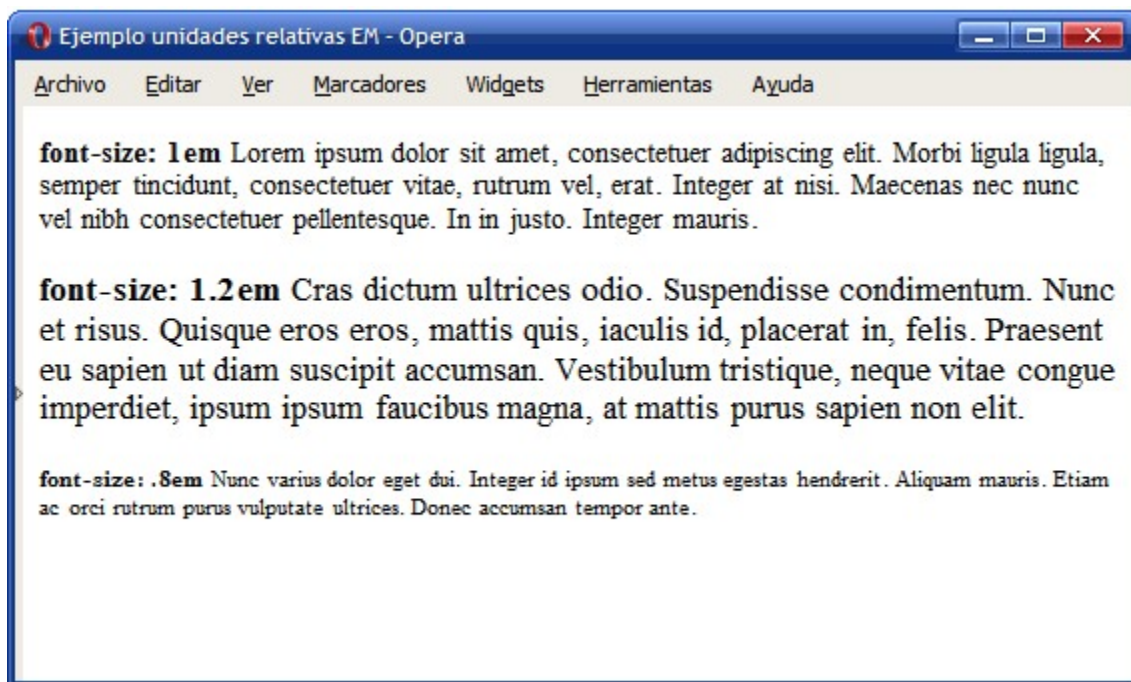


Figura 3.1. Ejemplo de tamaño de letra definido con la unidad relativa em

El primer párrafo muestra la letra con un tamaño de 1em, es decir, el tamaño por defecto en el navegador del usuario. El segundo párrafo ha establecido el tamaño de letra en 1.2em, es decir, un 20% más grande que el tamaño por defecto. Por último, el tercer párrafo ha indicado un tamaño de .8em, es decir, un 20% inferior al tamaño por defecto.

Cuando se estudian por primera vez, las unidades relativas parecen demasiado complicadas. Al fin y al cabo, siempre se debe tomar la referencia de la unidad para obtener su valor real. Sin embargo, sus ventajas son mucho mayores que sus inconvenientes.

El ejemplo anterior establece el tamaño de la letra mediante los valores 1em, 1.2em y .8em. En otras palabras, el código anterior está estableciendo los tamaños de letra a "normal", "grande" y "pequeño" respectivamente. Independientemente del tamaño de letra por defecto del dispositivo del usuario, el primer párrafo se verá con un tamaño de letra "normal" (1em), el segundo párrafo se verá más "grande" de lo normal (1.2em) y el último párrafo se verá "pequeño" (.8em).

De esta forma, si el usuario tiene problemas de visión y aumenta el tamaño de letra en su navegador, las proporciones se mantendrán. Si el tamaño de letra por defecto es 12, el primer párrafo se verá con tamaño 12, pero si el usuario aumenta el tamaño de letra por defecto a 20, el primer párrafo se verá con tamaño 20. Como se ve, las unidades relativas permiten mantener las proporciones del diseño independientemente del tamaño de letra por defecto del navegador del usuario.

Como se verá más adelante, la propiedad font-size permite establecer el tamaño de letra del texto de un elemento. En este caso, la referencia para el valor de font-size de

un elemento siempre es el tamaño de letra de su elemento padre (es decir, del elemento en el que se encuentra). Si el elemento no se encuentra dentro de ningún otro elemento, la referencia es el tamaño de letra del elemento <body>. Si no se indica de forma explícita un valor para el tamaño de letra del elemento <body>, la referencia es el tamaño de letra por defecto del navegador.

Siguiendo esta norma, si en el ejemplo anterior se modifica el tamaño de letra del elemento <body> (que es el elemento padre de los tres párrafos) y se le asigna un valor de 0.8em, el aspecto que muestran los párrafos en el navegador es el siguiente:

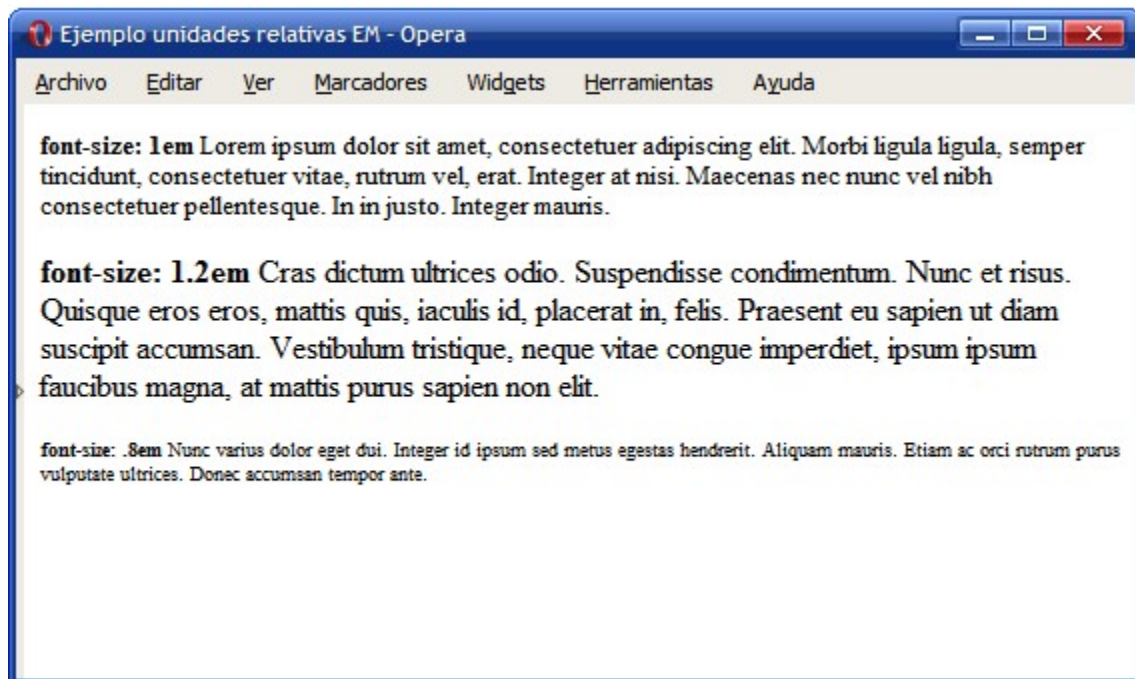


Figura 3.2. Ejemplo de tamaño de letra definido con la unidad relativa em

Al haber reducido el tamaño de letra que era la referencia del tamaño de letra de los tres párrafos, su texto se ve con una letra más pequeña, aunque manteniendo las proporciones: el primer párrafo se ve con un tamaño de letra normal, el segundo se ve con un tamaño grande y el tercero se visualiza con un tamaño de letra más pequeño de lo normal.

El funcionamiento de la unidad ex es idéntico a em, salvo que en este caso, la referencia es la altura de la letra x minúscula.

Aunque puede resultar paradójico, las medidas indicadas en píxel también se consideran relativas, ya que el aspecto de los elementos dependerá de la resolución del dispositivo en el que se visualiza el documento HTML. Cuando se visualiza un documento en un dispositivo de alta resolución (por ejemplo una impresora de 1200 dpi) se reescalan los píxel del documento y cada uno de los píxel originales se visualizan como un conjunto de píxel del dispositivo de alta resolución.

Las distintas unidades se pueden mezclar entre los diferentes elementos de una misma página, como en el siguiente ejemplo:

```
body{font-size: 10px; }  
h1 {font-size: 2.5em; }
```

En primer lugar, se establece un tamaño de letra base de 10 píxel para toda la página. A continuación, se asigna un tamaño de 2.5em al elemento <h1>, por lo que su tamaño de letra real será de $2.5 \times 10\text{px} = 25\text{px}$.

Como se vio en los capítulos anteriores, muchas propiedades CSS se heredan desde los elementos padre hasta los hijos. Así por ejemplo, si se establece el tamaño de letra al elemento <body>, todos los elementos de la página tendrán el mismo tamaño de letra, salvo que indiquen otro valor.

3.1.2. Unidades absolutas

Las unidades absolutas definen las medidas de forma completa, ya que sus valores reales no se calculan a partir de otro valor de referencia, sino que son directamente los valores indicados. A continuación se muestra la lista completa de unidades absolutas definidas por CSS y su significado:

- in, del inglés "*inches*", pulgadas (1 pulgada son 2.54 centímetros)
- cm, centímetros
- mm, milímetros
- pt, puntos (1 punto equivale a 1 pulgada/72, es decir, unos 0.35 milímetros)
- pc, picas (1 pica equivale a 12 puntos, es decir, unos 4.23 milímetros)

A continuación se muestran ejemplos de utilización de unidades absolutas:

```
body{margin: 0.5in; }  
h1 {line-height: 2cm; }  
p{word-spacing: 4mm; }  
a {font-size: 12pt}  
span{font-size: 1pc }
```

Su uso es idéntico al de las unidades relativas, siendo su única diferencia que los valores indicados son directamente los valores que se utilizan, sin necesidad de calcular los valores reales en función de otras referencias.

De todas las unidades absolutas, la única que se utiliza con cierta frecuencia es la de los puntos (pt). El motivo es que se trata de la unidad preferida para indicar el tamaño de letra del texto para los documentos que se van a imprimir, es decir, para el medio print de CSS (como se verá más adelante).

3.1.3. Porcentajes

CSS define otra unidad de medida relativa basada en los porcentajes. Un porcentaje está formado por un valor numérico seguido del símbolo % y siempre está referenciado

a otra medida. Cada una de las propiedades de CSS que permiten indicar como valor un porcentaje, define el valor al que hace referencia ese porcentaje.

Los porcentajes se pueden utilizar por ejemplo para establecer el valor del tamaño de letra de los elementos:

```
body{font-size: 1em; }  
h1 {font-size: 200%; }  
h2 {font-size: 150%; }
```

Los tamaños establecidos para los elementos <h1> y <h2> mediante las reglas anteriores, son equivalentes a 2em y 1.5em respectivamente, por lo que es más habitual definirlos mediante em.

Los porcentajes también se utilizan para establecer la anchura de los elementos:

```
div#contenido{width: 600px; }  
div.principal{width: 80%; }
```

```
<div id="contenido">  
<div class="principal">  
...  
</div>  
</div>
```

En el ejemplo anterior, la referencia del valor 80% es la anchura de su elemento padre. Por tanto, el elemento <div> cuyo atributo class vale principal tiene una anchura de 80% x 600px = 480px.

3.1.4. Recomendaciones

En general, se recomienda el uso de unidades **relativas siempre que sea posible**, ya que mejora la accesibilidad de la página y permite que los documentos se adapten fácilmente a cualquier medio y dispositivo.

El documento *"Recomendaciones sobre técnicas CSS para la mejora de la accesibilidad de los contenidos HTML"* (<http://www.w3.org/TR/WCAG10-CSS-TECHS/>) elaborado por el organismo W3C, recomienda el uso de la unidad em para indicar el tamaño del texto y para todas las medidas que sean posibles.

Normalmente se utilizan píxel y porcentajes para definir el layout del documento (básicamente, la anchura de las columnas y elementos de las páginas) y em y porcentajes para el tamaño de letra de los textos.

Por otra parte, uno de los problemas habituales cuando se utilizan unidades relativas es el problema de *"el texto cada vez se ve más pequeño"* o *"el texto cada vez se ve más grande"*. El siguiente ejemplo muestra el primer caso:

```
div{font-size: 0.9em; }
```

```
<div>
<p>Texto 1</p>
<div>
<p>Texto 2</p>
<div>
<p>Texto 3</p>
</div>
</div>
</div>
```

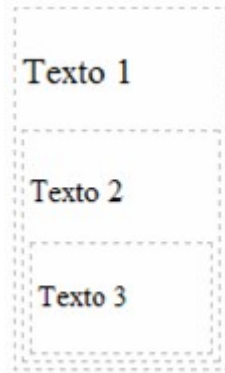


Figura 3.3. El texto cada vez se ve más pequeño

En el ejemplo anterior, el tamaño del texto de todos los elementos `<div>` se define mediante la medida relativa 0.9em. Como se trata de una medida relativa, su valor real se calcula a partir del tamaño de letra de su elemento padre. De esta forma, el tamaño de letra del primer `<div>` es igual a 0.9em respecto del tamaño de letra por defecto.

En el segundo elemento `<div>`, el tamaño de letra es 0.9em respecto al tamaño de letra del primer `<div>`, es decir, $0.9em \times 0.9em = 0.81em$ respecto del tamaño de letra por defecto, por lo que su letra se ve más pequeña que la del primer `<div>`.

Por último, el tamaño de letra del tercer `<div>` será igual a 0.9em respecto al tamaño de la letra del segundo elemento `<div>`, es decir, $0.9em \times 0.9em \times 0.9em = 0.729em$ respecto del tamaño de letra por defecto. De esta forma, el tamaño de letra de este tercer `<div>` es mucho más pequeño que el del primer `<div>`. Si se anidan varios elementos `<div>`, la letra se hará tan pequeña que no será posible leerla.

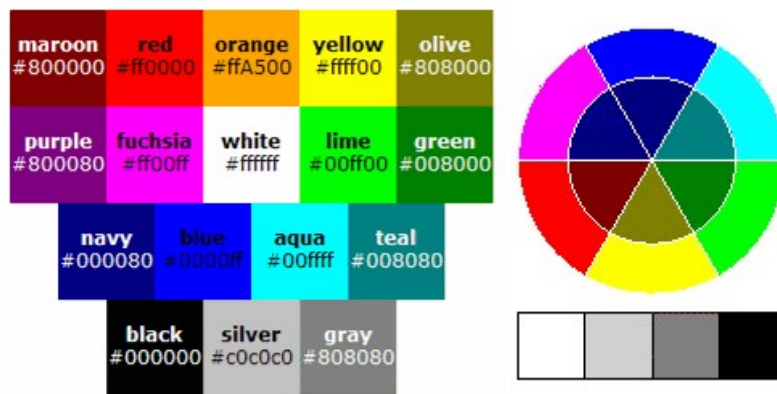
En el caso de que se indique un valor mayor que 1 para la medida relativa, el comportamiento es muy similar al descrito anteriormente, salvo que en este caso el tamaño de letra cada vez es mayor.

3.2 Colores

Los colores en CSS se pueden indicar de **cinco formas diferentes**: palabras clave, colores del sistema, RGB hexadecimal, RGB numérico y RGB porcentual. Aunque el método **más habitual es el del RGB hexadecimal**, a continuación se muestran todas las alternativas que ofrece CSS.





























3.2.1. Palabras clave

En la recomendación [HTML 3.2](#) (aprobada el 14 de enero de 1997) se incluyeron dieciséis nombres de colores: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white y yellow. **Estos colores son los colores de la paleta VGA de Windows**. La tabla e ilustración siguientes muestran estos 16 colores,






























Aunque es una forma muy sencilla de referirse a los colores básicos, este método prácticamente no se utiliza en las hojas de estilos de los sitios web reales, ya que se trata de una gama de colores muy limitada.

Además de la lista básica, los navegadores modernos soportan muchos otros nombres de colores.

Color	Nombre	Código RGB	Color	Nombre	Código RGB
	AliceBlue	#F0F8FF		LightPink	#FFB6C1
	AntiqueWhite	#FAEBD7		LightSalmon	#FFA07A
	Aqua	#00FFFF		LightSeaGreen	#20B2AA
	Aquamarine	#7FFFD4		LightSkyBlue	#87CEFA
	Azure	#F0FFFF		LightSlateGray	#778899
	Beige	#F5F5DC		LightSlateGrey	#778899
	Bisque	#FFE4C4		LightSteelBlue	#B0C4DE
	Black	#000000		LightYellow	#FFFFE0
	BlanchedAlmond	#FFEBCD		Lime	#00FF00
	Blue	#0000FF		LimeGreen	#32CD32
	BlueViolet	#8A2BE2		Linen	#FAF0E6
	Brown	#A52A2A		Magenta	#FF00FF
	BurlyWood	#DEB887		Maroon	#800000
	CadetBlue	#5F9EA0		MediumAquamarine	#66CDAA

Color	Nombre	Código RGB	Color	Nombre	Código RGB
	Chartreuse	#7FFF00		MediumBlue	#0000CD
	Chocolate	#D2691E		MediumOrchid	#BA55D3
	Coral	#FF7F50		MediumPurple	#9370DB
	CornflowerBlue	#6495ED		MediumSeaGreen	#3CB371
	Cornsilk	#FFF8DC		MediumSlateBlue	#7B68EE
	Crimson	#DC143C		MediumSpringGreen	#00FA9A
	Cyan	#00FFFF		MediumTurquoise	#48D1CC
	DarkBlue	#00008B		MediumVioletRed	#C71585
	DarkCyan	#008B8B		MidnightBlue	#191970
	DarkGoldenrod	#B8860B		MintCream	#F5FFFA
	DarkGray	#A9A9A9		MistyRose	#FFE4E1
	DarkGrey	#A9A9A9		Moccasin	#FFE4B5
	DarkGreen	#006400		NavajoWhite	#FFDEAD
	DarkKhaki	#BDB76B		Navy	#000080
	DarkMagenta	#8B008B		OldLace	#FDF5E6
	DarkOliveGreen	#556B2F		Olive	#808000
	DarkOrange	#FF8C00		OliveDrab	#6B8E23
	DarkOrchid	#9932CC		Orange	#FFA500
	DarkRed	#8B0000		OrangeRed	#FF4500
	DarkSalmon	#E9967A		Orchid	#DA70D6
	DarkSeaGreen	#8FBC8F		PaleGoldenrod	#EEE8AA
	DarkSlateBlue	#483D8B		PaleGreen	#98FB98
	DarkSlateGray	#2F4F4F		PaleTurquoise	#AFEEEE
	DarkSlateGrey	#2F4F4F		PaleVioletRed	#DB7093
	DarkTurquoise	#00CED1		PapayaWhip	#FFEFD5
	DarkViolet	#9400D3		PeachPuff	#FFDAB9
	DeepPink	#FF1493		Peru	#CD853F
	DeepSkyBlue	#00BFFF		Pink	#FFC0CB
	DimGray	#696969		Plum	#DDA0DD
	DimGrey	#696969		PowderBlue	#B0E0E6
	DodgerBlue	#1E90FF		Purple	#800080
	FireBrick	#B22222		Red	#FF0000
	FloralWhite	#FFFAF0		RosyBrown	#BC8F8F
	ForestGreen	#228B22		RoyalBlue	#4169E1
	Fuchsia	#FF00FF		SaddleBrown	#8B4513
	Gainsboro	#DCDCDC		Salmon	#FA8072
	GhostWhite	#F8F8FF		SandyBrown	#F4A460
	Gold	#FFD700		SeaGreen	#2E8B57
	Goldenrod	#DAA520		Seashell	#FFF5EE
	Gray	#808080		Sienna	#A0522D
	Grey	#808080		Silver	#C0C0C0
	Green	#008000		SkyBlue	#87CEEB
	GreenYellow	#ADFF2F		SlateBlue	#6A5ACD
	Honeydew	#F0FFF0		SlateGray	#708090
	HotPink	#FF69B4		SlateGrey	#708090
	IndianRed	#CD5C5C		Snow	#FFFAFA

Color	Nombre	Código RGB	Color	Nombre	Código RGB
	Indigo	#4B0082		SpringGreen	#00FF7F
	Ivory	#FFFFF0		SteelBlue	#4682B4
	Khaki	#F0E68C		Tan	#D2B48C
	Lavender	#E6E6FA		Teal	#008080
	LavenderBlush	#FFF0F5		Thistle	#D8BFD8
	LawnGreen	#7CFC00		Tomato	#FF6347
	LemonChiffon	#FFFACD		Turquoise	#40E0D0
	LightBlue	#ADD8E6		Violet	#EE82EE
	LightCoral	#F08080		Wheat	#F5DEB3
	LightCyan	#E0FFFF		White	#FFFFFF
	LightGoldenrodYellow	#FAFAD2		WhiteSmoke	#F5F5F5
	LightGray	#D3D3D3		Yellow	#FFFF00
	LightGreen	#90EE90		YellowGreen	#9ACD32
	LightGrey	#D3D3D3			

3.2.2. RGB decimal

En el campo del diseño gráfico, se han definido varios modelos para hacer referencia a los colores. Los dos modelos más conocidos son RGB y CMYK. Simplificando su explicación, el modelo RGB consiste en definir un color indicando la cantidad de color rojo, verde y azul que se debe mezclar para obtener ese color. Técnicamente, el modelo RGB es un modelo de tipo "aditivo", ya que los colores se obtienen sumando sus componentes.

Por lo tanto, en el modelo RGB un color se define indicando sus tres componentes R (rojo), G (verde) y B (azul). Cada una de las componentes puede tomar un valor entre cero y un valor máximo. De esta forma, el color rojo puro en RGB se crea mediante el máximo valor de la componente R y un valor de 0 para las componentes G y B.

Si todas las componentes valen 0, el color creado es el negro y si todas las componentes toman su valor máximo, el color obtenido es el blanco. En CSS, las componentes de los colores definidos mediante RGB decimal pueden tomar valores entre 0 y 255. El siguiente ejemplo establece el color del texto de un párrafo:

```
p {color: rgb(71, 98, 176); }
```

La sintaxis que se utiliza para indicar los colores es rgb() y entre paréntesis se indican las tres componentes RGB, en ese mismo orden y separadas por comas. El color del ejemplo anterior se obtendría mezclando las componentes R=71, G=98, B=176, que se corresponde con un color azul claro.

Si se indica un valor menor que 0 para una componente, automáticamente se transforma su valor en 0. Igualmente, si se indica un valor mayor que 255, se transforma automáticamente su valor a 255.

3.2.3. RGB porcentual

Otra forma de indicar las componentes RGB de un color es mediante un porcentaje. El funcionamiento y la sintaxis de este método es el mismo que el del RGB decimal. La única diferencia en este caso es que el valor de las componentes RGB puede tomar valores entre 0% y 100%. El mismo color del ejemplo anterior se puede representar de forma porcentual:

```
p {color: rgb(27%, 38%, 69%); }
```

Al igual que sucede con el RGB decimal, si se indica un valor inferior a 0%, se transforma automáticamente en 0% y si se indica un valor superior a 100%, se trunca su valor a 100%.

3.2.4. RGB hexadecimal

Aunque es el método más complicado para indicar los colores, se trata del **método más utilizado** con mucha diferencia. De hecho, prácticamente todos los sitios web reales utilizan exclusivamente este método.

Definir un color en CSS con el método RGB hexadecimal requiere realizar los siguientes pasos:

1. Determinar las componentes RGB decimales del color original, por ejemplo: R = 71, G = 98, B = 176
2. Transformar el valor decimal de cada componente al sistema numérico hexadecimal. Se trata de una operación exclusivamente matemática, por lo que puedes utilizar una calculadora. En el ejemplo anterior, el valor hexadecimal de cada componente es: R = 47, G = 62, B = B0
3. Para obtener el color completo en formato RGB hexadecimal, se concatenan los valores hexadecimales de las componentes RGB en ese orden y se les añade el prefijo #. De esta forma, el color del ejemplo anterior es #4762B0 en formato RGB hexadecimal.

Siguiendo el mismo ejemplo de las secciones anteriores, el color del párrafo se indica de la siguiente forma utilizando el formato RGB hexadecimal:

```
p {color: #4762B0; }
```

Recuerda que aunque es el método más complicado para definir un color, se trata del método que utilizan la inmensa mayoría de sitios web, por lo que es imprescindible dominarlo. Afortunadamente, todos los programas de diseño gráfico convierten de forma automática los valores RGB decimales a sus valores RGB hexadecimales, por lo que no tienes que hacer ninguna operación matemática:

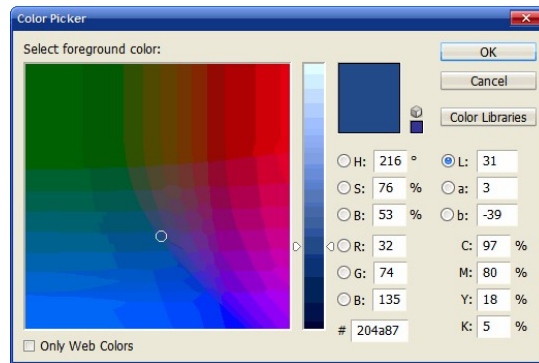


Figura 3.5. Herramienta de color de Photoshop para definir los colores según los modelos RGB, CMYK, Lab, HSB y RGB hexadecimal

Una de las ventajas del formato RGB hexadecimal es que se pueden comprimir sus valores cuando todas sus componentes son iguales dos a dos:

#AAA = #AAAAAA
 #FFF = #FFFFFF
 #A0F = #AA00FF
 #369 = #336699

En el siguiente ejemplo se establece el color de fondo de la página a blanco, el color del texto a negro y el color de la letra de los titulares se define de color rojo:

```
body{background-color: #FFF; color: #000; }
h1, h2, h3, h4, h5, h6 {color: #C00; }
```

Las letras que forman parte del color en formato RGB hexadecimal se pueden escribir en mayúsculas o minúsculas indistintamente. No obstante, se recomienda escribirlas siempre en mayúsculas o siempre en minúsculas para que la hoja de estilos resultante sea más limpia y homogénea.

3.2.5. Colores del sistema

Los colores del sistema son similares a los colores indicados mediante su nombre, pero en este caso hacen referencia al color que muestran algunos elementos del sistema operativo del usuario.

Color	Nombre	Concepto
	ActiveBorder	Borde de la ventana activa
	ActiveCaption	Título de la ventana activa
	AppWorkspace	Color de fondo del interfaz de múltiples documentos
	Background	Color de fondo del escritorio
	ButtonFace	Color frontal de los elementos 3D
	ButtonHighlight	Sombra oscura de los elementos 3D (bordes opuestos a la fuente de luz)
	ButtonShadow	Sombra de los elementos 3D
	ButtonText	Texto de los botones
	CaptionText	Texto en título, caja de tamaño y caja de flecha de desplazamiento
	GrayText	Texto en gris (desactivado). Este color es #000000 si no se puede mostrar un color gris sólido
	Highlight	Elemento(s) seleccionado(s) en un control
	HighlightText	Texto de (de los) elemento(s) seleccionado(s) en un control
	InactiveBorder	Borde de la ventana inactiva
	InactiveCaption	Título de la ventana inactiva
	InactiveCaptionText	Texto de un título inactivo
	InfoBackground	Color de fondo de los tooltips
	InfoText	Texto de los tooltips
	Menu	Fondo de los menús
	MenuText	Texto de los menús
	Scrollbar	Área gris de la barra de desplazamiento
	ThreeDDarkShadow	Sombra oscura de los elementos 3D
	ThreeDFace	Color frontal de los elementos 3D
	ThreeDHighlight	Color brillante de los elementos 3D
	ThreeDLightShadow	Color claro de los elementos 3D (bordes frente a la fuente de luz)
	ThreeDShadow	Sombra oscura de los elementos 3D
	Window	Fondo de la ventana
	WindowFrame	Marco de la ventana
	WindowText	Texto en las ventanas

Aunque es posible definir los colores en CSS utilizando estos nombres, se trata de un método que nunca se utiliza, por lo que se puede considerar prácticamente como una rareza de CSS.

3.2.6. Colores web safe

Como cada componente RGB de los colores puede tomar un valor entre 0 y 255, el número total de colores que se pueden representar con este formato es de $256 \times 256 \times 256 = 16.777.216$ colores. Sin embargo, en la década de los 90 los monitores de los usuarios no eran capaces de mostrar más de 256 colores diferentes.

A partir de todos los colores disponibles, se eligieron 216 colores que formaron la paleta de colores "web safe". Esta paleta de colores podía ser utilizada por los diseñadores con la seguridad de que se verían correctamente en cualquier navegador de cualquier sistema operativo de cualquier usuario.

Hoy en día, su importancia ha descendido notablemente, ya que prácticamente todos los usuarios utilizan dispositivos con una profundidad de color de 16 y 32 bits.

Colores web safe

000	300	600	900	C00	*F00*
003	303	603	903	C03	*F03*
006	306	606	906	C06	F06
009	309	609	909	C09	F09
00C	30C	60C	90C	C0C	F0C
00F	30F	60F	90F	C0F	*F0F*
030	330	630	930	C30	F30
033	333	633	933	C33	F33
036	336	636	936	C36	F36
039	339	639	939	C39	F39
03C	33C	63C	93C	C3C	F3C
03F	33F	63F	93F	C3F	F3F
060	360	660	960	C60	F60
063	363	663	963	C63	F63
066	366	666	966	C66	F66
069	369	669	969	C69	F69
06C	36C	66C	96C	C6C	F6C
06F	36F	66F	96F	C6F	F6F
090	390	690	990	C90	F90
093	393	693	993	C93	F93
096	396	696	996	C96	F96
099	399	699	999	C99	F99
09C	39C	69C	99C	C9C	F9C
09F	39F	69F	99F	C9F	F9F
0C0	3C0	6C0	9C0	CC0	FC0
0C3	3C3	6C3	9C3	CC3	FC3
0C6	3C6	6C6	9C6	CC6	FC6
0C9	3C9	6C9	9C9	CC9	FC9
0CC	3CC	6CC	9CC	CCC	FCC
0CF	3CF	6CF	9CF	CCF	FCF
0F0	3F0	*6F0*	9F0	CF0	*FF0*
0F3	*3F3*	*6F3*	9F3	CF3	*FF3*
0F6	*3F6*	6F6	9F6	*CF6*	*FF6*
0F9	3F9	6F9	9F9	CF9	FF9
0FC	*3FC*	6FC	9FC	CFC	FFC
0FF	*3FF*	*6FF*	9FF	CFF	*FFF*

EJERCICIO: Realizar el diseño de una cabecera sobre micología, utilizando cada uno de los modelos de colores estudiados.

4. Modelo de cajas

El modelo de cajas o *"box model"* es seguramente la característica más importante del lenguaje de hojas de estilos CSS, ya que condiciona el diseño de todas las páginas web. El modelo de cajas es el comportamiento de CSS que hace que **todos los elementos de las páginas se representen mediante cajas rectangulares.**

Las cajas de una página **se crean automáticamente.** Cada vez que se inserta una etiqueta HTML, se crea una nueva caja rectangular que encierra los contenidos de ese elemento. La siguiente imagen muestra las tres cajas rectangulares que crean las tres etiquetas HTML que incluye la página:

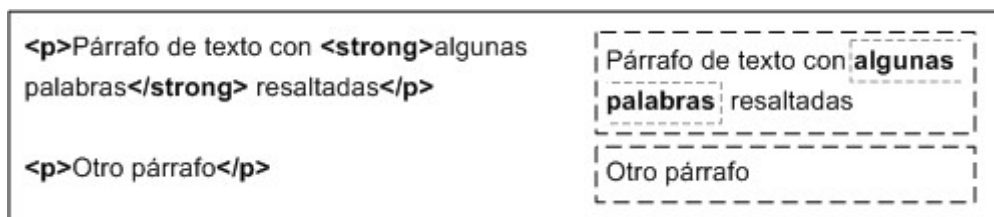
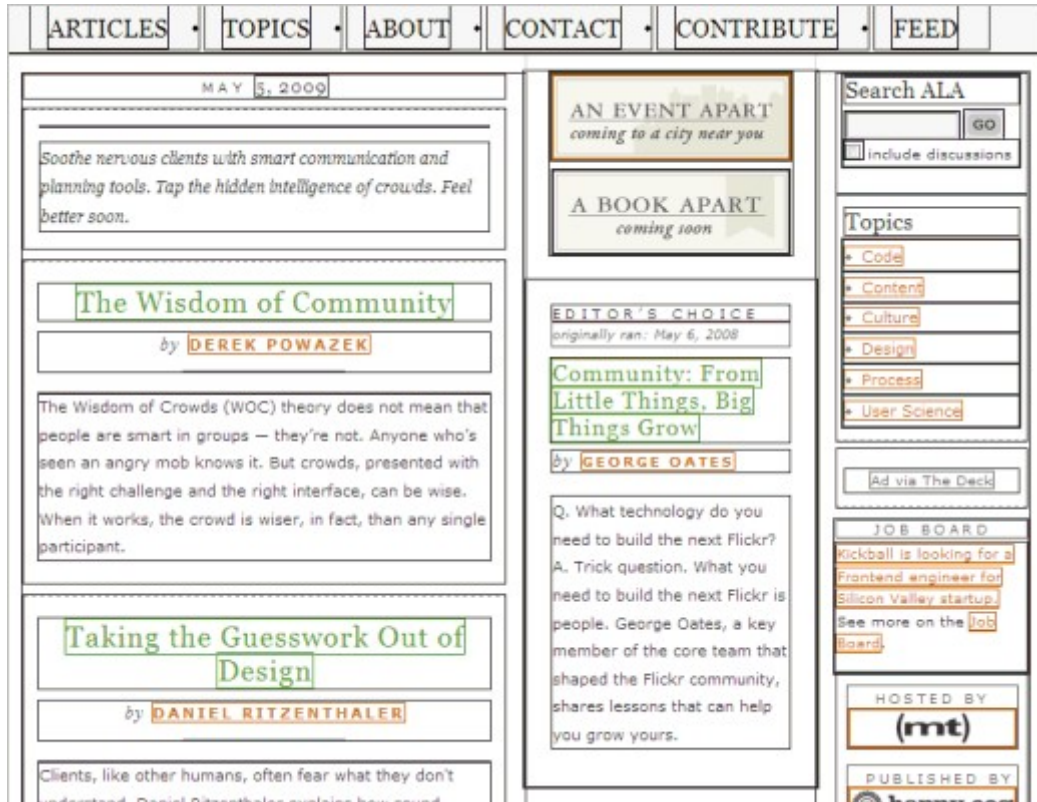


Figura 4.1. Las cajas se crean automáticamente al definir cada elemento HTML

Las cajas de las páginas **no son visibles a simple vista** porque inicialmente no muestran ningún color de fondo ni ningún borde. La siguiente imagen muestra las cajas que forman una página después de forzar a que todas las cajas muestren un borde:



Los navegadores crean y colocan las cajas de forma automática, pero CSS permite modificar todas sus características. Cada una de las cajas está formada por seis partes, tal y como muestra la siguiente imagen:

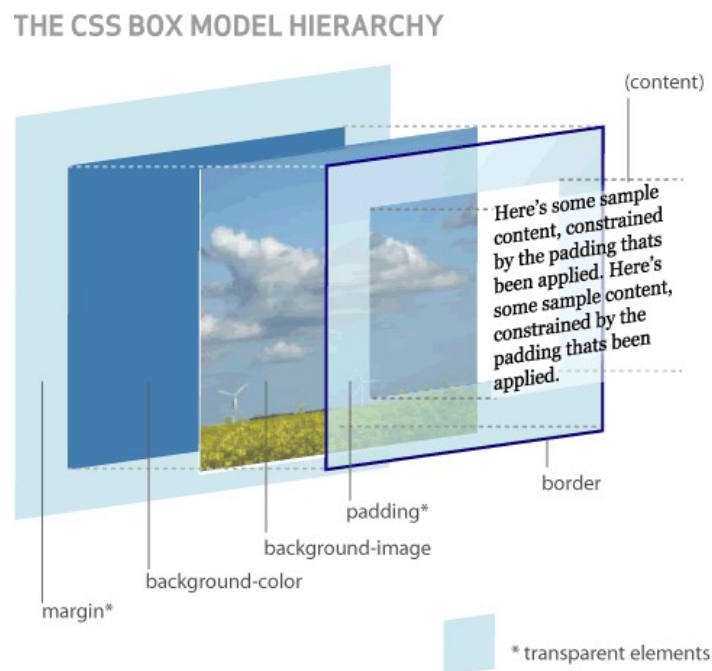
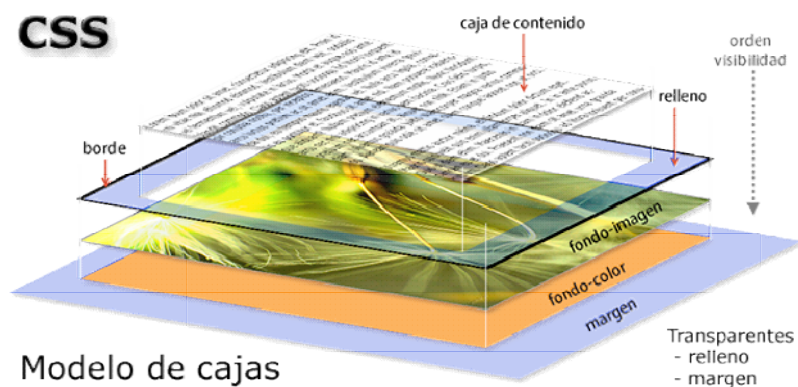


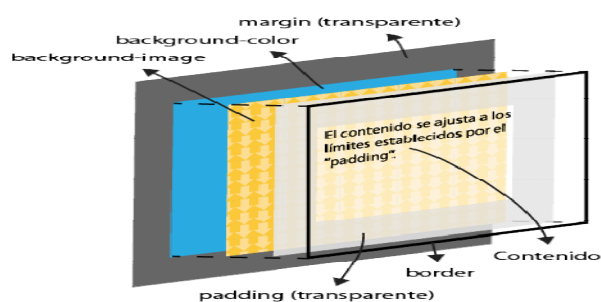
Figura 4.3. Representación tridimensional del box model de CSS

(Esquema utilizado con permiso de <http://www.hicksdesign.co.uk/boxmodel/>)

Otra forma de verlo:



Otra mas..



Las partes que componen cada caja y su orden de visualización desde el punto de vista del usuario son las siguientes:

- Contenido (*content*): se trata del contenido HTML del elemento (las palabras de un párrafo, una imagen, el texto de una lista de elementos, etc.)
- Relleno (*padding*): espacio libre opcional existente entre el contenido y el borde.
- Borde (*border*): línea que encierra completamente el contenido y su relleno.
- Imagen de fondo (*backgroundimage*): imagen que se muestra por detrás del contenido y el espacio de relleno.
- Color de fondo (*background color*): color que se muestra por detrás del contenido y el espacio de relleno.
- Margen (*margin*): separación opcional existente entre la caja y el resto de cajas adyacentes.

El relleno y el margen son transparentes, por lo que en el espacio ocupado por el relleno se muestra el color o imagen de fondo (si están definidos) y en el espacio ocupado por el margen se muestra el color o imagen de fondo de su elemento padre (si están definidos). Si ningún elemento padre tiene definido un color o imagen de fondo, se muestra el color o imagen de fondo de la propia página (si están definidos).

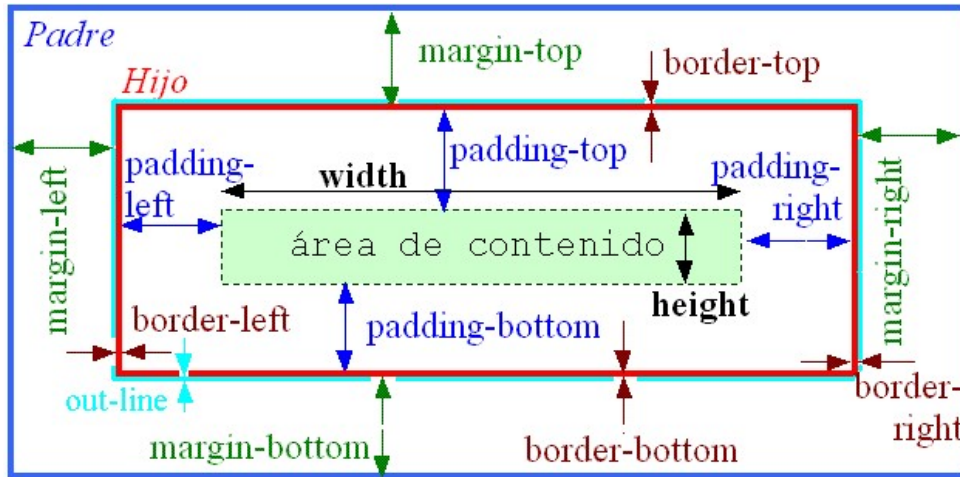
Si una caja define tanto un color como una imagen de fondo, la imagen tiene más prioridad y es la que se visualiza. No obstante, si la imagen de fondo no cubre totalmente la caja del elemento o si la imagen tiene zonas transparentes, también se visualiza el color de fondo. Combinando imágenes transparentes y colores de fondo se pueden lograr efectos gráficos muy interesantes.

El complemento TILT 3D de Firefox, es muy interesante porque ofrece una visión tridimensional de las cajas que forman una página. Este complemento también puede incorporarse a Google Chrome.



4.1 Dimensiones de la caja

El siguiente diagrama muestra cómo se relacionan estas áreas y la terminología usada para referirse a las partes de margin, border y padding:



Las dimensiones del área del contenido de una caja --el *ancho del contenido* y la *altura del contenido*-- dependen de varios factores: si el elemento que genera la caja tiene asignadas las propiedades 'width' o 'height', si la caja contiene texto u otras cajas, si la caja es una tabla, etc. El ancho y la altura de la caja lo veremos mas adelante.

El *ancho de la caja* está dado por la suma de los márgenes, bordes y rellenos izquierdos y derechos, y el *ancho del contenido*. La *altura* está dada por la suma de los márgenes, bordes y rellenos superiores e inferiores, y la altura del contenido.

El estilo del fondo de las distintas áreas de una caja es determinado como sigue:

- *Area del contenido*: La propiedad 'background' del elemento generador.
- *Area del relleno (padding)*: La propiedad 'background' del elemento generador.
- *Area del borde (border)*: Las propiedades del borde del elemento generador.
- *Area del margen (margin)*: Los márgenes son siempre transparentes.

Ejemplo de márgenes, rellenos y bordes

Este ejemplo ilustra cómo interactúan los márgenes, los rellenos y los bordes. El documento HTML de ejemplo:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<HTML>
<HEAD>
<TITLE>Ejemplo de márgenes, rellenos y bordes</TITLE>
<STYLE type="text/css">
  UL {
    background: green;
    margin: 12px 12px 12px 12px;
    padding: 3px 3px 3px 3px;
```

```

width:200px;
/* No se especifica border */
}
LI {
color: black;          /* el color del texto es negro */
background: gray;      /* Contenido, padding será gris */
margin: 12px 12px 12px 12px;
padding: 12px 0px 12px 12px; /* Note 0px para padding right */
list-style: none       /* sin viñeta antes de un ítem */
                      /* Ningún borde definido */
}
LI.withborder {
border-style: dashed;
border-width: medium;  /* pone el ancho de border para todos los lados */
border-color: black;
}
</STYLE>
</HEAD>
<BODY>
<UL>
<LI>El primer elemento
<LI class="withborder">El segundo elemento de la lista es
más largo y se ajusta.
</UL>
</BODY>
</HTML>

```

Resultado de una estructura del documento con (entre otras relaciones) un elemento UL que tiene dos LI hijos.

4.2 Propiedades del margen: 'margin-top', 'margin-right', 'margin-bottom', 'margin-left' y 'margin'

Las propiedades del margen especifican el ancho del área del margen de una caja. La propiedad resumida 'margin' determina el margen para los cuatro lados mientras que las otras propiedades sólo determinan su lado respectivo.

<medida>

Especifica un ancho fijo.

<porcentaje>

El porcentaje es calculado con respecto al *ancho* del bloque de contención de la caja generada. Esto es así para 'margin-top' y 'margin-bottom', excepto en el contexto de la página, donde los porcentajes se refieren a la altura de la caja de la página.

auto

(Esto es así:

margin: arriba derecha abajo izquierda; Es decir que empieza con la margen superior y va en sentido de las manecillas del reloj.

Y se simplifica así:

margin: arribayabajoderechaeizquierda; es decir que da el mismo margen arriba y abajo y luego el mismo margen derecho e izquierdo.

En resumen ***margin: 20px auto;*** lo que hace es dar un margen de 20 px arriba y abajo y un margen automático a los lados, es por eso que centra la capa.)

Los valores negativos para las propiedades de los márgenes son permitidos, pero pueden existir limitaciones específicas de la implementación.

'margin-top', 'margin-right', 'margin-bottom', 'margin-left'

Estas propiedades determinan los márgenes superior, derecho, inferior e izquierdo de una caja.

```
H1 { margin-top: 2em }
```

'margin'

La propiedad 'margin' es una propiedad resumida para establecer 'margin-top', 'margin-right', 'margin-bottom', y 'margin-left' en un mismo lugar de la hoja de estilo.

Si hay sólo un valor, se aplica a todos los lados. Si hay dos valores, los márgenes superior e inferior son determinados por el primer valor y los márgenes derecho e izquierdo son determinados por el segundo. Si hay tres valores, el superior es definido por el primer valor, el izquierdo y el derecho son definidos por el segundo, y el inferior es definido por el tercero. Si hay cuatro valores, ellos se aplican al superior, derecho, inferior e izquierdo, respectivamente.

```
BODY { margin: 2em } /* todos los márgenes en 2em */  
BODY { margin: 1em 2em } /* top & bottom = 1em, right & left = 2em */  
BODY { margin: 1em 2em 3em } /* top=1em, right=2em, bottom=3em, left=2em */
```

La última regla del ejemplo de arriba es equivalente al ejemplo de abajo:

```
BODY {  
margin-top: 1em;  
margin-right: 2em;  
margin-bottom: 3em;  
margin-left: 2em; /* copiado del lado opuesto (right) */  
}
```

Márgenes cerrados

En esta especificación, la expresión *márgenes cerrados* significa que los márgenes adyacentes (ningún área de relleno o de borde los separa) de dos o más cajas (que pueden estar una al lado de la otra o anidadas) se combinan para formar un solo margen.

En CSS2, los márgenes horizontales nunca se cierran.

Los márgenes verticales pueden cerrarse entre ciertas cajas:

- Dos o más márgenes verticales adyacentes de cajas de bloques en el flujo normal se cierran. El ancho del margen resultante es el máximo de los anchos de los márgenes adyacentes. En el caso de márgenes negativos, el máximo absoluto de los márgenes adyacentes negativos es restado del máximo de los márgenes adyacentes positivos. Si no hay ningún margen positivo, el máximo absoluto de los márgenes adyacentes negativos es restado de cero.
- Los márgenes verticales entre una caja flotante y cualquier otra caja no se cierran.
- Los márgenes de cajas con posiciones absoluta y relativa no se cierran.

4.3 Propiedades del relleno: 'padding-top', 'padding-right', 'padding-bottom', 'padding-left' y 'padding'

Las propiedades del relleno especifican el ancho del área de relleno de una caja. La propiedad resumida 'padding' define el relleno para los cuatro lados mientras que las otras propiedades de relleno sólo definen sus respectivos lados.

Pueden tomar uno de los siguientes valores:

<medida>

Especifica una medida fija.

<porcentaje>

El porcentaje es calculado con respecto al *ancho* del bloque de contención de la caja generada, aún para 'padding-top' y 'padding-bottom'.

A diferencia de las propiedades del margen, los valores para los valores de relleno no pueden ser negativos. Al igual que las propiedades del margen, los valores de porcentaje para las propiedades del relleno se refieren al ancho del bloque de contención de la caja generada.

'padding-top', 'padding-right', 'padding-bottom', 'padding-left'

Estas propiedades determinan el relleno superior, derecho, inferior e izquierdo de una caja.

```
BLOCKQUOTE { padding-top: 0.3em }
```

'padding'

La propiedad 'padding' es una propiedad resumida para definir 'padding-top', 'padding-right', 'padding-bottom' y 'padding-left' en un mismo lugar en la hoja de estilo.

Si hay sólo un valor, se aplica a todos los lados. Si hay dos valores, los rellenos superior e inferior son determinados por el primer valor y los rellenos derecho e izquierdo son determinados por el segundo. Si hay tres valores, el superior es definido por el primer valor, el izquierdo y el derecho son definidos por el segundo, y el inferior es definido

por el tercero. Si hay cuatro valores, ellos se aplican al superior, derecho, inferior e izquierdo, respectivamente.

El color de la superficie o la imagen del área de relleno es especificado a través de la propiedad 'background':

```
H1 {  
background: white;  
padding: 1em 2em;  
}
```

El ejemplo de arriba especifica un relleno vertical igual a '1em' ('padding-top' y 'padding-bottom') y un relleno horizontal de '2em' ('padding-right' y 'padding-left'). La unidad 'em' es relativa al tamaño de la fuente del elemento: '1em' es igual al tamaño de la fuente en uso.

4.4 Propiedades del borde

Las propiedades del borde especifican el ancho, color y estilo del área del borde de una caja. Estas propiedades se aplican a todos los elementos.

***Nota.** Particularmente para HTML, las aplicaciones del usuario pueden procesar los bordes de ciertos elementos (ej., botones, menús, etc.) de manera diferente a los elementos "ordinarios".*

Ancho del borde: 'border-top-width', 'border-right-width', 'border-bottom-width', 'border-left-width' y 'border-width'

Las propiedades del ancho del borde especifican la medida del área del borde. Las propiedades definidas en esta sección se refieren al tipo de valor de **<border-width>**, que puede tomar uno de los siguientes valores:

thin

Un borde fino.

medium

Un borde mediano.

thick

Un borde grueso.

<medida>

El grosor del borde tiene un valor explícito. Las dimensiones del borde explícitas no pueden ser negativas.

La interpretación de los primeros tres valores depende de la aplicación del usuario. Sin embargo, se deben mantener las siguientes relaciones:

'thin' <= 'medium' <= 'thick'.

Además, estos grosores deben mantenerse constantes a lo largo de todo un documento.

'border-top-width', 'border-right-width', 'border-bottom-width', 'border-left-width'

Estas propiedades determinan al ancho de los bordes superior, derecho, inferior e izquierdo de una caja.

'border-width'

Esta propiedad es una propiedad resumida para especificar 'border-top-width', 'border-right-width', 'border-bottom-width' y 'border-left-width' en un mismo lugar en la hoja de estilo.

Si hay sólo un valor, se aplica a todos los lados. Si hay dos valores, los bordes superior e inferior son determinados por el primer valor y los bordes derecho e izquierdo son determinados por el segundo. Si hay tres valores, el superior es definido por el primer valor, el izquierdo y el derecho son definidos por el segundo, y el inferior es definido por el tercero. Si hay cuatro valores, ellos se aplican al superior, derecho, inferior e izquierdo, respectivamente.

En los ejemplos de abajo, los comentarios indican los grosores resultantes en los bordes superior, derecho, inferior e izquierdo:

```
H1 { border-width: thin }           /* thin thinthinthin */
H1 { border-width: thin thick }     /* thin thick thin thick */
H1 { border-width: thin thick medium } /* thin thick medium thick */
```

4.5 Color del borde: 'border-top-color', 'border-right-color', 'border-bottom-color', 'border-left-color' y 'border-color'

Las propiedades del color del borde especifican el color del borde de una caja.

'border-top-color', 'border-right-color', 'border-bottom-color', 'border-left-color'

'border-color'

La propiedad 'border-color' determina el color de los cuatro bordes. Los valores tienen los siguientes significados:

<color>

Especifica un valor de color.

transparent

El borde es transparente (no obstante puede tener grosor).

La propiedad 'border-color' puede tener de uno a cuatro valores, y los valores son aplicados a los distintos lados.

Si el color del borde de un elemento no es especificado con una propiedad del borde, las aplicaciones del usuario deben tomar el valor de la propiedad 'color' del elemento como el valor computado para el color del borde.

En este ejemplo, el borde será una línea negra sólida.

```
P {  
  color: black;  
  background: white;  
  border: solid;  
}
```

4.6 Estilo del borde: 'border-top-style', 'border-right-style', 'border-bottom-style', 'border-left-style' y 'border-style'

Las propiedades del estilo del borde especifican el estilo de la línea del borde de una caja (sólida, doble, punteada, etc.). Las propiedades definidas en esta sección se refieren al tipo de valor de **<border-style>**, que puede estar constituido por uno de los siguientes:

none

Ningún borde. Este valor fuerza al valor computado de 'border-width' a '0'.

hidden

Igual a 'none', excepto en términos de resolución de conflictos de bordes para los elementos de tabla.

dotted

El borde es una serie de puntos.

dashed

El borde es una serie de pequeños segmentos de línea.

solid

El borde es un único segmento de línea.

double

El borde son dos líneas sólidas. La suma de las dos líneas y el espacio entre ellas es igual al valor de 'border-width'.

groove

El borde luce como si estuviese tallado en el lienzo.

ridge

Lo opuesto a 'groove': el borde parece que estuviera sobresaliendo del lienzo.

inset

El borde hace que toda la caja luzca como si estuviera empotrada en el lienzo.

outset

Lo opuesto a 'inset': el borde hace que toda la caja parezca sobresalir del lienzo.

Todos los bordes son dibujados por encima del fondo de la caja. El color de los bordes con valores de 'groove', 'ridge', 'inset' y 'outset' depende de la propiedad 'color' del elemento.

Las aplicaciones de usuario HTML con conformidad pueden interpretar a 'dotted', 'dashed', 'double', 'groove', 'ridge', 'inset' y 'outset' como 'solid'.

'border-top-style', 'border-right-style', 'border-bottom-style', 'border-left-style'

'border-style'

La propiedad 'border-style' determina el estilo de los cuatro bordes. Puede tener de uno a cuatro valores, y los valores son distribuidos para los distintos lados.

```
#xy34 { border-style: solid dotted }
```

En el ejemplo de arriba, los bordes horizontales serán 'solid' y los bordes verticales serán 'dotted'.

Como el valor inicial del estilo de borde es 'none', ningún borde será visible a menos que se establezca el estilo de borde.

4.7 Propiedades resumidas del borde: 'border-top', 'border-bottom', 'border-right', 'border-left' y 'border'

'border-top', 'border-right', 'border-bottom', 'border-left'

Esta es una propiedad resumida para definir el ancho, estilo y color del borde superior, derecho, inferior e izquierdo de una caja.

```
H1 { border-bottom: thick solid red }
```

La regla de arriba determinará el ancho, el estilo y el color del borde por **debajo** del elemento H1. Los valores omitidos son colocados en sus valores iniciales. Como la siguiente regla no especifica un color de borde, el borde tendrá el color especificado en la propiedad 'color':

```
H1 { border-bottom: thick solid }
```

'border'

La propiedad 'border' es una propiedad resumida para colocar el mismo ancho, color y estilo a los cuatro bordes de una caja. A diferencia de las propiedades resumidas 'margin' y 'padding', la propiedad 'border' no puede definir diferentes valores para los cuatro bordes. Para eso, deben usarse una a más de las otras propiedades del borde.

Por ejemplo, la primera regla de abajo es equivalente al conjunto de cuatro reglas mostradas a continuación de ella.

```
P { border: solid red }  
P {  
  border-top: solid red;  
  border-right: solid red;  
  border-bottom: solid red;
```

```
border-left: solid red  
}
```

Como, hasta cierto punto, las propiedades tienen un funcionamiento que se superpone, el orden en que las reglas son especificadas es importante.

EJERCICIO:

Realizar una página que tenga solamente un div central de tres centímetros cuadrados, cuyo contenido será un refrán, debe de mostrarse el borde, debe tener un padding amplio, debe tener una imagen de fondo, a poder ser con transparencias, debe tener también un color llamativo de fondo y un margen amplio.

EJEMPLO DE LA APLICACIÓN DE CSS A LISTAS.

Las listas se utilizan para enumerar una serie de elementos, se utiliza la marca HTML ul (UnorderedList), y cada ítem de la lista con la marca HTML li (ListItem).

Las CSS nos permiten configurar las listas por medio de tres propiedades:

```
list-style-type  
list-style-position  
list-style-image
```

A list-style-type puede asignársele alguno de estos valores:

```
none  
disc  
circle  
square  
decimal  
decimal-leading-zero  
lower-roman  
upper-roman  
lower-alpha  
upper-alpha
```

Los valores de list-style-position:

```
inside  
outside
```

Los valores de list-style-image:

```
none  
url
```

Veamos un ejemplo que prueba todos los valores posibles que puede tomar la propiedad list-style-type:

```
<html>  
<head>  
<title>Problema</title>  
<link rel="StyleSheet" href="estilos.css" type="text/css">  
</head>  
<body>  
<ul class="vacio">  
<li>Brasil</li>  
<li>Uruguay</li>  
<li>Argentina</li>  
</ul>  
<ul class="circulorelleno">  
<li>Brasil</li>  
<li>Uruguay</li>  
<li>Argentina</li>  
</ul>  
<ul class="circulovacio">  
<li>Brasil</li>  
<li>Uruguay</li>  
<li>Argentina</li>  
</ul>  
<ul class="cuadrado">
```

```

<li>Brasil</li>
<li>Uruguay</li>
<li>Argentina</li>
</ul>
<ulclass="decimal">
<li>Brasil</li>
<li>Uruguay</li>
<li>Argentina</li>
</ul>
<ulclass="romanominuscula">
<li>Brasil</li>
<li>Uruguay</li>
<li>Argentina</li>
</ul>
<ulclass="romanomayuscula">
<li>Brasil</li>
<li>Uruguay</li>
<li>Argentina</li>
</ul>
<ulclass="letrasminusculas">
<li>Brasil</li>
<li>Uruguay</li>
<li>Argentina</li>
</ul>
<ulclass="letrasmayusculas">
<li>Brasil</li>
<li>Uruguay</li>
<li>Argentina</li>
</ul>
</body>
</html>

```

Luego la hoja de estilo es:

```

ul.vacio{
list-style-type:none;
}
ul.circulorelleno{
list-style-type:disc;
}
ul.decimal{
list-style-type:decimal;
}
ul.romanominuscula{
list-style-type:lower-roman;
}
ul.romanomayuscula{
list-style-type:upper-roman;
}
ul.circulovacio{
list-style-type:circle;
}
ul.cuadrado{
list-style-type:square;
}
ul.letrasminusculas{
list-style-type:lower-alpha;
}
ul.letrasmayusculas{
list-style-type:upper-alpha;
}

```

Lo que podemos ver es que cuando definimos las clases, le antecedemos al punto, el nombre de la marca donde se aplica dicha clase (en este caso ul, es decir que esta clase sólo tiene sentido aplicarla a dicha marca).

APLICACIÓN DE HOJAS DE ESTILOS A TABLAS

Veamos con un ejemplo como podemos afectar una tabla HTML con CSS.

```
<html>
<head>
<title>Problema</title>
<link rel="StyleSheet" href="estilos.css" type="text/css">
</head>
<body>
<table>
<caption>
cantidad de lluvia caida en mm.
</caption>
<thead>
<tr>
<th>Provincia</th><th>Enero</th><th>Febrero</th><th>Marzo</th>
</tr>
</thead>
<tbody>
<tr>
<th>Córdoba</th>
<td>210</td><td>170</td><td>120</td>
</tr>
<tr>
<th>Buenos Aires</th>
<td>250</td><td>190</td><td>140</td>
</tr>
<tr>
<th>Santa Fe</th>
<td>175</td><td>140</td><td>120</td>
</tr>
</tbody>
</table>
</body>
</html>
```

La hoja de estilo definida a esta tabla es:

```
caption
{
font-family:arial;
font-size:15px;
text-align: center;
margin: 0px;
font-weight: bold;
padding:10px;
}

table
{
border-collapse: collapse;
}

th
{
border-right: 1px solid #fff;
border-bottom: 1px solid #fff;
padding: 0.5em;
background-color:#6495ed;;
}
```

```

}

theadth
{
background-color: #6495ed;
color: #fff;
}

tbodyth
{
font-family:arial;
font-weight: normal;
background-color: #6495ed;
color:#ff0;
}

td {
border: 1px solid #000;
padding: .5em;
background-color:#ed8f63;
width:100px;
text-align:center;
}

```

La marca caption dentro de una tabla es el título que debe aparecer arriba.

La propiedad border-collapse puede tomar dos valores: collapse o separate. Separate deja las celdas con unos pixeles de separación, no así collapse.

El resto es la definición de una serie de reglas para las marcas th, th dentro de la marca tbody, th dentro de la marca thead y por último td.

APLICACIÓN DE HOJAS DE ESTILOS A UN FORMULARIO

Un formulario es el elemento esencial para el envío de datos al servidor por parte del visitante del sitio.

Veamos un ejemplo donde implementamos un formulario y le aplicamos una serie de reglas de estilo a las diferentes marcas HTML que intervienen:

```

<html>
<head>
<title>Problema</title>
<link rel="StyleSheet" href="estilos.css" type="text/css">
</head>
<body>
<div id="contenedorform">
<form method="post" action="#">
<label>Ingresa nombre:</label>
<input type="text" name="nombre" size="30">
<br>
<label>Ingresa mail:</label>
<input type="text" name="mail" size="45">
<br>
<label>Comentarios:</label>
<textareaname="comentarios" cols="30" rows="5"></textarea>
<br>

```

```
<input class="botonsubmit" type="submit" value="confirmar">
</form>
</div>
</body>
</html>
```

La hoja de estilo que se aplica es:

```
#contenedorform {
width:500px;
margin-left:20px;
margin-top:10px;
background-color:#ffe;
border:1px solid #CCC;
padding:10px 0 10px 0;
}

#contenedorform form label {
width:120px;
float:left;
font-family:verdana;
font-size:14px;
}
.botonsubmit {
color:#f00;
background-color:#bbb;
border: 1px solid #fff;
}
```

Podemos observar que definimos un div contenedor y dentro de este el formulario. Para que los textos aparezcan a la izquierda, definimos una serie de label que las flotamos a izquierda, por lo que los controles del formulario aparecerán a derecha todos encolumnados.