CENTRE ESPECÍFIC
D´EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

TEMA 3.
CSS 2

LM
CFGS DAW

Autor: Pascual Ligero

Revisado por:

Fco. Javier Valero – franciscojavier.valero@ceedcv.es

2019/2020

Versión:191012.2129

Licencia

CC BY-NC-SA 3.0 ES Reconocimiento – NoComercial – CompartirIgual (by-nc-sa): No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

NOTA: Esta es una obra derivada de la original realizada por Pascual Ligero.

Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

Importante

Atención

Interesante

ÍNDICE DE CONTENIDO

UD03.2. CSS

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 uno de los errores más habituales de los diseñadores que empiezan con CSS. Algunas propiedades permiten indicar medidas negativas, aunque habitualmente sus valores son positivos. Si el valor decimal de una medida es inferior a `1`, se puede omitir el `0` de la izquierda (`0.5em` es equivalente a `.5em`).

1.1.1 Unidades absolutas

Una medida indicada mediante unidades absolutas está completamente definida, ya que su valor no depende de otro valor de referencia. A continuación se muestra la lista completa de unidades absolutas definidas por CSS y su significado:

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

A continuación se muestran ejemplos de utilización de **unidades absolutas**:

```
/* El cuerpo de la página debe mostrar un margen de media pulgada */
body { margin: 0.5in; }

/* Los elementos <h1> deben mostrar un interlineado de 2 centímetros */
h1 { line-height: 2cm; }

/* Las palabras de todos los párrafos deben estar separadas 4 milímetros entre si
*/
p { word-spacing: 4mm; }

/* Los enlaces se deben mostrar con un tamaño de letra de 12 puntos */
a { font-size: 12pt }

/* Los elementos <span> deben tener un tamaño de letra de 1 pica */
span { font-size: 1pc }
```

La principal ventaja de las unidades absolutas es que su valor es directamente el valor que se debe utilizar, sin necesidad de realizar cálculos intermedios. Su principal desventaja es que son muy poco flexibles y no se adaptan fácilmente a los diferentes medios.

De todas las unidades absolutas, la única que suele utilizarse es el punto (`pt`). Se trata de la unidad de medida preferida para establecer el tamaño del texto en los documentos que se van a imprimir, es decir, para el medio `print` de CSS, tal y como se verá más adelante.

1.1.2 Unidades relativas

La unidades relativas, a diferencia de las absolutas, no están completamente definidas, ya **que su valor siempre está referenciado respecto a otro valor**. A pesar de su aparente dificultad, s**on las más utilizadas** en el diseño web por la flexibilidad con la que se adaptan a los diferentes medios.

A continuación se muestran las tres unidades de medida relativas definidas por CSS y la referencia que toma cada una para determinar su valor real:

- **`em`,** (no confundir con la etiqueta `<em>` de HTML) relativa respecto del tamaño de letra del elemento.
- **`ex`,** relativa respecto de la altura de la letra `x` (*"equis minúscula"*) del tipo y tamaño de letra del elemento.
- **`px`,** (píxel) relativa respecto de la resolución de la pantalla del dispositivo en el que se visualiza la página HTML.

Aunque no es una definición exacta, la unidad `1em` equivale a la anchura de la letra `M` (*"eme mayúscula"*) del tipo y tamaño de letra del elemento.

Si se considera el siguiente ejemplo:

```
p { margin: 1em; }
```

La regla CSS anterior indica que los párrafos deben mostrar un margen de anchura igual a `1em`. Como se trata de una unidad de medida relativa, **es necesario realizar un cálculo matemático para determinar la anchura real de ese margen**.

La unidad de medida `em` siempre hace referencia al tamaño de letra del elemento. Por otra parte, todos los navegadores muestran por defecto el texto de los párrafos con un tamaño de letra de 16 píxel. Por tanto, en este caso el margen de `1em` equivale a un margen de anchura `16px`.

A continuación se modifica el ejemplo anterior para cambiar el tamaño de letra de los párrafos:

```
p { font-size: 32px; margin: 1em; }
```

El valor del margen sigue siendo el mismo en unidades relativas (`1em`) pero su valor real ha variado porque el tamaño de letra de los párrafos ha variado. En este caso, el margen tendrá una anchura de `32px`, ya que `1em` siempre equivale al tamaño de letra del elemento.

La gran ventaja de las unidades relativas es que siempre mantienen las proporciones del diseño de la página. Establecer el margen de un elemento con el valor `1em` equivale a indicar que *"el margen del elemento debe ser del mismo tamaño que su letra y debe cambiar proporcionalmente"*.

En efecto, si el tamaño de letra de un elemento aumenta hasta un valor enorme, su margen de `1em` también será enorme.

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, por lo que su valor es aproximadamente la mitad que el de la unidad `em`.

Por último, 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 la página HTML.

Si un elemento tiene una anchura de `400px`, ocupará la mitad de una pantalla con una resolución de `800x600`, pero ocupará menos de la tercera parte en una pantalla con resolución de `1440x900`.

Las unidades de medida se pueden mezclar en 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 x 10px = 25px`.

Como se vio en los capítulos anteriores, el valor de la mayoría de propiedades CSS se hereda de padres a 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.

1.1.3 Porcentajes

El porcentaje también es una unidad de medida relativa, aunque por su importancia CSS la trata de forma separada a `em`, `ex` y `px`. 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` es decir `480px`.**

1.1.4 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.

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

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.

2.1.1 Palabras clave

CSS define 17 palabras clave para referirse a los colores básicos. Las palabras se corresponden con el nombre en inglés de cada color:

`aqua`, `black`, `blue`, `fuchsia`, `gray`, `green`, `lime`, `maroon`, `navy`, `olive`, `orange`, `purple`, `red`, `silver`, `teal`, `white`, `yellow`

Figura 2.1 Colores definidos mediante las palabras clave de CSS

La imagen anterior ha sido extraída de la sección sobre colores de la especificación oficial de CSS.

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. La lista completa se puede ver en en.wikipedia.org/wiki/Websafe.

2.1.2 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.

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.

2.1.3 RGB porcentual

Las componentes RGB de un color también se pueden indicar mediante un porcentaje. El funcionamiento y la sintaxis de este método es el mismo que el del RGB decimal. La única diferencia es que en este caso el valor de las componentes RGB puede tomar valores entre 0% y 100%. Por tanto, para transformar un valor RGB decimal en un valor RGB porcentual, es preciso realizar una regla de tres considerando que 0 es igual a 0% y 255 es igual a 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%.

2.1.4 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.

Para entender el modelo RGB hexadecimal, en primer lugar es preciso introducir un concepto matemático llamado *sistema numérico hexadecimal*.

Definir un color en CSS con el método RGB hexadecimal requiere realizar los siguientes pasos: - Determinar las componentes RGB decimales del color original, por ejemplo: R = 71, G = 98, B = 176 - 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** - 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:

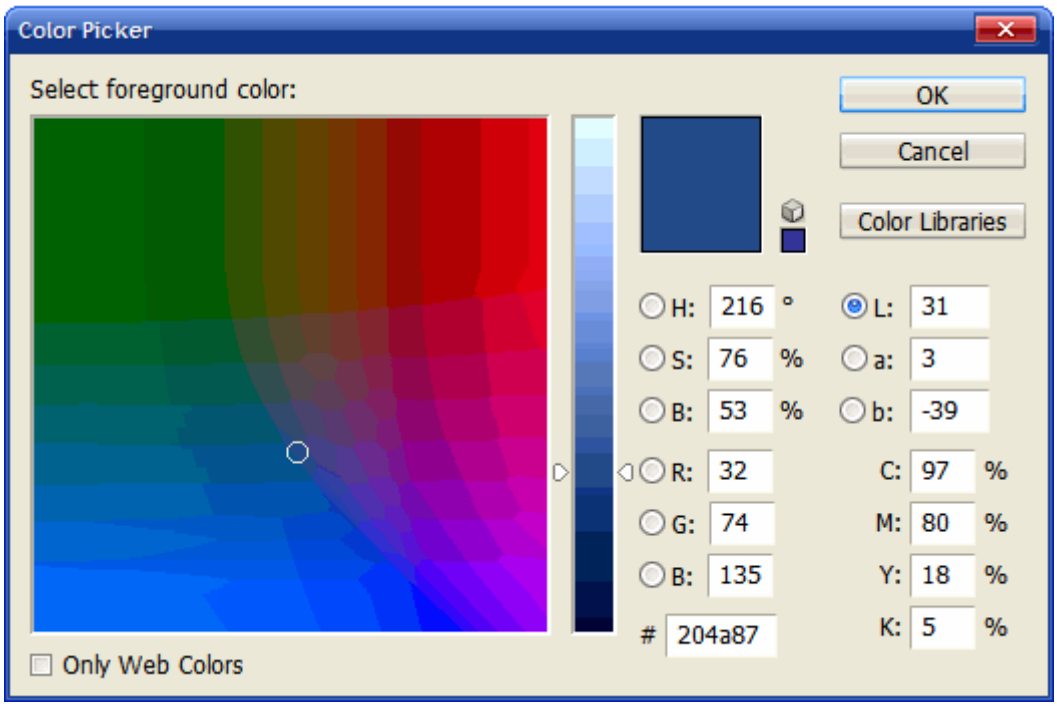

Figura 2.2 Herramienta de color de Photoshop para definir los colores según los modelos RGB, CMYK, Lab, HSB y RGB hexadecimal

El formato RGB hexadecimal es la forma más compacta de indicar un color, ya que incluso es posible 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.

3. 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:

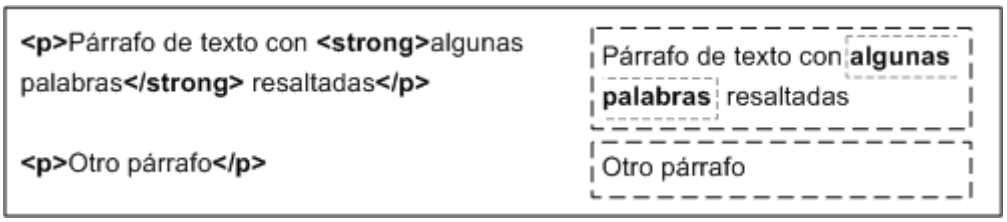

Figura 3.1 Las cajas se crean automáticamente al definir cada elemento HTML

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 la página web de http://www.alistapart.com/ después de forzar a que todas las cajas muestren su borde:

Figura 3.2 Cajas que forman la página alistapart.com

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:

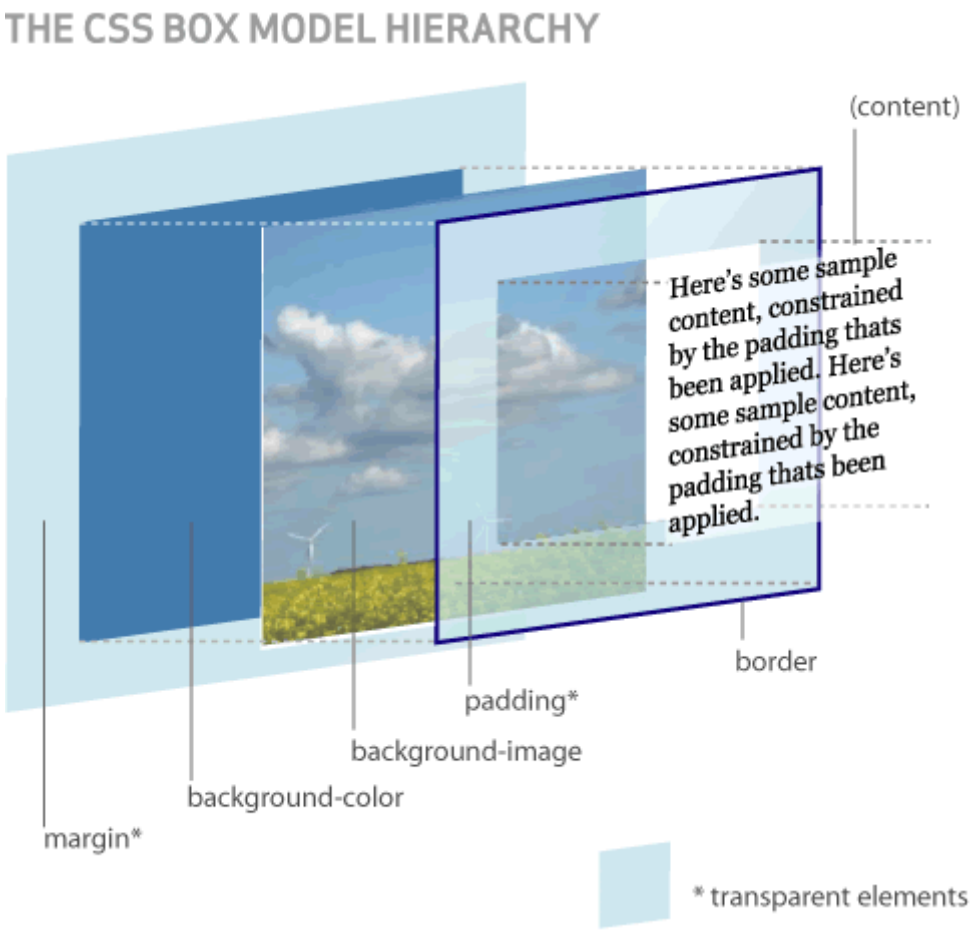

Figura 4.3 Representación tridimensional del box model de CSS

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** (*background image*): 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 imagenes transparentes y colores de fondo se pueden lograr efectos gráficos muy interesantes.

3.1 Anchura y altura

3.1.1 Anchura

La propiedad CSS que controla la **anchura** de la caja de los elementos se denomina **`width`**.

Propiedad	**width**
Valores	unidad de medida | porcentaje | auto | inherit
Se aplica a	Todos los elementos, salvo los elementos en línea que no sean imágenes, las filas de tabla y los grupos de filas de tabla
Valor inicial	auto
Descripción	Establece la anchura de un elemento

La propiedad **`width`** no admite valores negativos y los valores en porcentaje se calculan a partir de la anchura de su elemento padre. El valor `inherit` indica que la anchura del elemento se hereda de su elemento padre. El valor `auto`, que es el que se utiliza si no se establece de forma explícita un valor a esta propiedad, indica que el navegador debe calcular automáticamente la anchura del elemento, teniendo en cuenta sus contenidos y el sitio disponible en la página.

El siguiente ejemplo establece el valor de la anchura del elemento `<div>` lateral:

```
#lateral { width: 200px; }
```

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

3.1.2 Altura

La propiedad CSS que controla la **altura** de los elementos se denomina **`height`**.

Propiedad	**height**
Valores	unidad de medida | porcentaje | auto | inherit
Se aplica a	Todos los elementos, salvo los elementos en línea que no sean imágenes, las columnas de tabla y los grupos de columnas de tabla
Valor inicial	auto
Descripción	Establece la altura de un elemento

Al igual que sucede con `width`, la propiedad `height` no admite valores negativos. Si se indica un porcentaje, se toma como referencia la altura del elemento padre. Si el elemento padre no tiene una altura definida explícitamente, se asigna el valor `auto` a la altura.

El valor `inherit` indica que la altura del elemento se hereda de su elemento padre. El valor `auto`, que es el que se utiliza si no se establece de forma explícita un valor a esta propiedad, indica que el navegador debe calcular automáticamente la altura del elemento, teniendo en cuenta sus contenidos y el sitio disponible en la página.

El siguiente ejemplo establece el valor de la altura del elemento de cabecera:

```
#cabecera { height: 60px; }

<div id="cabecera">
  ...
</div>
```

3.2 Margen y relleno

3.2.1 Margen

CSS define cuatro propiedades para controlar cada uno de los márgenes horizontales y verticales de un elemento.

Propiedades	**margin-top, margin-right, margin-bottom, margin-left**
Valores	unidad de medida | porcentaje | auto | inherit
Se aplica a	Todos los elementos, salvo margin-top y margin-bottom que sólo se aplican a los elementos de bloque y a las imágenes
Valor inicial	0
Descripción	Establece cada uno de los márgenes horizontales y verticales de un elemento

Cada una de las propiedades establece la separación entre el borde lateral de la caja y el resto de cajas adyacentes:

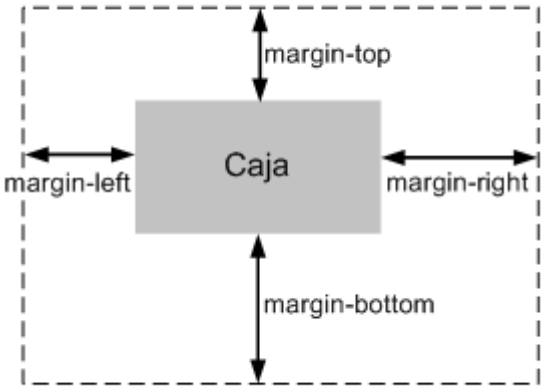

Figura 4.4 Las cuatro propiedades relacionadas con los márgenes

Las unidades más utilizadas para indicar los márgenes de un elemento son los **píxeles** (cuando se requiere una precisión total), los **em** (para hacer diseños que mantengan las proporciones) y los **porcentajes** (para hacer diseños líquidos o fluidos).

El siguiente ejemplo añade un margen izquierdo al segundo párrafo:

```
.destacado {
  margin-left: 2em;
}

<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nam et elit.
Vivamus placerat lorem. Maecenas sapien. Integer ut massa. Cras diam ipsum,
laoreet non, tincidunt a, viverra sed, tortor.</p>

<p class="destacado">Vestibulum lectus diam, luctus vel, venenatis ultrices,
cursus vel, tellus. Etiam placerat erat non sem. Nulla molestie odio non
nisl tincidunt faucibus.</p>

<p>Aliquam euismod sapien eu libero. Ut tempor orci at nulla. Nam in eros
egestas massa vehicula nonummy. Morbi posuere, nibh ultricies consectetuer
tincidunt,
risus turpis laoreet elit, ut tincidunt risus sem et nunc.</p>
```

A continuación se muestra el aspecto del ejemplo anterior en cualquier navegador:

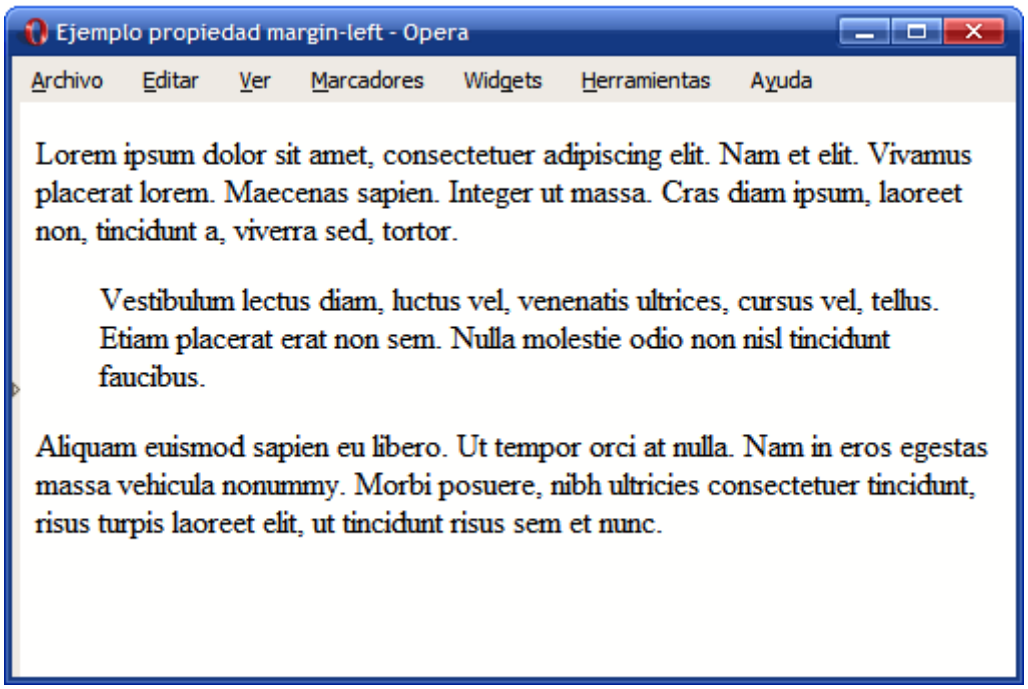

Figura 3.5 Ejemplo de propiedad margin-left

Los márgenes verticales (**`margin-top` y `margin-bottom`**) sólo se pueden aplicar a los elementos de bloque y las imágenes, mientras que los márgenes laterales (`margin-left` y `margin-right`) se pueden aplicar a cualquier elemento, tal y como muestra la siguiente imagen:

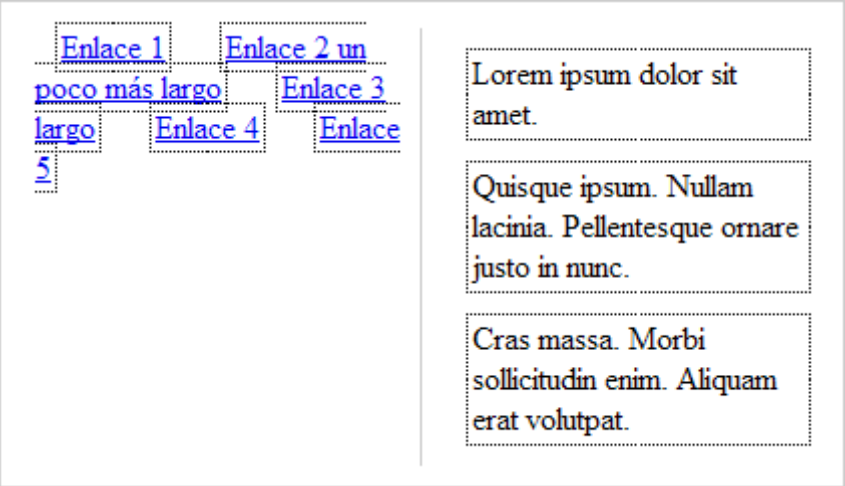

Figura 3.6 Los márgenes verticales sólo se aplican a los elementos de bloque e imágenes

La imagen anterior muestra el resultado de aplicar los mismos márgenes a varios enlaces **(elementos en línea)** y varios párrafos **(elementos de bloque)**. En los **elementos en línea los márgenes verticales no tienen ningún efecto**, por lo que los enlaces no muestran ninguna separación vertical, al contrario de lo que sucede con los párrafos. Sin embargo, **los márgenes laterales funcionan sobre cualquier tipo de elemento**, por lo que los enlaces se muestran separados entre sí y los párrafos aumentan su separación con los bordes laterales de su elemento contenedor.

Además de las cuatro propiedades que controlan cada uno de los márgenes del elemento, CSS define una propiedad especial que permite establecer los cuatro márgenes de forma simultánea. Estas propiedades especiales se denominan *"propiedades shorthand"* y CSS define varias propiedades de este tipo, como se verá más adelante.

La propiedad que permite definir de forma simultanea los cuatro márgenes se denomina `margin`.

Propiedad	**margin**
Valores	(unidad de medida | porcentaje | auto) {1, 4} | inherit
Se aplica a	Todos los elementos salvo algunos casos especiales de elementos mostrados como tablas
Valor inicial	-
Descripción	Establece de forma directa todos los márgenes de un elemento

La notación `{1, 4}` de la definición anterior significa que la propiedad `margin` admite entre uno y cuatro valores, con el siguiente significado:

- Si solo se indica **un valor**, todos los márgenes tienen ese valor.
- Si se indican **dos valores**, el primero se asigna al margen superior e inferior y el segundo se asigna a los márgenes izquierdo y derecho.
- Si se indican **tres valores**, el primero se asigna al margen superior, el tercero se asigna al margen inferior y el segundo valor se asigna los márgenes izquierdo y derecho.
- Si se indican los **cuatro valores**, el orden de asignación es: margen superior, margen derecho, margen inferior y margen izquierdo.

El ejemplo anterior de márgenes se puede reescribir utilizando la propiedad `margin`:

Código CSS original:

```
div img {
  margin-top: .5em;
  margin-bottom: .5em;
  margin-left: 1em;
  margin-right: .5em;
}
```

Alternativa directa:

```
div img {
  margin: .5em .5em .5m 1em;
}
```

Otra alternativa:

```
div img {
  margin: .5em;
  margin-left: 1em;
}
```

El comportamiento de los márgenes verticales es más complejo de lo que se puede imaginar. Cuando se juntan dos o más márgenes verticales, se fusionan de forma automática y la altura del nuevo margen será igual a la altura del margen más alto de los que se han fusionado.

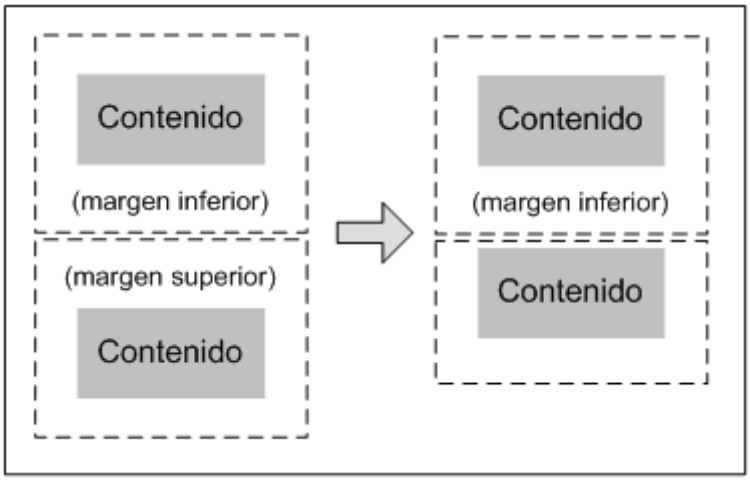

Figura 3.7 Fusión automática de los márgenes verticales

De la misma forma, si un elemento está contenido dentro de otro elemento, sus márgenes verticales se fusionan y resultan en un nuevo margen de la misma altura que el mayor margen de los que se han fusionado:

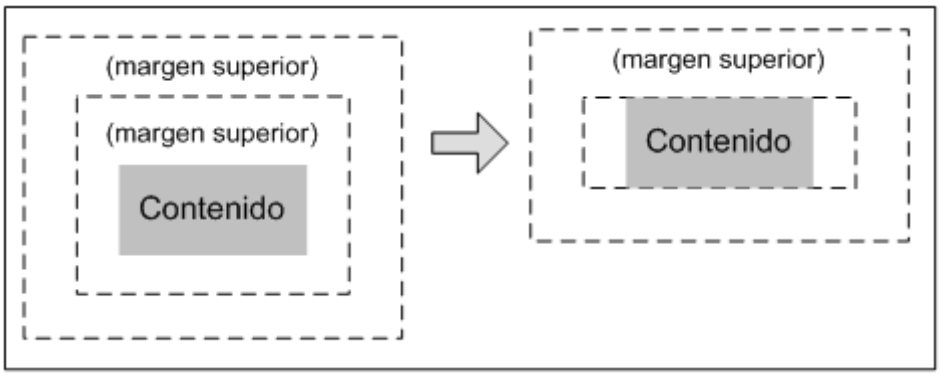

Figura 3.8 Fusión de los márgenes de los elementos interiores

Aunque en principio puede parecer un comportamiento extraño, la razón por la que se propuso este mecanismo de fusión automática de márgenes verticales es el de dar uniformidad a las páginas web habituales. En una página con varios párrafos, si no se diera este comportamiento y se estableciera un determinado margen a todos los párrafos, el primer párrafo no mostraría un aspecto homogéneo respecto de los demás.

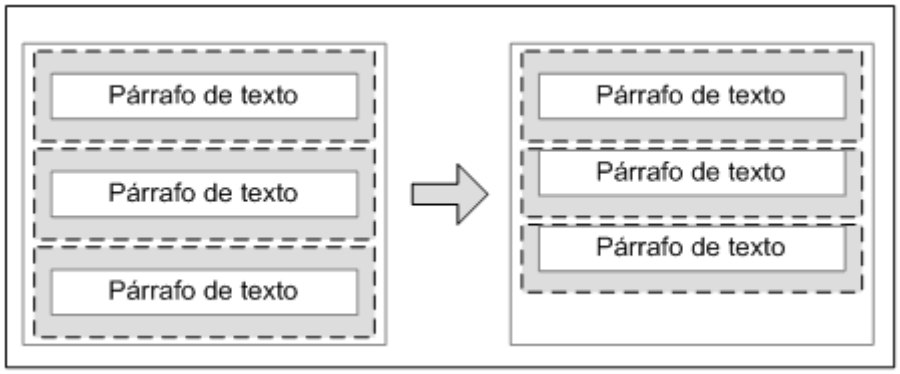

Figura 4.9 Motivo por el que se fusionan automáticamente los márgenes verticales

En el caso de un elemento que se encuentra en el interior de otro y sus márgenes se fusionan de forma automática, se puede evitar este comportamiento añadiendo un pequeño relleno (`padding: 1px`) o un borde (`border: 1px solid transparent`) al elemento contenedor.

3.2.2 Relleno

CSS define cuatro propiedades para controlar cada uno de los espacios de relleno horizontales y verticales de un elemento.

Propiedades	**padding-top, padding-right, padding-bottom, padding-left**
Valores	unidad de medida | porcentaje | inherit
Se aplica a	Todos los elementos excepto algunos elementos de tablas como grupos de cabeceras y grupos de pies de tabla
Valor inicial	0
Descripción	Establece cada uno de los rellenos horizontales y verticales de un elemento

Cada una de estas propiedades establece la separación entre el contenido y los bordes laterales de la caja del elemento:

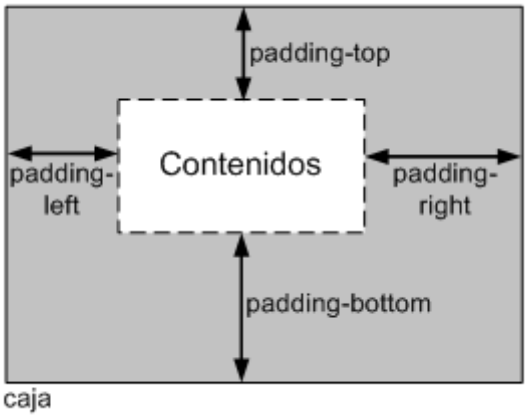

Figura 4.10 Las cuatro propiedades relacionadas con los rellenos

Como sucede con los márgenes, CSS también define una propiedad de tipo *"shorthand"* llamada `padding` para establecer los cuatro rellenos de un elemento de forma simultánea.

Propiedad	**padding**
Valores	(unidad de medida | porcentaje) {1, 4} | inherit
Se aplica a	Todos los elementos excepto algunos elementos de tablas como grupos de cabeceras y grupos de pies de tabla
Valor inicial	-
Descripción	Establece de forma directa todos los rellenos de los elementos

La notación `{1, 4}` de la definición anterior significa que la propiedad `padding` admite entre uno y cuatro valores, con el mismo significado que el de la propiedad `margin`. Ejemplo:

```
body {padding: 2em}     /* Todos los rellenos valen 2em */
body {padding: 1em 2em} /* Superior e inferior = 1em, Izquierdo y derecho = 2em
*/
body {padding: 1em 2em 3em} /* Superior = 1em, derecho = 2em, inferior = 3em,
izquierdo = 2em */
body {padding: 1em 2em 3em 4em} /* Superior = 1em, derecho = 2em, inferior = 3em,
izquierdo = 4em */
```

3.3 Bordes

CSS permite modificar el aspecto de cada uno de los cuatro **bordes de la caja** de un elemento. Para cada borde se puede establecer su anchura o grosor, su color y su estilo, por lo que en total CSS define 20 propiedades relacionadas con los bordes.

3.3.1 Anchura

La anchura de los bordes se controla con las cuatro propiedades siguientes:

Propiedades	**border-top-width, border-right-width, border-bottom-width, border-left-width**
Valores	(unidad de medida | thin | medium | thick) | inherit
Se aplica a	Todos los elementos
Valor inicial	Medium

Propiedades	**border-top-width, border-right-width, border-bottom-width, border-left-width**
Descripción	Establece la anchura de cada uno de los cuatro bordes de los elementos

La anchura de los bordes se indica mediante una medida o mediante las palabras clave `thin` (borde delgado), `medium` (borde normal) y `thick` (borde ancho).

La unidad de medida más habitual para establecer el grosor de los bordes es el píxel, ya que es la que permite un control más preciso sobre el grosor. Las palabras clave apenas se utilizan, ya que el estándar CSS no indica explícitamente el grosor al que equivale cada palabra clave, por lo que pueden producirse diferencias visuales entre navegadores. Así por ejemplo, el grosor `medium` equivale a `4px` en algunas versiones de Internet Explorer y a `3px` en el resto de navegadores.

El siguiente ejemplo muestra un elemento con cuatro anchuras diferentes de borde:

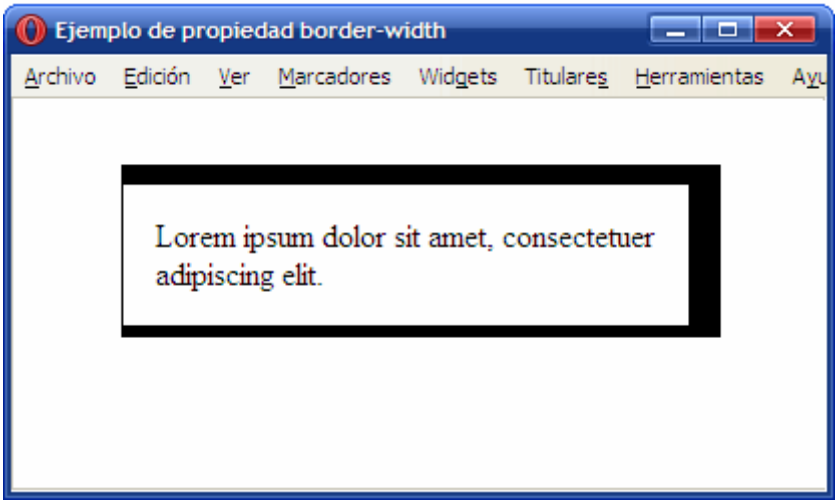

Figura 4.11 Ejemplo de propiedad border-width

Las reglas CSS utilizadas se muestran a continuación:

```
div {
  border-top-width: 10px;
  border-right-width: 1em;
  border-bottom-width: thick;
  border-left-width: thin;
}
```

Si se quiere establecer de forma simultánea la anchura de todos los bordes de una caja, es necesario utilizar una propiedad llamada `border-width`:

Propiedad	**border-width**
Valores	(unidad de medida | thin | medium | thick) {1, 4} | inherit
Se aplica a	Todos los elementos
Valor inicial	Medium
Descripción	Establece la anchura de todos los bordes del elemento

La propiedad `border-width` permite indicar entre uno y cuatro valores. El significado de cada caso es el habitual de las propiedades *"shorthand"*:

```
p { border-width: thin }                /* thin thin thin thin */
p { border-width: thin thick }          /* thin thick thin thick */
```

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

Si se indica un solo valor, se aplica a los cuatro bordes. Si se indican dos valores, el primero se aplica al borde superior e inferior y el segundo valor se aplica al borde izquierdo y derecho.

Si se indican tres valores, el primero se aplica al borde superior, el segundo se aplica al borde izquierdo y derecho y el tercer valor se aplica al borde inferior. Si se indican los cuatro valores, el orden de aplicación es superior, derecho, inferior e izquierdo.

3.3.2 Color

El color de los bordes se controla con las cuatro propiedades siguientes:

Propiedades	**border-top-color, border-right-color, border-bottom-color, border-left-color**
Valores	color | transparent | inherit
Se aplica a	Todos los elementos
Valor inicial	-
Descripción	Establece el color de cada uno de los cuatro bordes de los elementos

El ejemplo anterior se puede modificar para mostrar cada uno de los bordes de un color diferente:

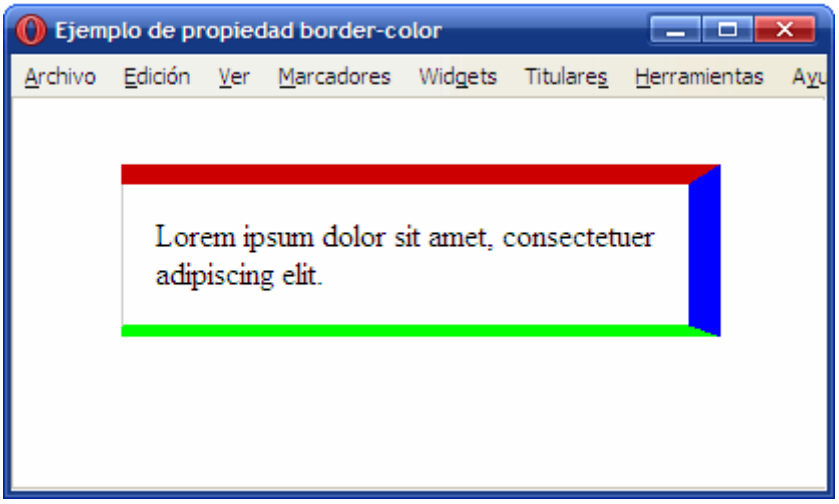

Figura 3.12 Ejemplo de propiedad border-color

Las reglas CSS necesarias para mostrar los colores anteriores son las siguientes:

```
div {
  border-top-color: #CC0000;
  border-right-color: blue;
  border-bottom-color: #00FF00;
  border-left-color: #CCC;
}
```

CSS incluye una propiedad llamada `border-color` para establecer de forma simultánea el color de todos los bordes de una caja:

Propiedad	**border-color**
Valores	(color | transparent) {1, 4} | inherit

Propiedad	**border-color**
Se aplica a	Todos los elementos
Valor inicial	-
Descripción	Establece el color de todos los bordes del elemento

En este caso, al igual que sucede con la propiedad `border-width`, es posible indicar de uno a cuatro valores y las reglas de aplicación son idénticas a las de la propiedad `border-width`.

3.3.3 Estilo

Por último, CSS permite establecer el estilo de cada uno de los bordes mediante las siguientes propiedades:

Propiedades	**border-top-style, border-right-style, border-bottom-style, border-left-style**
Valores	none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset | inherit
Se aplica a	Todos los elementos
Valor inicial	none
Descripción	Establece el estilo de cada uno de los cuatro bordes de los elementos

El estilo de los bordes sólo se puede indicar mediante alguna de las palabras reservadas definidas por CSS. Como el valor por defecto de esta propiedad es `none`, los elementos no muestran ningún borde visible a menos que se establezca explícitamente un estilo de borde.

Siguiendo el ejemplo anterior, se puede modificar el estilo de cada uno de los bordes:

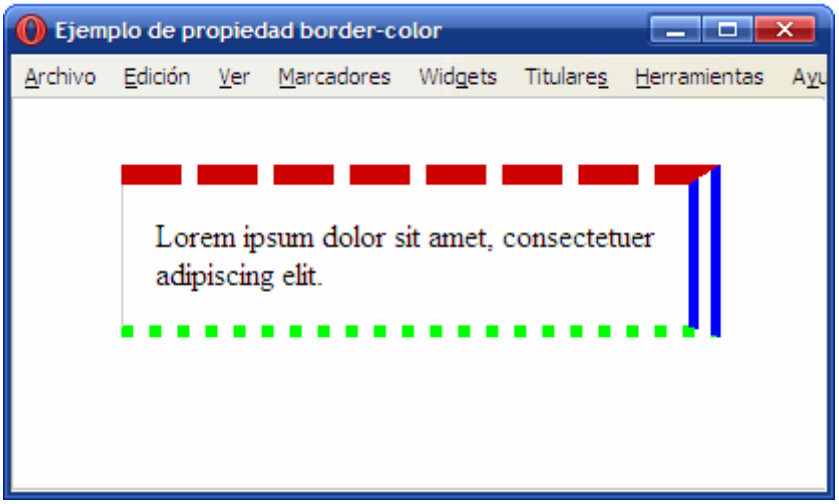

Figura 4.13 Ejemplo de propiedad border-style

Las reglas CSS necesarias para mostrar los estilos anteriores son las siguientes:

```
div {
  border-top-style: dashed;
  border-right-style: double;
  border-bottom-style: dotted;
  border-left-style: solid;
}
```

El aspecto con el que los navegadores muestran los diferentes tipos de borde se muestra a continuación:

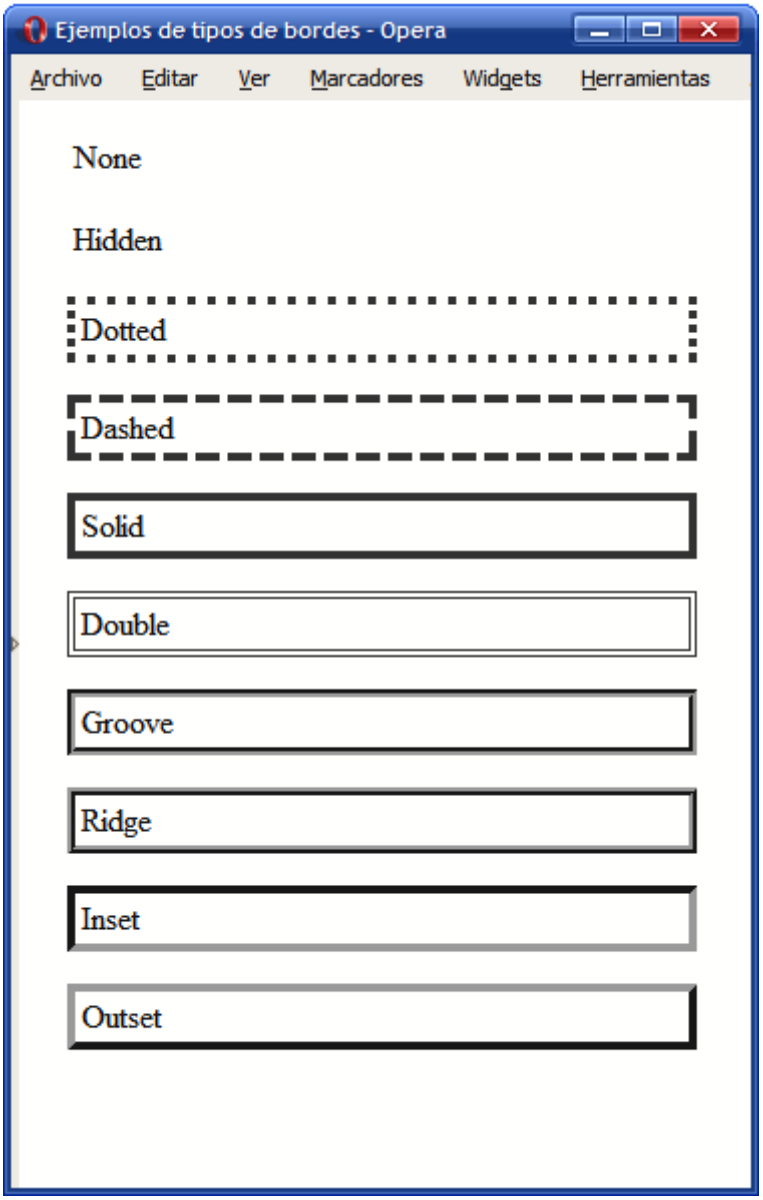

Figura 3.14 Tipos de bordes definidos por CSS

Los bordes más utilizados son `solid` y `dashed`, seguidos de `double` y `dotted`. Los estilos `none` y `hidden` son idénticos visualmente, pero se diferencian en la forma que los navegadores resuelven los conflictos entre los bordes de las celdas adyacentes en las tablas.

Para establecer de forma simultánea los estilos de todos los bordes de una caja, es necesario utilizar la propiedad llamada `border-style`:

Propiedad	**border-style**
Valores	(none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset) {1, 4} | inherit
Se aplica a	Todos los elementos
Valor inicial	-
Descripción	Establece el estilo de todos los bordes del elemento

Como es habitual, la propiedad permite indicar de uno a cuatro valores diferentes.

3.3.4 Propiedades shorthand

Como sucede con los márgenes y los rellenos, CSS define una serie de propiedades de tipo *"shorthand"* que permiten establecer todos los atributos de los bordes de forma simultánea. CSS incluye una propiedad *"shorthand"* para cada uno de los cuatro bordes y una propiedad *"shorthand"* global.

Propiedades	**border-top, border-right, border-bottom, border-left**
Valores	(unidad de medida_borde || color_borde || estilo_borde) | inherit
Se aplica a	Todos los elementos
Valor inicial	-
Descripción	Establece el estilo completo de cada uno de los cuatro bordes de los elementos

El significado de cada uno de los valores especiales es el siguiente:

- `<medida_borde>`: una medida CSS o alguna de las siguientes palabras clave: `thin`, `medium`, `thick`.
- `<color_borde>`: un color de CSS o la palabra clave `transparent`
- `<estilo_borde>`: una de las siguientes palabras clave: `none`, `hidden`, `dotted`, `dashed`, `solid`, `double`, `groove`, `ridge`, `inset`, `outset`.

Las propiedades *"shorthand"* permiten establecer alguno o todos los atributos de cada borde. El siguiente ejemplo establece el color y el tipo del borde inferior, pero no su anchura:

```
h1 {
  border-bottom: solid red;
}
```

En el ejemplo anterior, la anchura del borde será la correspondiente al valor por defecto (`medium`). Este otro ejemplo muestra la forma habitual utilizada para establecer el estilo de cada borde:

```
div {
  border-top: 1px solid #369;
  border-bottom: 3px double #369;
}
```

Por ultimo, CSS define una propiedad de tipo *"shorthand"* global para establecer el valor de todos los atributos de todos los bordes de forma directa:

Propiedad	**border**
Valores	(unidad de medida_borde || color_borde || estilo_borde) | inherit
Se aplica a	Todos los elementos
Valor inicial	-
Descripción	Establece el estilo completo de todos los bordes de los elementos

Las siguientes reglas CSS son equivalentes:

```
div {
  border-top: 1px solid red;
  border-right: 1px solid red;
```

```
  border-bottom: 1px solid red;
  border-left: 1px solid red;
}

div { border: 1px solid red; }
```

Como el valor por defecto de la propiedad `border-style` es `none`, si una propiedad *shorthand* no establece explícitamente el estilo de un borde, el elemento no muestra ese borde:

```
/* Sólo se establece el color, por lo que el estilo es
    "none" y el borde no se muestra */
div { border: red; }

/* Se establece el grosor y el color del borde, pero no
   su estilo, por lo que es "none" y el borde no se muestra */
div { border-bottom: 5px blue; }
```

Cuando los cuatro bordes no son idénticos pero sí muy parecidos, se puede utilizar la propiedad `border` para establecer de forma directa los atributos comunes de todos los bordes y posteriormente especificar para cada uno de los cuatro bordes sus propiedades particulares:

```
h1 {
  border: solid #000;
  border-top-width: 6px;
  border-left-width: 8px;
}
```

3.4 Margen, relleno, bordes y modelo de cajas

La anchura y altura de un elemento no solamente se calculan teniendo en cuenta sus propiedades width y height. El margen, el relleno y los bordes establecidos a un elemento determinan la anchura y altura final del elemento. En el siguiente ejemplo se muestran los estilos CSS de un elemento:

```
div {
  width: 300px;
  padding-left:  50px;
  padding-right: 50px;
  margin-left:   30px;
  margin-right:  30px;
  border: 10px solid black;
}
```

La anchura total con la que se muestra el elemento no son los `300` píxel indicados en la propiedad `width`, sino que también **se añaden todos sus márgenes, rellenos y bordes**:

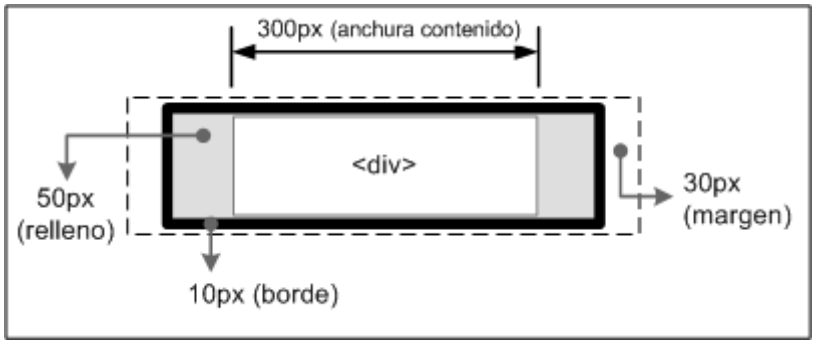

Figura 3.15 La anchura total de un elemento tiene en cuenta los márgenes, rellenos y bordes

De esta forma, la anchura del elemento en pantalla sería igual a la suma de los márgenes, los bordes, los rellenos y la anchura original:

```
30px + 10px + 50px + 300px + 50px + 10px + 30px = 480 píxeles
```

Así, la anchura/altura establecida con CSS siempre hace referencia a la anchura/altura del contenido. La anchura/altura total del elemento debe tener en cuenta además los valores del resto de partes que componen la caja del *box model*.

3.5 Fondos

El último elemento que forma el ***box model*** es el fondo de la caja del elemento. El fondo puede ser un color simple o una imagen. El fondo solamente se visualiza en el área ocupada por el contenido y su relleno, ya que el color de los bordes se controla directamente desde los bordes y las zonas de los márgenes siempre son transparentes.

Para establecer un color o imagen de fondo en la página entera, se debe establecer un fondo al elemento `<body>`. Si se establece un fondo a la página, como el valor inicial del fondo de los elementos es transparente, todos los elementos de la página se visualizan con el mismo fondo a menos que algún elemento especifique su propio fondo.

CSS define cinco propiedades para establecer el fondo de cada elemento (`background-color`, `background-image`, `background-repeat`, `background-attachment`, `background-position`) y otra propiedad de tipo *"shorthand"* (`background`).

La propiedad **`background-color`** permite mostrar un color de fondo sólido en la caja de un elemento. Esta propiedad no permite crear degradados ni ningún otro efecto avanzado.

Propiedad	**background-color**
Valores	color | transparent | inherit
Se aplica a	Todos los elementos
Valor inicial	transparent
Descripción	Establece un color de fondo para los elementos

El siguiente ejemplo muestra una página web con un color gris claro de fondo:

```
body {
  background-color: #F5F5F5;
}
```

Para crear efectos gráficos avanzados, es necesario utilizar la propiedad **`background-image`**, que permite mostrar una imagen como fondo de la caja de cualquier elemento:

Propiedad	**background-image**
Valores	url | none | inherit
Se aplica a	Todos los elementos
Valor inicial	none
Descripción	Establece una imagen como fondo para los elementos

CSS permite establecer de forma simultánea un color y una imagen de fondo. En este caso, la imagen se muestra delante del color, por lo que solamente si la imagen contiene zonas transparentes es posible ver el color de fondo.

El siguiente ejemplo muestra una imagen como fondo de toda la página:

```
body { background-image: url("imagenes/fondo.png") }
```

Las imágenes de fondo se indican a través de su URL, que puede ser absoluta o relativa. Suele ser recomendable crear una carpeta de imágenes que se encuentre en el mismo directorio que los archivos CSS y que almacene todas las imágenes utilizadas en el diseño de las páginas.

Así, las imágenes correspondientes al diseño de la página se mantienen separadas del resto de imágenes del sitio y el código CSS es más sencillo (por utilizar URL relativas) y más fácil de mantener (por no tener que actualizar URL absolutas en caso de que se cambie la estructura del sitio web).

Si la imagen que se quiere mostrar es demasiado grande para el fondo del elemento, solamente se muestra la parte de imagen comprendida en el tamaño del elemento. Si la imagen es más pequeña que el elemento, CSS la repite horizontal y verticalmente hasta llenar el fondo del elemento.

Este comportamiento es útil para establecer un fondo complejo a una página web entera. El siguiente ejemplo utiliza una imagen muy pequeña para establecer un fondo complejo a toda una página:

Imagen original

Figura 3.18 Imagen original utilizada para el fondo de la página

Reglas CSS

```
body {
  background-image:url(imagenes/fondo.gif);
}
```

Resultado

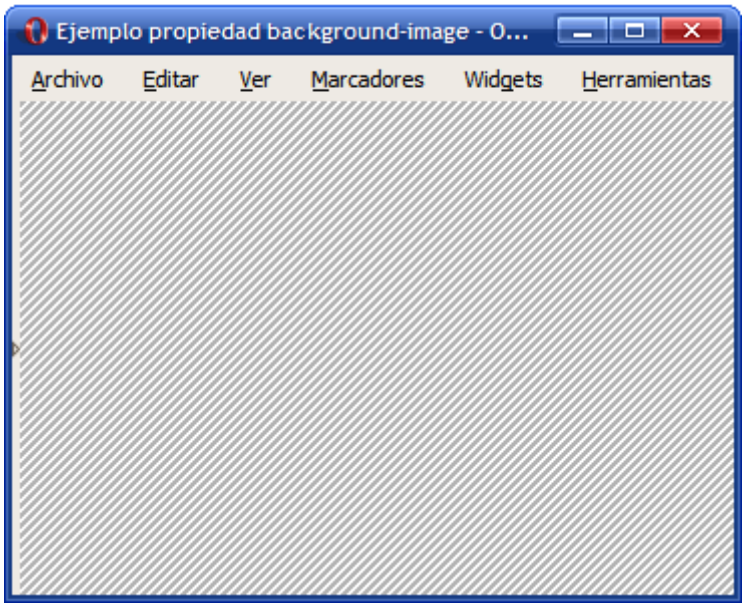

Figura 3.19 Página con una imagen de fondo

En ocasiones, no es conveniente que la imagen de fondo se repita horizontal y verticalmente. Para ello, CSS introduce la propiedad **`background-repeat`** que permite controlar la forma de repetición de las imágenes de fondo.

Propiedad	**background-repeat**
Valores	repeat | repeat-x | repeat-y | no-repeat | inherit
Se aplica a	Todos los elementos
Valor inicial	repeat
Descripción	Controla la forma en la que se repiten las imágenes de fondo

El valor `repeat` indica que la imagen se debe repetir en todas direcciones y por tanto, es el comportamiento por defecto. El valor `no-repeat` muestra una sola vez la imagen y no se repite en ninguna dirección. El valor `repeat-x` repite la imagen sólo horizontalmente y el valor `repeat-y` repite la imagen solamente de forma vertical.

La siguiente web utiliza el valor `repeat-x` para mostrar una imagen de fondo gris en la cabecera de la página:

Figura 3.20 Uso de repeat-x en la página de Kottke.org

Las reglas CSS definidas para la cabecera son:

```
#hdr {
  background: url("/images/ds.gif") repeat-x;
  width: 100%;
  text-align: center;
}
```

Por otra parte, el siguiente sitio utiliza el valor `repeat-y` para mostrar el fondo de una columna de contenidos en color gris:

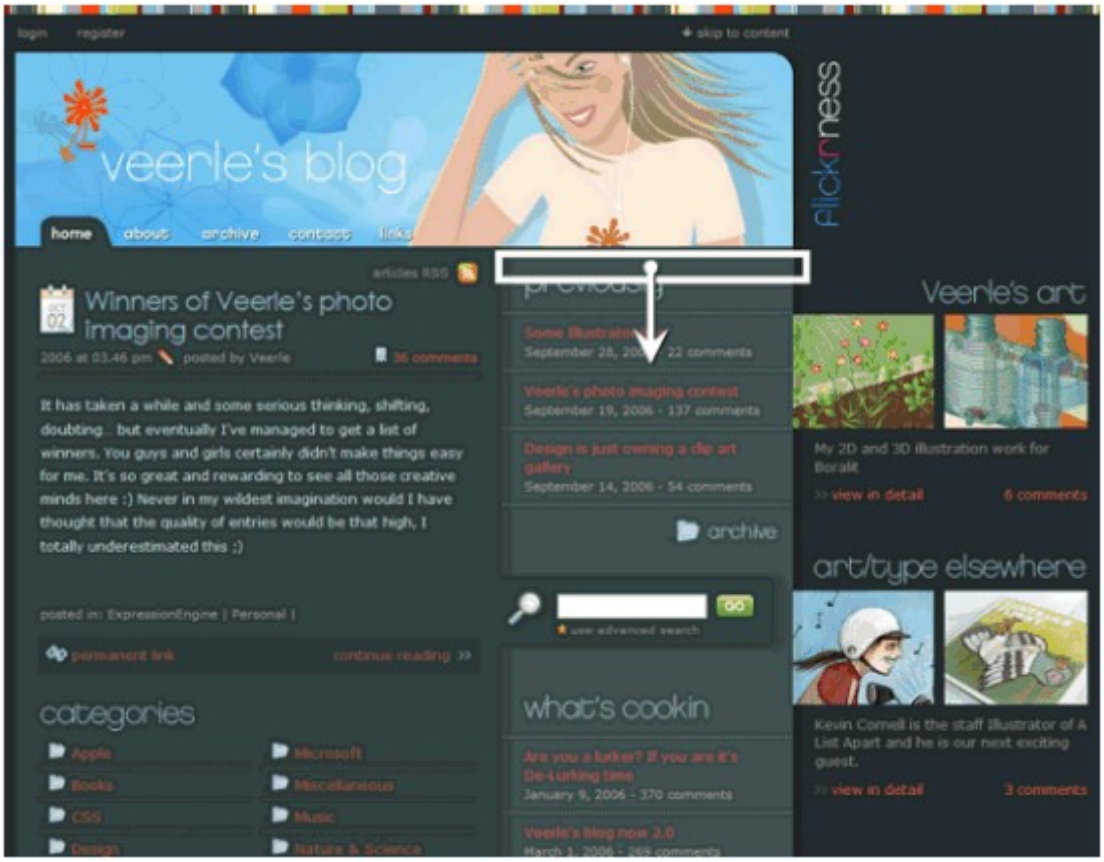

Figura 3.21 Uso de repeat-y en la página de Veerle.duoh.com

Las reglas CSS definidas para esa columna de contenidos son:

```
.wide #content-secondary {
  width: 272px;
  margin: 13px 0 0 0;
  position: relative;
  margin-left: -8px;
  background: url("./graphics/wide/bg-content-secondary.gif") repeat-y;
}
```

Además de seleccionar el tipo de repetición de las imágenes de fondo, CSS permite controlar la posición de la imagen dentro del fondo del elemento mediante la propiedad `background-position`.

Propiedad	**background-position**
Valores	((porcentaje | unidad de medida | left | center | right) (porcentaje | unidad de medida | top | center | bottom)?) | ((left | center | right) || (top | center | bottom)) | inherit
Se aplica a	Todos los elementos
Valor inicial	0% 0%
Descripción	Controla la posición en la que se muestra la imagen en el fondo del elemento

La propiedad **`background-position`** permite indicar la distancia que se desplaza la imagen de fondo respecto de su posición original situada en la esquina superior izquierda.

Si se indican **dos porcentajes o dos medidas**, el primero indica el desplazamiento **horizontal** y el

segundo el desplazamiento **vertical** respecto del origen (situado en la esquina superior izquierda). Si solamente se indica un porcentaje o una medida, se considera que es el desplazamiento horizontal y al desplazamiento vertical se le asigna automáticamente el valor de 50%.

Cuando se utilizan porcentajes, su interpretación no es intuitiva. Si el valor de la propiedad `background-position` se indica mediante dos porcentajes `x% y%`, el navegador coloca el punto `(x%, y%)` de la imagen de fondo en el punto `(x%, y%)` del elemento.

Las palabras clave permitidas son equivalentes a algunos porcentajes significativos: **`top`** = 0%, **`left`** = 0%, **`center`** = 50%, **`bottom`** = 100%, **`right`** = 100%.

CSS permite mezclar porcentajes y palabras clave, como por ejemplo `50% 2cm`, `center 2cm`, `center 10%`.

Si se utilizan solamente palabras clave, el orden es indiferente y por tanto, es equivalente indicar `top left` y `left top`.

El siguiente ejemplo muestra una misma imagen de fondo posicionada de tres formas diferentes:

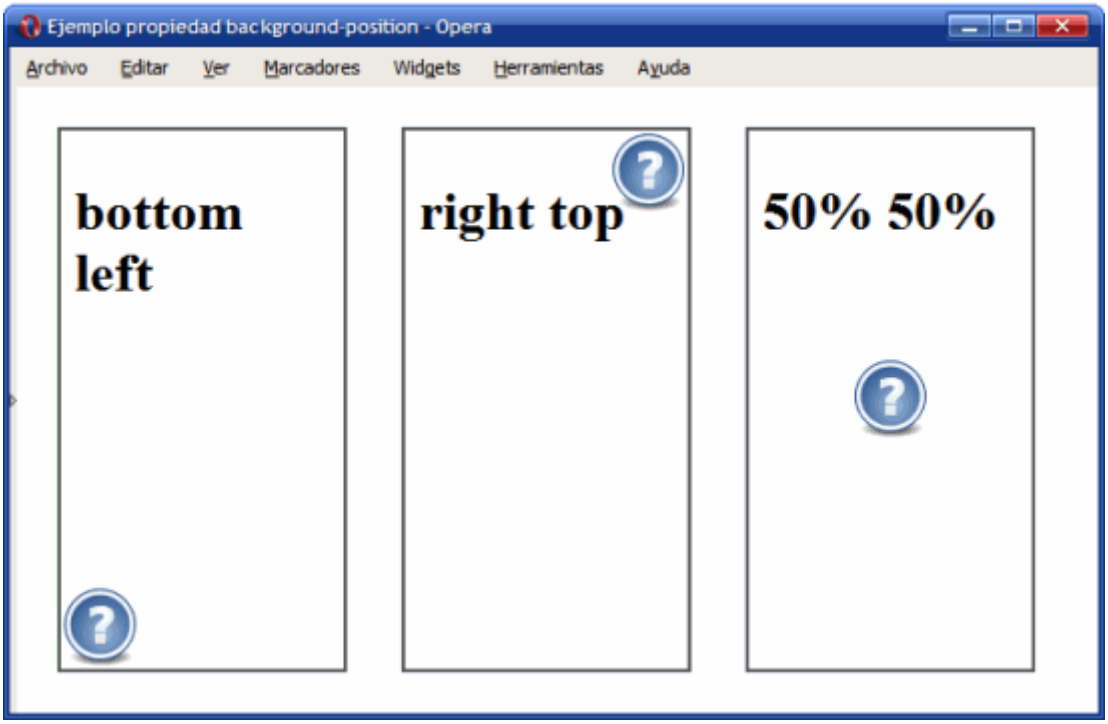

Figura 3.22 Ejemplo de propiedad background-position

Las reglas CSS del ejemplo anterior se muestran a continuación:

```
#caja1 {
  background-image: url("images/help.png");
  background-repeat: no-repeat;
  background-position: bottom left; //Abajo izquierda
}
#caja2 {
  background-image: url("images/help.png");
  background-repeat: no-repeat;
  background-position: right top; //Derecha Arriba
}
#caja3 {
```

```
  background-image: url("images/help.png");
  background-repeat: no-repeat;
  background-position: 50% 50%;
}

<div id="caja1"><h1>bottom left</h1></div>
<div id="caja2"><h1>right top</h1></div>
<div id="caja3"><h1>50% 50%</h1></div>
```

Opcionalmente, se puede indicar que el fondo permanezca fijo cuando la ventana del navegador se desplaza mediante las barras de *scroll*. Se trata de un comportamiento que en general no es deseable y que algunos navegadores no soportan correctamente. La propiedad que controla este comportamiento es `background-attachment`.

Propiedad	**background-attachment**
Valores	scroll | fixed | inherit
Se aplica a	Todos los elementos
Valor inicial	scroll
Descripción	Controla la forma en la que se visualiza la imagen de fondo: permanece fija cuando se hace scroll en la ventana del navegador o se desplaza junto con la ventana

Para hacer que una imagen de fondo se muestre fija al desplazar la ventana del navegador, se debe añadir la propiedad `background-attachment: fixed`.

Por último, CSS define una propiedad de tipo *"shorthand"* para indicar todas las propiedades de los colores e imágenes de fondo de forma directa. La propiedad se denomina **`background`** y es la que generalmente se utiliza para establecer las propiedades del fondo de los elementos.

Propiedad	**background**
Valores	(background-color || background-image || background-repeat || background-attachment || background-position) | inherit
Se aplica a	Todos los elementos
Valor inicial	-
Descripción	Establece todas las propiedades del fondo de un elemento

El orden en el que se indican las propiedades es indiferente, aunque en general se sigue el formato indicado de color, url de imagen, repetición y posición.

El siguiente ejemplo muestra la ventaja de utilizar la propiedad `background`:

```
/* Color e imagen de fondo de la página mediante una propiedad shorthand */
body {
      background: #222d2d url(./graphics/colorstrip.gif) repeat-x 0 0;
}

/* La propiedad shorthand anterior es equivalente a las siguientes propiedades */
body {
  background-color: #222d2d;
  background-image: url("./graphics/colorstrip.gif");
  background-repeat: repeat-x;
  background-position: 0 0;
}
```

4. PSEUDO-CLASES

Las pseudo-clases **nos permiten modificar una característica particular de una entidad**. El ejemplo mas común es modificar como se debe comportar un enlace al pasar el puntero por encima. Como vemos mas adelante, **podemos definir el elemento a para que tenga un color de fondo y que al pasar por encima el color de fondo cambie, esto podemos hacer con el uso de la pseudoclase :hover**

4.1 La pseudo-clase :**first-child**

La pseudo-clase :first-child selecciona el primer elemento hijo de un elemento. Si se considera el siguiente ejemplo:

```
p em:first-child {
  color: red;
}
```

```
<p>Lorem
      <span>
            <em>ipsum dolor</em>
      </span>
      sit amet, consectetuer adipiscing elit. Praesent odio sem, tempor quis,
      <em>auctor eu</em>,
      tempus at, enim. Praesent nulla ante, <em>ultricies</em> id, porttitor ut,
      pulvinar quis, dui.
</p>
```

El selector **p em:first-child** selecciona el primer elemento que sea hijo de un elemento y que se encuentre dentro de un elemento <p>. Por tanto, en el ejemplo anterior **sólo el primer se ve de color rojo.**

La pseudo-clase :first-child también se puede utilizar en los selectores simples, como se muestra a continuación:

```
p:first-child { ... }
```

La regla CSS anterior aplica sus estilos al primer párrafo de cualquier elemento.

4.2 Las pseudo-clases **:link y :visited**

Las pseudo-clases :link y :visited se pueden utilizar para aplicar diferentes estilos a los enlaces de una misma página:

- La pseudo-clase :link se aplica a todos los enlaces que todavía no han sido visitados por el usuario.
- La pseudo-clase :visited se aplica a todos los enlaces que han sido visitados al menos una vez por el usuario.

El navegador gestiona de forma automática el cambio de enlace no visitado a enlace visitado.

Aunque el usuario puede borrar la cache y el historial de navegación de forma explícita, los navegadores también borran de forma periódica la lista de enlaces visitados.

Por su propia definición, las pseudo-clases :link y :visited son mutuamente excluyentes, de forma que un mismo enlace no puede estar en los dos estados de forma simultánea.

Como los navegadores muestran por defecto los enlaces de color azul y los enlaces visitados de color morado, es habitual modificar los estilos para adaptarlos a la guía de estilo del sitio web:

```
a:link    { color: red;   }
a:visited { color: green; }
```

4.3 Las pseudo-clases **:hover, :active y :focus**

Las pseudo-clases :hover, :active y :focus permiten al diseñador web variar los estilos de un elemento en respuesta a las acciones del usuario. Al contrario que las pseudo-clases :link y :visited que sólo se pueden aplicar a los enlaces, estas pseudo-clases se pueden aplicar a cualquier elemento.

A continuación se indican las acciones del usuario que activan cada pseudo-clase:

- :hover, se activa cuando el usuario pasa el ratón o cualquier otro elemento apuntador por encima de un elemento.
- :active, se activa cuando el usuario activa un elemento, por ejemplo cuando pulsa con el ratón sobre un elemento. El estilo se aplica durante un espacio de tiempo prácticamente imperceptible, ya que sólo dura desde que el usuario pulsa el botón del ratón hasta que lo suelta.
- :focus, se activa cuando el elemento tiene el foco del navegador, es decir, cuando el elemento está seleccionado. Normalmente se aplica a los elementos <input> de los formularios cuando están activados y por tanto, se puede escribir directamente en esos campos.

De las definiciones anteriores se desprende que un mismo elemento puede verse afectado por varias pseudo-clases diferentes de forma simultánea.

Debido a esta característica y al comportamiento en cascada de los estilos CSS, es importante cuidar el orden en el que se establecen las diferentes pseudo-clases. El siguiente ejemplo muestra el único orden correcto para establecer las cuatro pseudo-clases principales en un enlace:

```
a:link    { ... }
a:visited { ... }
a:hover   { ... }
a:active  { ... }
```

5. BIBLIOGRAFÍA

http://librosweb.es/libro/css/