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.

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.

Unidades relativas

La 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 `<em>` 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; }
```

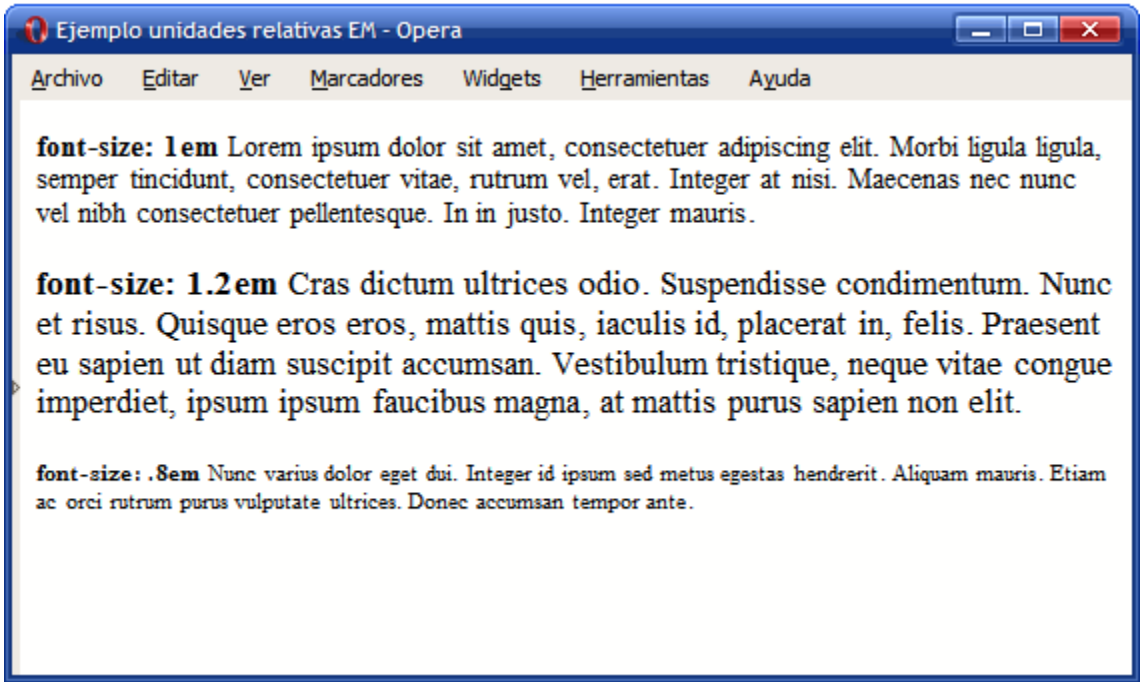
Como `em` es una unidad relativa, el valor `0.9` indicado sólo tiene sentido cuando se tiene en consideración su referencia. Para la unidad `em`, la referencia es el tamaño de letra por defecto del sistema (ordenador, dispositivo móvil, etc.) del usuario.

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 x 12pt = 10.8pt`.

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:



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.

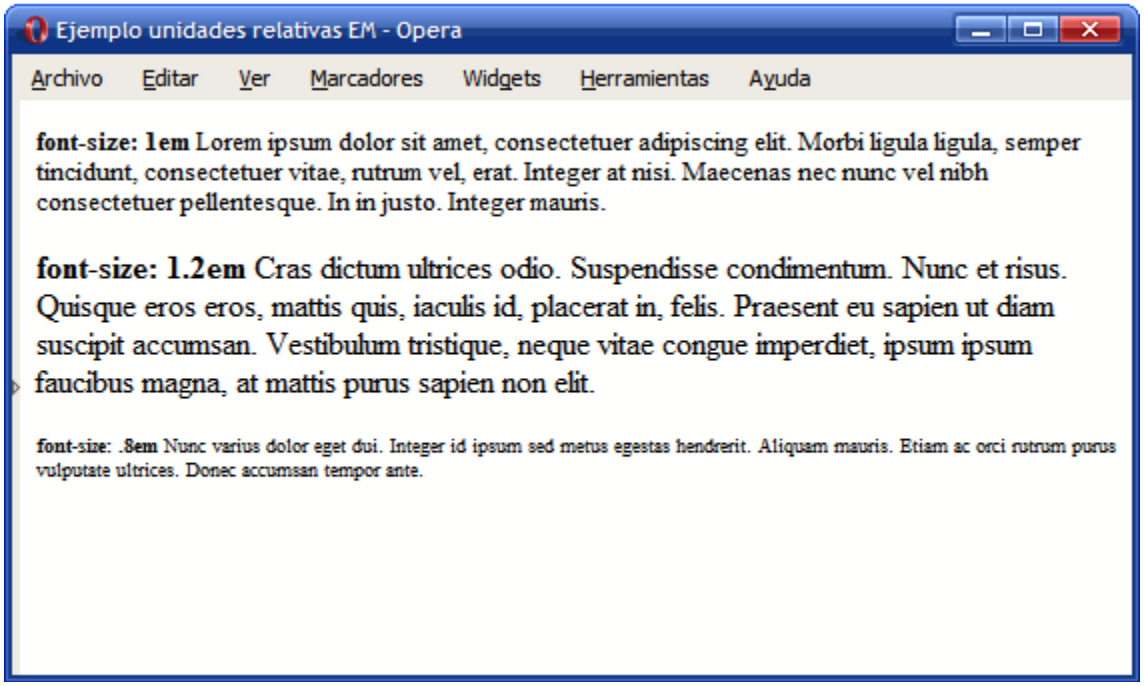
Cuando se estudian por primera vez, las unidades relativas parecen demasiado complicadas. Al fin y al cabo, siempre se debe tomar la referencia de la unidad para obtener su valor real. Sin embargo, sus ventajas son mucho mayores que sus inconvenientes.

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:



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 x 10px = 25px`.

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.

Sin embargo, las medidas relativas no se heredan directamente, sino que se heredan sus valores reales una vez calculados. El siguiente ejemplo muestra este comportamiento:

```
body {
  font-size: 12px;
  text-indent: 3em;
}
h1 { font-size: 15px }
```

La propiedad `text-indent`, como se verá en los próximos capítulos, se utiliza para tabular la primera línea de un texto. El elemento `<body>` define un valor para esta propiedad, pero el elemento `<h1>` no lo hace, por lo que heredará el valor de su elemento padre. Sin embargo, el valor heredado no es `3em`, sino `36px`.

Si se heredara el valor `3em`, al multiplicarlo por el valor de `font-size` del elemento `<h1>` (que vale `15px`) el resultado sería `3em x 15px = 45px`. No obstante, como se ha comentado, los valores que se heredan no son los relativos, sino los valores ya calculados.

Por lo tanto, en primer lugar se calcula el valor real de `3em` para el elemento `<body>`: `3em x 12px = 36px`. Una vez calculado el valor real, este es el valor que se hereda para el resto de elementos.

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).

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`.

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.

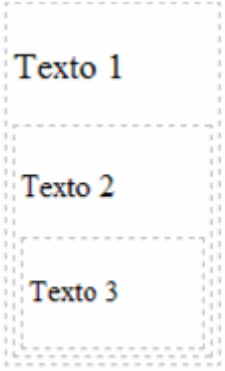
El documento *"Recomendaciones sobre técnicas CSS para la mejora de la accesibilidad de los contenidos HTML"* (http://www.w3.org/TR/WCAG10-CSS-TECHS/) elaborado por el organismo W3C, recomienda el uso de la unidad `em` para indicar el tamaño del texto y para todas las medidas que sean posibles.

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>
```



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 x 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 x 0.9em x 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.

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.

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
```

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 http://en.wikipedia.org/wiki/Websafe.

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`.

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%`.

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*. Cuando realizamos operaciones matemáticas, siempre utilizamos 10 símbolos para representar los números (del `0` al `9`). Por este motivo, se dice que utilizamos un sistema numérico decimal.

No obstante, el sistema decimal es solamente uno de los muchos sistemas numéricos que se han definido. Entre los sistemas numéricos alternativos más utilizados se encuentra el sistema hexadecimal, que utiliza 16 símbolos para representar sus números.

Como sólo conocemos 10 símbolos numéricos, el sistema hexadecimal utiliza también seis letras (de la `A` a la `F`) para representar los números. De esta forma, en el sistema hexadecimal, después del `9` no va el `10`, sino la `A`. La letra `B` equivale al número `11`, la `C` al `12`, la `D` al `13`, la `E` al `14` y la `F` al número `15`.

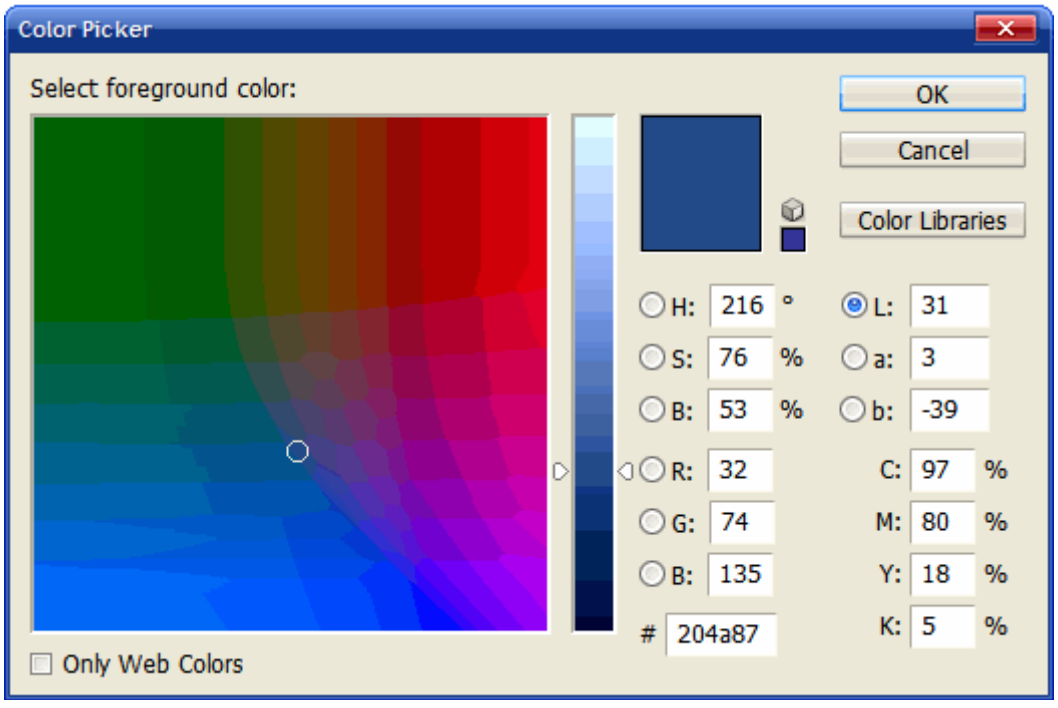
Definir un color en CSS con el método RGB hexadecimal requiere realizar los siguientes pasos:

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:



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.

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.

Existen varios colores definidos, como por ejemplo `ActiveBorder`, que hace referencia al color del borde de las ventanas activas. La lista completa de colores definidos se puede ver en http://www.w3.org/TR/CSS21/ui.html#system-colors.

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.

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 x 256 x 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. No obstante, el auge en el uso de los dispositivos móviles hace que siga siendo un tema a considerar, ya que las pantallas de muchos móviles sólo pueden representar un número reducido de colores.

La lista completa de colores web safe y sus valores hexadecimales se pueden consultar en http://en.wikipedia.org/wiki/Web_colors#Web-safe_colors.