

INTER
ACCIÓN
2004



HOJAS DE ESTILO Y USABILIDAD

Ferran Perdrix Sapiña



Grupo GRIHO (UdL)
C/Jaume II n° 69
ferran@griho.net



Índice

1. Introducción	3
1.1. Introducción a las Hojas de Estilo.	4
1.2. Beneficios de las Hojas de Estilo.	4
1.3. Inconvenientes de las Hojas de Estilo.	6
2. Usando las Hojas de Estilo	7
2.1. Incrustar las Hojas de Estilo dentro de las páginas Web.	7
2.2. Enlazar con una Hoja de Estilo desde un documento HTML.	8
2.3. Importar a un documento HTML una Hoja de Estilo externa.	9
2.4. Añadir estilos en los elementos de los documentos HTML.	10
2.5. Hojas de Estilo múltiples.	10
2.6. Asociación con XML.	11
3. Sintaxis básica	13
3.1. Agrupamiento.	13
3.2. Herencia.	14
3.3. Clase como selector.	15
3.4. ID como selector.	15
3.5. Selectores contextuales.	16
3.6. Comentarios.	17
3.7. Pseudo-clases y pseudo-elementos.	17
3.8. Diseño en cascada.	21
3.9. Modelo de formato.	24
4. Fuentes Descargables	33
4.1. Cómo usar fuentes descargables.	33
4.2. Herramientas de diseño.	34
5. Usabilidad y Accesibilidad con las Hojas de Estilo	34
6. Programas Asociados a las Hojas de Estilo	37
6.1. Editor: EDIPO.	37
6.2. Validador: W3C CSS Validator.	38
6.3. Navegadores: Compatibilidad con CSS.	39
7. Referencias	39
Apéndice A. Propiedades de CSS (Nivel 1)	
Apéndice B. Unidades	
Apéndice C. Gramáticas	
Apéndice D. Glosario de Términos	

1. Introducción

Construir páginas Web con HTML (como se define en [HTML32], [HTML40], [RFC1766], [RFC1866], [RFC1942] y [RFC2070]) es como pintar un cuadro con un pincel. Tan sólo los más perspicaces y tenaces pueden acertar con el resultado exacto que desean. Claramente no es la mejor herramienta para la precisión ni la flexibilidad.

Aquellos que hayan utilizado HTML con cierta asiduidad, habrán comprendido que no es una herramienta efectiva para crear páginas Web obligándonos, por ejemplo, a utilizar GIF's invisibles como espaciadores para colocar las cosas en el sitio que deseamos. Como consecuencia lógica nuestro código se complica, aumentan los recursos usados y el peso de las páginas con el consiguiente aumento de ancho de banda necesario y el retraso implícito en el tiempo de descarga: no es la solución más óptima posible.

Mejorando este escenario y bajo el nombre de HTML Dinámico [HTMLDIN] se engloba un conjunto de técnicas con dos objetivos claros: proporcionar un control absoluto al diseñador de páginas HTML y romper con el carácter estático de este tipo de documentos.

Los tres componentes del HTML Dinámico son:

- Hojas de Estilo
- Posicionamiento de Contenidos
- Fuentes Descargables

Veamos una pequeña definición de cada uno de estos componentes:

- a) Las hojas de estilo permiten especificar atributos para los elementos de su página Web. Las hojas de estilo nos permiten un mayor control sobre el aspecto de nuestros documentos. Con ellas podemos especificar muchos atributos tales como colores, márgenes, alineación de elementos, tipos y tamaños de letras, y muchos más. Podemos utilizar bordes para hacer que ciertos elementos resalten del resto de un documento. Podemos especificar que se utilicen diferentes fuentes para diferentes elementos tales como párrafos o cabeceras. Además podemos emplear hojas de estilo como patrones o páginas maestras de forma que múltiples páginas puedan tener el mismo aspecto. Las hojas de estilo pueden crearse empleando dos tipos de sintaxis, CSS (*Cascade Style Sheets*) [W3CSTYLE] y *JavaScript* [JAVAMAN], [JAVASUN].
- b) Con el posicionamiento de contenidos se puede asegurar que las diferentes partes serán mostradas exactamente donde se quiera que aparezcan y se podrá modificar su aspecto y posición tras ser mostrada. Ya no se está restringido al posicionamiento secuencial de los elementos sobre la página. Especificando la posición de los bloques podemos decidir donde se mostrara cada elemento en vez de dejar esta tarea al arbitrio del navegador. Utilizando JavaScript podemos cambiar el aspecto de la página dinámicamente. Podemos hacer que los elementos aparezcan o se desvanezcan, podemos cambiar su color y su posición, etc. Podemos añadir animaciones dentro de nuestras páginas moviendo y modificando ciertas partes de la misma. El uso conjunto las hojas de estilo y el posicionamiento de contenidos nos permite crear páginas Web que

utilicen diferentes estilos en diferentes partes de la página. Para el posicionamiento de contenidos también podemos utilizar dos tipos de sintaxis: CSS y JavaScript.

- c) Con las fuentes descargables podemos asegurar que siempre se utilizara la fuente correcta, pues podemos enviar la fuente adjunta con nuestra página. De esta forma se garantiza que las página siempre serán mostradas con la fuente que deseemos. Ya no es necesario emplear las fuentes genéricas para conseguir que las páginas tengan aproximadamente el mismo aspecto en todas las plataformas. Tampoco estaremos restringidos por las características propias de las fuentes en cada plataforma. Para proteger los derechos de autor de los diseñadores de fuentes, estas están protegidas de forma que es imposible que el usuario las copie y las pueda usar de nuevo. Así podemos incluir fuentes en nuestras páginas sin tener que preocuparnos de que los usuarios las puedan copiar.

1.1. Introducción a las Hojas de Estilo.

En el año 1996, las Hojas de Estilo entran en escena. Oficialmente llamadas *Cascading Style Sheets* (CSS), aparecen como un añadido al lenguaje HTML que entre otras cosas promete:

- Control más preciso que el existente sobre la presentación, fuentes, colores, fondos y otros efectos tipográficos.
- Un modo de mejorar la apariencia y formatear un gran número de páginas con tan sólo modificar un documento.
- Compatibilidad frente a los navegadores y las plataformas.
- Menos y más claro código, páginas más pequeñas y descargas más rápidas.

Netscape [NETSCAPE] e Internet Explorer [EXPLORER] añadieron etiquetas como ** y atributos como *color* a la especificación del lenguaje HTML original. Esto produjo una mayor complejidad del diseño de páginas Web al complicar la definición de las estructuras con el contenido de las mismas. Para solucionar este problema el W3C (*World Wide Web Consortium*) [W3C], consorcio sin ánimo de lucro encargado de la estandarización del lenguaje HTML, creó las Hojas de Estilo añadiéndolas en la versión HTML 4.0. De este modo desde las versiones Netscape 4.0 e Internet Explorer 4.0 se admite el uso de las CSS. Dado este apoyo que ofrecen la mayoría de los navegadores más usados, las Hojas de Estilo están realizando estas promesas siendo un pilar básico del uso del HTML Dinámico como un motor de cambio en el modo de crear las páginas Web.

1.2. Beneficios de las Hojas de Estilo.

Entonces, ¿qué es tan especial en las Hojas de Estilo?, ¿cómo nos pueden ayudar a crear páginas Web?, veamos los beneficios que nos plantea más en detalle:

Separación entre contenido y estructura.

HTML desde sus inicios no pretendió controlar el contenido o la apariencia de las páginas Web. Se trata de un lenguaje que define la estructura y la función de los elementos dentro de la página, dejando a los navegadores decidir cómo estos elementos deben ser mostrados al usuario final. La inclusión de nuevas etiquetas en el lenguaje HTML por parte de Netscape introdujo cierto control en la apariencia, pero a costa de complicar mucho el código necesario para crear o mover contenidos en las páginas Web. Las CSS nos permiten tener el control deseado separando el contenido de la parte que define la estructura. Así el código HTML permanece claro y simple, como originalmente pretendía.

Control más preciso de la apariencia.

La etiqueta `` nos permite escalar el texto, y las etiquetas de tabla nos ayudan a definir los márgenes. Pero a pesar de esto, lo que se puede hacer con HTML es muy limitado. No se puede crear texto con exactamente 80 píxeles de alto, no se pueden especificar márgenes de forma fácil, no se puede controlar el espacio entre líneas o palabras, no se pueden posicionar imágenes de forma precisa en la pantalla. Hasta ahora. Las Hojas de Estilo hacen todo esto posible y aporta muchas más posibilidades. Con HTML, el tamaño de la fuente se especifica con un sistema de medidas predeterminadas por el *browser* (en el ejemplo, `SIZE=5`), con las Hojas de Estilo hemos especificado el tamaño en puntos tipográficos (y podemos hacerlo en cm., píxeles, cuadratines, altura de la x, etc.). Más aún, las Hojas de Estilo permiten aplicar prácticamente todas las propiedades a cualquier elemento de la página, mientras que HTML sólo permite un número limitado de propiedades para cada elemento.

Páginas más pequeñas y ágiles.

Las Hojas de Estilo son texto llano, al igual que el lenguaje HTML. No hay gráficos, no hay programas ejecutables, no precisa *plug-ins* ni *streaming* de datos ni retrasos. Es tan rápido, como mínimo, como gestionar el código HTML. Además, con las Hojas de Estilo, se pueden definir aspectos que anteriormente eran GIFs, tablas u otros trucos anidados en el código. Por tanto, obtendremos unas páginas con menos código, menos gráficos lo que se traduce en ficheros de menor tamaño.

Gestión de cambios más rápido y fácil.

Las Hojas de Estilo con HTML 4.0 [HTML40] determinan cómo se visualizan los elementos HTML, exactamente igual como en HTML 3.2 [HTML32] se usaban la etiqueta `` o el atributo `color`. Estas Hojas de Estilo se guardan normalmente en documentos HTML externos permitiendo ahorrar mucho trabajo al poder actualizar la presentación de mi sitio Web con tan solo realizar cambios en un único documento. Así, las Hojas de Estilo suponen un avance en el modo de crear y modificar las páginas Web, habilitando la posibilidad de definir el estilo y la presentación de muchas páginas con un pequeño cambio. Un cambio global de apariencia es posible con la modificación de unas pocas líneas dentro del archivo que guarda la Hoja de Estilo a aplicar. Además reducimos las posibilidades de cometer errores.

Transparente respecto al navegador.

Contrariamente a otras tecnologías Web, el uso de las Hojas de Estilo es transparente a los usuarios. Éstos no aprecian nada al abrir la página con cualquier navegador. Simplemente si usan un navegador antiguo que no reconoce CSS ignora las Hojas de Estilo, mientras que si las soporta las usa para formatear el contenido.

Otras ventajas.

El lenguaje de las Hojas de Estilo, aunque muy potente, es relativamente sencillo y fácil de aprender.

Los documentos que usan Hojas de Estilo generalmente resultan más compactos.

Las Hojas de Estilo pueden aplicarse de varias maneras y combinarse formando una cascada de estilos con la información de cada una.

Pueden usarse con otros lenguajes de programación (como JavaScript) para conseguir efectos dinámicos en las páginas.

Se pueden especificar Hojas de Estilo para diferentes navegadores y tipos de medios (impresos, braille, auditivos, etc.).

El usuario con alguna discapacidad (o simplemente por preferencias) puede definir su propia Hoja de Estilo y la regla **!important** obliga a su navegador a suplantar la Hoja de Estilo del autor.

1.3. Inconvenientes de las Hojas de Estilo.

En cuanto a las desventajas en el uso de las Hojas de Estilo, la única de importancia es el soporte irregular que tienen las CSS por parte de los navegadores. Ciertas propiedades que funcionan en un *browser* no funcionan en otros, o existen diferencias en un mismo navegador según sea para Windows, Mac u otros. También existen diferencias entre distintas versiones de un mismo browser. Esto puede provocar que:

- Nuestra página sea visualizada por el lector con un formato no deseado por nosotros. En todo caso, el navegador aplicará el formato predeterminado y nuestro trabajo de composición habrá sido inútil.
- Algunas propiedades de las CSS (como las que afectan la posición o visibilidad de los elementos) pueden provocar que una parte del contenido de nuestra página resulte inaccesible desde ciertos navegadores si no son utilizadas correctamente.

Debe entenderse que las Hojas de Estilo fueron diseñadas para permitir que los autores *influyan* en la composición de la página, pero no para que la controlen. Una CSS *sugiere* al browser un estilo de composición para el documento pero no puede forzarlo a aplicar un formato determinado.

Las Hojas de Estilo son una herramienta que puede resultar muy efectiva para lograr una presentación atractiva de la página siempre que la página no sea dependiente de la Hoja de Estilo. Se debe considerar en todo momento aquellos navegadores que no soportan CSS, cuidando que los mismos puedan mostrar la página correctamente y en su totalidad aún cuando nuestras reglas de estilo no sean aplicadas.

2. Usando las Hojas de Estilo

Actualmente hay 4 métodos para poder usar las Hojas de Estilo asociadas a las páginas Web, cada una con sus ventajas e inconvenientes:

- Incrustar las Hojas de Estilo dentro de las páginas Web.
- Enlazar con una Hoja de Estilo desde un documento HTML.
- Importar a un documento HTML una Hoja de Estilo externa.
- Añadir estilos en los elementos de los documentos HTML.

Debemos recordar que es posible usar más de uno de estos métodos en cada página Web. De hecho, la potencia de las Hojas de Estilo radica, en gran modo, en la forma como se combinan estos estilos y se añaden en cascada dentro de cada página Web.

2.1. Incrustar Hojas de Estilo dentro de las páginas Web.

Usando este método toda la información acerca del estilo de la página se sitúa en la sección <HEAD> del código HTML, separándolo de la sección <BODY>. Este método debería ser usado cuando un único documento usa un único estilo. A continuación se muestra un ejemplo de cómo quedaría la definición de la Hoja de Estilo declarada en la sección <HEAD> de un documento HTML:

```
<HTML>
<HEAD>
<TITLE>Mi Primera Hoja de Estilo</TITLE>
<STYLE TYPE="text/css">
<!--
H1 { color: green; font-family: impact }
P { background: yellow; font-family: courier }
-->
</STYLE>
</HEAD>
<BODY>
<H1>Hojas de Estilo: Tutorial CSS</H1>
<P>Ferran Perdrix Sapiña, 2004</P>
</BODY>
</HTML>
```

Si nos fijamos en el ejemplo, probablemente observaremos dos curiosidades en el código: el atributo `TYPE="text/css"` [RFC2318] y las etiquetas de comentarios. `TYPE="text/css"` especifica el tipo MIME [RFC2045], así los navegadores que no soportan las Hojas de Estilo o CSS pueden ignorar el código que define el estilo. Se debe usar esta especificación previa.

Las etiquetas de comentarios (`<!--` y `-->`) son también muy importantes. Algunos navegadores viejos (como el Internet Explorer 2.0 para Mac) no pueden reconocer las etiquetas de estilo a pesar del atributo `TYPE="text/css"` y visualizan el código de las etiquetas en la página Web que ven los usuarios. Este es un efecto no deseado que se puede evitar usando las etiquetas de comentarios.

2.2. Enlazar con una Hoja de Estilo desde un documento HTML.

Este método pone de manifiesto la potencia de las Hojas de Estilo. Además de poder incluir código de estilo en una página Web, también se puede usar en múltiples documentos HTML una única declaración de Hoja de Estilo. Este estilo se aplicará a todas las páginas que le referencien y un cambio en este estilo se verá automáticamente reflejado en las páginas Web afectadas. Así pues, es indicado cuando se desea gobernar el estilo de muchas páginas a la vez.

Así es como trabaja: se crea la página Web de forma normal pero en lugar de usar la etiqueta `<STYLE>`, usaremos la etiqueta `<LINK>` en la sección `<HEAD>`, como en el siguiente ejemplo:

```
<HTML>
<HEAD>
<TITLE> Mi Primera Hoja de Estilo </TITLE>
<LINK REL=stylesheet HREF="miestilo.css" TYPE="text/css">
</HEAD>
<BODY>
<H1>Hojas de Estilo: Tutorial CSS</H1>
<P>Ferran Perdrix Sapiña, 2004</P>
</BODY>
</HTML>
```

Notar que en este caso no hace falta usar las etiquetas de comentario. Además se pueden usar URLs relativas o absolutas en el atributo `HREF` (véase [RFC1630], [RFC1738], [RFC1808], [RFC2396]).

Por otro lado, se debe crear un documento llamado `miestilo.css` que por ejemplo contenga el siguiente código:

```
H1 { color: green; font-family: impact }
P { background: yellow; font-family: courier }
```

Este archivo debe tener extensión `.css` y puede ser escrito en cualquier editor de texto.

2.3. Importar a un documento HTML una Hoja de Estilo externa.

Importar una Hoja de Estilo externa funciona de forma similar a enlazarla. La diferencia estriba en que no se permite combinar el uso de Hojas de Estilo enlazadas con otros métodos, mientras que si que es posible hacerlo usando la importación de Hojas de Estilo externas. Veamos un ejemplo:

```
<HTML>
<HEAD>
<TITLE> Mi Primera Hoja de Estilo </TITLE>
<STYLE TYPE="text/css">
<!--
@import url(company.css);
H1 { color: orange; font-family: impact }
-->
</STYLE>
</HEAD>
<BODY>
<H1>Hojas de Estilo: Tutorial CSS</H1>
<P>Ferran Perdrix Sapiña, 2004</P>
</BODY>
</HTML>
```

Nuevamente el archivo `company.css` contiene tan solo etiquetas de estilo como estas:

```
H1 { color: green; font-family: times }
P { background: yellow; font-family: courier }
```

En el ejemplo anterior, el navegador primero importa las reglas del archivo `company.css` (la línea que especifica la instrucción `@import` debe ir siempre primero) y después se permiten especificar otras reglas adicionales que conformaran en conjunto el estilo para esa página. También se puede observar que `H1` tiene dos definiciones de estilo, una en el fichero importado y otra en la definición incrustada en la misma página. La resolución de estos conflictos se explica más adelante cuando entremos en detalla del comportamiento en cascada de las Hojas de Estilo.

La flexibilidad que nos aporta este método es enorme, pudiendo importar tantos ficheros como deseemos y sobrescribiendo o añadiendo las instrucciones que se deban en cada página individual. Por desgracia pocos navegadores soportan este método para añadir Hojas de Estilo en las páginas Web. Por ejemplo, sólo las versiones de Internet Explorer 4, 5 o superiores soportan la importación de ficheros de Hojas de Estilo.

2.4. Añadir estilos en los elementos de los documentos HTML.

Por último, se puede definir estilo directamente dentro de los elementos del lenguaje HTML. Esto implica mezclar las reglas de estilo en medio de nuestro código HTML, perdiendo las ventajas relativas al hecho de separar contenido y formato. Veamos un ejemplo ilustrativo:

```
<HTML>
<HEAD>
<TITLE> Mi Primera Hoja de Estilo </TITLE>
</HEAD>
<BODY>
<H1 STYLE="color: orange; font-family: impact"> Hojas de Estilo:
Tutorial CSS </H1>
<P STYLE="background: yellow; font-family: courier"> Ferran Perdrix
Sapiña, 2004</P>
</BODY>
</HTML>
```

En este caso no precisamos ningún código en el inicio de la página, simplemente añadiendo el atributo `STYLE` dentro de la sección que queremos aplicar el estilo ya le estamos dando al navegador toda la información que necesita.

El gran problema de este método es que debes especificar el estilo cada vez que se usa. Así el próximo texto `<H1>` volvería al estilo por defecto del navegador hasta que de nuevo se encuentre un nuevo atributo `STYLE`.

Este método se considera el menos potente, aunque tiene su utilidad para especificar casos concretos que se deben tratar de forma diferente de todo el resto de apariciones del mismo elemento.

2.5. Hojas de Estilo múltiples.

Múltiples Hojas de Estilo puede solaparse una respecto a otras. Esto es debido a las diversas formas en que se puede especificar que Hoja de Estilo usar. Se puede especificar el estilo dentro de un elemento HTML, dentro de la etiqueta `<head>` de una página HTML o en un fichero CSS externo. Por tanto, para cada elemento HTML puede existir un cierto conflicto para saber que estilo se debe aplicar. Para evitar esta ambigüedad se debe seguir el siguiente orden de aplicación:

1. Estilo definido directamente en el elemento HTML.
2. Hoja de Estilo de la página (definido en la etiqueta `<head>`).
3. Hoja de Estilo externa (en fichero CSS).
4. Valores por defecto del navegador.

Siendo el estilo definido en el propio elemento el que prevalece respecto a la Hoja de Estilo de la página, a la Hoja de Estilo externa o a los valores por defecto del navegador en este orden de prioridad.

Por tanto, algunas propiedades pueden definir el mismo selector en diferentes Hojas de Estilo, entonces los valores serán aplicados preservando el más específico. Por ejemplo, una Hoja de Estilo externa tiene estas propiedades para el selector <H3>:

```
h3
{
color: red;
text-align: left;
font-size: 8pt
}
```

Y una Hoja de Estilos Interna tiene estas propiedades para el mismo selector:

```
h3
{
text-align: right;
font-size: 20pt
}
```

Finalmente la página también enlaza con una Hoja de Estilos externa que define lo siguiente sobre el mismo selector:

```
color: red;
text-align: right;
font-size: 20pt
```

En este caso descrito, el color será heredado de la Hoja de Estilo externa, mientras que *text-alignment* y el *font-size* son reemplazados por los valores de la Hoja de Estilos interna.

2.6. Asociación con XML.

Las Hojas de Estilo se pueden asociar a un documento XML [XML10] usando la instrucción de proceso `xml-stylesheet`. Esta instrucción sigue el mismo concepto que el enlace de la definición de HTML 4.0 `<LINK REL="stylesheet">` [HTML40].

La instrucción `xml-stylesheet` es parseada en el mismo modo que una etiqueta de inicio (*start-tag*), con la excepción que tan sólo pueden ser referenciadas las entidades predefinidas. La siguiente gramática sigue la misma notación que la recomendada para XML. Los símbolos de la gramática no se definen aquí, pero están disponibles en la definición de XML [XML10]:

```
[1] StyleSheetPI ::= '<?xml-stylesheet' (S PseudoAtt)* S? '?>'
[2] PseudoAtt ::= Name S? '=' S? PseudoAttValue
[3] PseudoAttValue ::= ('"' ([^"<&] | CharRef | PredefEntityRef)* '"')
| "'" ([^'<&] | CharRef | PredefEntityRef)* "'"
- (Char* '?>' Char*)
[4] PredefEntityRef ::= '&amp;' | '&lt;' | '&gt;' | '&quot;' | '&apos;'
```

Los valores *PseudoAttValue*, *CharRef* o *PredefEntityRef* son interpretados del mismo modo que un valor de atributo XML normal.

Se definen los siguientes pseudo-atributos:

```
href CDATA #REQUIRED
type CDATA #REQUIRED
title CDATA #IMPLIED
media CDATA #IMPLIED
charset CDATA #IMPLIED
alternate (yes|no) "no"
```

La semántica de los pseudo-atributos es exactamente como con la instrucción HTML `<LINK REL="stylesheet">`, con la excepción del pseudo-atributo `alternate`. Si se especifica `alternate="yes"`, entonces la instrucción de proceso tiene la semántica de `<LINK REL="alternate stylesheet">` en lugar de `<LINK REL="stylesheet">`.

En algunos casos nos puede interesar enlazar una Hoja de Estilo externa al documento XML. Por ejemplo, en las versiones anteriores de HTTP [NEGOT], [RFC2068] o la definición de HTML 4.0 permiten asociar Hojas de Estilo a un documento XML mediante la etiqueta `LINK` de la sección *header*. Todos los enlaces externos al documento XML se consideran que ocurren antes que los enlaces definidos en la instrucción de proceso `xml-stylesheet`.

A continuación se muestran varios ejemplos en HTML y su correspondiente instrucción de proceso:

```
<LINK href="mystyle.css" rel="style sheet" type="text/css">
<?xml-stylesheet href="mystyle.css" type="text/css"?>

<LINK href="mystyle.css" title="Compact" rel="stylesheet"
type="text/css">
<?xml-stylesheet href="mystyle.css" title="Compact"
type="text/css"?>

<LINK href="mystyle.css" title="Medium" rel="alternate stylesheet"
type="text/css">
<?xml-stylesheet alternate="yes" href="mystyle.css" title="Medium"
type="text/css"?>
```

También se permiten múltiples instrucciones de proceso `xml-stylesheet`, exactamente con la misma semántica que la etiqueta `LINK REL="stylesheet"`.

Veamos algunos ejemplos, primero definimos el código HTML:

```
<LINK rel="alternate stylesheet" title="compact" href="small-
base.css" type="text/css">
<LINK rel="alternate stylesheet" title="compact" href="small-
extras.css" type="text/css">
<LINK rel="alternate stylesheet" title="big print"
href="bigprint.css" type="text/css">
<LINK rel="stylesheet" href="common.css" type="text/css">
```

Este código es equivalente a:

```
<?xml-stylesheet alternate="yes" title="compact" href="small-  
base.css" type="text/css"?>  
<?xml-stylesheet alternate="yes" title="compact" href="small-  
extras.css" type="text/css"?>  
<?xml-stylesheet alternate="yes" title="big print"  
href="bigprint.css" type="text/css"?>  
<?xml-stylesheet href="common.css" type="text/css"?>
```

3. Sintaxis básica

Diseñar hojas de estilo sencillas es fácil. Sólo hay que saber un poco de HTML y algo de terminología básica de publicación electrónica. Por ejemplo, para hacer que el color de los elementos 'H1' sea azul, basta con decir:

```
H1 { color: blue }
```

Este ejemplo es una regla CSS sencilla. Una regla consta de dos partes principales: un selector ('H1') y una declaración ('*color: blue*'). La declaración tiene dos partes: una propiedad ('*color*') y un valor ('*blue*'). Aunque este ejemplo sólo intenta influir en una sola de las propiedades necesarias para representar un documento HTML, por sí mismo ya constituye una hoja de estilo. Combinada con otras hojas de estilo determinará la presentación final del documento.

El selector es el nexo entre el documento HTML y la Hoja de Estilo, y todos los tipos de elemento HTML son posibles selectores. Los tipos de elemento HTML se definen en la especificación de HTML [HTML40].

La propiedad 'color' es una de las de alrededor de cincuenta propiedades que determinan la presentación de un documento HTML. En el apéndice A se define la lista de propiedades y sus posibles valores. Los creadores de páginas HTML sólo necesitan escribir Hojas de Estilo si desean sugerir un estilo específico para sus documentos. Cada Agente de Usuario (AU, normalmente un "navegador", "explorador" o "browser") tendrá una hoja de estilo predeterminada que presentará los documentos de una manera razonable – aunque discutiblemente "aburrida". Por otro lado, basándonos en su descripción léxica [FLEX] y de gramática de producciones [YACC], la gramática formal del lenguaje CSS1 y CSS2 se define en el Apéndice C.

3.1. Agrupamiento.

Para reducir el tamaño de las hojas de estilo, se pueden agrupar selectores en listas separándolos con una coma:

```
H1, H2, H3 { font-family: helvetica }
```

Análogamente, las declaraciones también pueden agruparse:

```
H1 {
  font-weight: bold;
  font-size: 12pt;
  line-height: 14pt;
  font-family: helvetica;
  font-variant: normal;
  font-style: normal;
}
```

Además, algunas propiedades tienen su propia sintaxis de agrupamiento:

```
H1 { font: bold 12pt/14pt helvetica }
```

Que es equivalente al ejemplo previo.

3.2. Herencia.

En el primer ejemplo de este capítulo 3, se establecía que el color de los elementos 'H1' fuera azul. Supongamos que hay un elemento 'H1' que contiene un elemento enfatizado:

```
<H1>¡El título <EM>es</EM> importante!</H1>
```

Si no se ha asignado ningún color al elemento 'EM', la palabra "es" enfatizada heredará el color del elemento padre, es decir, también aparecerá en azul. Otras propiedades de estilo se heredan de la misma manera, p.ej., 'font-family' y 'font-size'. Para establecer una propiedad de un documento a un "valor por defecto", basta con establecer esa propiedad en un elemento del que desciendan todos los elementos visibles. En los documentos HTML, el elemento 'BODY' puede servir para este propósito:

```
BODY {
  color: black;
  background: url(textura.gif) white;
}
```

Esto funcionará incluso si el autor ha omitido la etiqueta 'BODY' (lo cual es legal), ya que el intérprete HTML sobreentenderá la etiqueta ausente. El ejemplo anterior establece que el color del texto es negro y que el fondo es una imagen. El fondo será blanco si la imagen no está disponible. Algunas propiedades de estilo no se heredan del elemento padre al elemento hijo. El porqué de esto se comprende intuitivamente en la mayoría de los casos. P.ej., la propiedad 'background' no se hereda, pero el fondo del elemento padre se verá por defecto a través del elemento hijo. Muchas veces el valor de una propiedad es un porcentaje que se refiere a otra propiedad:

```
P { font-size: 10pt }
P { line-height: 120% }/* relativo a 'font-size', es decir, 12pt */
```

Para todas las propiedades que admiten como valor un porcentaje, se define a qué propiedad se refiere éste. En el ejemplo, los elementos hijos de un elemento 'P' heredarán el valor calculado de 'line-height' (es decir, 12pt), no el porcentaje.

3.3. Clase como selector.

Para incrementar el grado de control sobre los elementos, se ha añadido un nuevo elemento a HTML: el atributo 'CLASS'. Se pueden clasificar todos los elementos contenidos en el elemento 'BODY', y en la Hoja de Estilo se puede hacer referencia a cada clase:

```
<HTML>
<HEAD>
  <TITLE>Titulo</TITLE>
  <STYLE TYPE="text/css">
    H1.tutorial { color: #00FF00 }
  </STYLE>
</HEAD>
<BODY>
  <H1 CLASS=tutorial>Hojas de Estilo</H1>
</BODY>
</HTML>
```

Las reglas normales de herencia también se aplican a los elementos clasificados: heredan valores de su padre en la estructura del documento. Se puede hacer referencia a todos los elementos de la misma clase omitiendo el nombre de la etiqueta en el selector:

```
.tutorial { color: green } /* todos los elementos de la clase
                           tutorial (CLASS=tutorial) */
```

Sólo se puede especificar una clase por selector. Por lo tanto, '*P.tutorial.marino*' es un selector no válido en CSS1. (Los selectores contextuales, descritos más adelante, pueden tener una clase por cada selector simple.) El CSS le da tanta potencia al atributo 'CLASS' que en muchos casos ni siquiera importa a qué elemento HTML se le asigne la clase. Puede conseguirse que un elemento cualquiera emule prácticamente a cualquier otro. No es recomendable aprovecharse de esta característica, ya que así se elimina un nivel de estructura que tiene significado universal (elementos HTML). Una estructura basada en 'CLASS' sólo es útil dentro de un dominio restringido, en el que el significado de una clase se haya establecido de antemano.

3.4. ID como selector.

HTML también introduce el atributo 'ID', del cual se garantiza que tiene un valor único en todo el documento. Por tanto puede ser de especial importancia como selector en una Hoja de Estilo, y se puede hacer referencia a él anteponiendo el símbolo '#':

```
#z98y { letter-spacing: 0.3em }
H1#z98y { letter-spacing: 0.5em }

<P ID=z98y>Texto ancho</P>
```

En este ejemplo, el primer selector corresponde al elemento 'P' debido al valor de su atributo 'ID'. El segundo selector especifica tanto un tipo de elemento ('H1') como un valor ID, y por tanto no corresponde al elemento 'P'. Usando el atributo ID como

selector se pueden establecer propiedades elemento por elemento. Si bien las Hojas de Estilo han sido diseñadas para acrecentar la estructura de los documentos, esta característica permitirá a los autores crear documentos que se verán bien en el lienzo sin sacar provecho de los elementos estructurales del HTML. Se recomienda encarecidamente no hacer uso de las Hojas de Estilo de esta manera.

3.5. Selectores contextuales.

La herencia permite a los diseñadores CSS escribir menos. En lugar de establecer todas las propiedades de estilo, se pueden crear primero los valores por defecto y después enumerar las excepciones. Para dar a los elementos 'EM' contenidos en un 'H1' un color diferente, se puede especificar lo siguiente:

```
H1 { color: blue }
EM { color: red }
```

Cuando esta Hoja de Estilo tenga efecto, todas las secciones enfatizadas saldrán en rojo, estén dentro o fuera de un 'H1'. Es probable que sólo quisiéramos que salieran en rojo los elementos 'EM' contenidos en un 'H1', y eso se puede conseguir de la siguiente manera:

```
H1 EM { color: red }
```

El selector es ahora un patrón de búsqueda en la pila de elementos abiertos. Este tipo de selectores se conoce como *selectores contextuales*. Los selectores contextuales consisten en varios selectores simples separados por espacios (todos los selectores descritos hasta ahora eran selectores simples). El selector sólo hará referencia a los elementos que se correspondan con el último selector simple (en este caso el elemento 'EM'), y sólo si concuerdan con el patrón de búsqueda. En CSS1 los selectores contextuales sólo buscan relaciones de descendencia, pero pueden introducirse otros tipos de relación (p.ej. padre-hijo) en revisiones posteriores. En el ejemplo anterior, el patrón de búsqueda concuerda si 'EM' es descendiente de 'H1', es decir, si 'EM' está en el interior de un elemento 'H1'.

```
UL LI { font-size: small }
UL UL LI { font-size: x-small }
```

Aquí, el primer selector corresponde a elementos 'LI' que tengan al menos un ascendiente 'UL'. El segundo selector corresponde a un subconjunto del primero, es decir, elementos 'LI' que tengan al menos dos ascendientes 'UL'. El conflicto se resuelve a favor del segundo selector por ser más específico, debido a que el patrón de búsqueda es más largo. Los selectores contextuales pueden buscar tipos de elementos, atributos CLASS, atributos ID o combinaciones de éstos:

```
DIV P { font: small sans-serif }
.rojizo H1 { color: red }
#x78y CODE { background: blue }
DIV.notaalmargen H1 { font-size: large }
```

El primer selector corresponde a todos los elementos 'P' que tengan a un 'DIV' entre sus ascendientes. El segundo selector corresponde a todos los elementos 'H1' que tengan un ascendiente de clase 'rojizo'.

El tercer selector corresponde a todos los elementos 'CODE' que sean descendientes del elemento de 'ID=x78y'. El cuarto selector corresponde a todos los elementos 'H1' que tengan un ascendiente 'DIV' de clase 'notaalmargen'.

Los selectores contextuales también se pueden agrupar:

```
H1 B, H2 B, H1 EM, H2 EM { color: red }
```

que es equivalente a:

```
H1 B { color: red }
H2 B { color: red }
H1 EM { color: red }
H2 EM { color: red }
```

3.6. Comentarios.

Los comentarios literales en hojas de estilo CSS son similares a los del lenguaje de programación C [ISO9899]:

```
EM { color: red } /* ¡¡rojo, pero que muy rojo!! */
```

No se pueden anidar comentarios. Para un analizador CSS1, un comentario equivale a espacio en blanco.

3.7. Pseudo-clases y pseudo-elementos.

En CSS1, el estilo normalmente se asocia con un elemento según la posición de éste en la estructura del elemento. Este modelo simple es suficiente para una amplia variedad de estilos, pero no cubre algunos efectos comunes. El concepto de pseudo-clases y pseudo-elementos extiende los modos de referencia de CSS1 para permitir que informaciones externas tengan influencia a la hora de dar formato.

Las pseudo-clases y los pseudo-elementos pueden usarse en CSS como selectores, pero no existen en el código fuente HTML, sino que son "insertados" por el AU bajo ciertas condiciones para que las hojas de estilo puedan hacer referencia a ellos. Se hace referencia a ellos como "clases" y como "elementos" ya que esta es una manera conveniente de describir su comportamiento. Más concretamente, su comportamiento se define mediante una *secuencia ficticia de etiquetas*.

Los pseudo-elementos se utilizan para hacer referencia a sub-partes de elementos, mientras que las pseudo-clases permiten a las hojas de estilo diferenciar entre distintos tipos de elementos. Veámoslo más en detalle:

Pseudo-clases de vínculo

Los Agentes de Usuario suelen representar los vínculos aún no visitados de manera distinta a los ya visitados. En CSS1, se puede actuar sobre esto a través de pseudo-clases del elemento 'A':

```
A:link { color: red }           /* vínculos no visitados */
A:visited { color: blue }      /* vínculos visitados */
A:active { color: lime }       /* vínculos activos */
```

Todos los elementos 'A' que tengan un atributo 'HREF' serán colocados en uno y sólo uno de estos grupos (es decir, los vínculos destino no se ven afectados). Los AAUU pueden elegir mover un elemento de '*visited*' a '*link*' al cabo de un cierto tiempo. Un vínculo '*active*' es uno que está siendo seleccionado en ese momento por el lector (p.ej., haciendo clic con un botón del ratón).

Una pseudo-clase de vínculo se trata como si la clase se hubiera insertado manualmente. No es necesario que un AU reformatee un documento mostrado actualmente a causa de transiciones entre pseudo-classes de vínculo. P.ej., una hoja de estilo puede especificar legalmente que el '*font-size*' de un vínculo '*active*' debería ser mayor que el de un vínculo '*visited*', pero no es necesario que el AU reformatee dinámicamente el documento cuando el lector seleccione el vínculo '*visited*'. Los selectores de pseudo-classes no hacen referencia a clases normales, y viceversa. Por tanto, la regla de estilo del ejemplo siguiente no tendrá ninguna influencia:

```
A:link { color: red }  
<A CLASS=link NAME=target5> ... </A>
```

En CSS1, las pseudo-classes de vínculo no tendrán ningún efecto en elementos que no sean 'A'. Por lo tanto, el tipo de elemento puede omitirse del selector:

```
A:link { color: red }  
:link { color: red }
```

Estos dos selectores seleccionarán en CSS1 los mismos elementos. Los nombres de pseudo-classes no requieren coincidencia de mayúsculas y minúsculas. Las pseudo-classes pueden utilizarse en selectores contextuales:

```
A:link IMG { border: solid blue }
```

Asimismo, las pseudo-classes pueden combinarse con las clases normales:

```
A.external:visited { color: blue }  
<A CLASS=external HREF="http://fue.ra/">vínculo externo</A>
```

Si el vínculo de este ejemplo ha sido visitado, será mostrado en azul. Obsérvese que los nombres de las clases normales preceden a las pseudo-classes en el selector.

Pseudo-elementos tipográficos

Algunos efectos tipográficos comunes están asociados no con elementos estructurales, sino más bien con elementos tipográficos resultantes del formato del documento en el lienzo. En CSS1 se puede hacer referencia a dos de estos elementos tipográficos a través de pseudo-elementos: la primera línea de un elemento, y la primera letra. Los AAUU pueden no tener en cuenta las reglas que contengan '*:first-line*' o '*:first-letter*' en el selector, o, alternativamente, soportar sólo parte de las propiedades de estos pseudo-elementos.

El pseudo-elemento '*first-line*'

El pseudo-elemento '*first-line*' se utiliza para aplicar estilos especiales a la primera línea de un elemento una vez que se ha dado formato a éste en el lienzo:

```
<STYLE TYPE="text/css">
  P:first-line { font-variant: small-caps }
</STYLE>
```

```
<P>La primera línea de un artículo del Newsweek.</P>
```

En un AU de modo texto, esto podría formatearse del siguiente modo:

```
LA PRIMERA LÍNEA DE UN
artículo del Newsweek.
```

La secuencia ficticia de etiquetas de este ejemplo es la siguiente:

```
<P>
<P:first-line>
La primera línea de un
</P:first-line>
artículo del Newsweek.
</P>
```

La etiqueta final de *'first-line'* se inserta al final de la primera línea que resulta al formatear en el lienzo. El pseudo-elemento *'first-line'* sólo puede asociarse a elementos en bloque. El pseudo-elemento *'first-line'* es similar a un elemento en línea, pero con ciertas restricciones. Solamente las siguientes propiedades se aplican al elemento *'first-line'*: propiedades de fuente, propiedades de color y fondo, *'word-spacing'*, *'letter-spacing'*, *'text-decoration'*, *'vertical-align'*, *'text-transform'*, *'line-height'*, *'clear'*.

El pseudo-elemento *'first-letter'*

El pseudo-elemento *'first-letter'* se utiliza para letras capitales, que es un efecto tipográfico muy común. Es similar a un elemento en línea si su propiedad *'float'* es *'none'*, de otro modo es similar a un elemento flotante. Estas son las propiedades que se aplican a un pseudo-elemento *'first-letter'*: propiedades de fuente, propiedades de color y fondo, *'text-decoration'*, *'vertical-align'* (sólo si *'float'* es *'none'*), *'text-transform'*, *'line-height'*, propiedades de margen, propiedades de relleno, propiedades de borde, *'float'*, *'clear'*. Así es como se consigue que una letra capital en texto se extienda a lo largo de dos líneas:

```
<HTML>
<HEAD>
  <TITLE>Titulo</TITLE>
  <STYLE TYPE="text/css">
    P { font-size: 12pt; line-height: 12pt }
    P:first-letter { font-size: 200%; float: left }
    SPAN { text-transform: uppercase }
  </STYLE>
</HEAD>
<BODY>
  <P><SPAN>Las primeras</SPAN> palabras de un artículo de The
Economist.</P>
</BODY>
</HTML>
```

Si un AU de modo texto soporta el pseudo-elemento *'first-letter'* (lo más seguro es que no lo soporte) el código anterior podría formatearse del siguiente modo:

```
| AS PRIMERAS
|__palabras de
un artículo de
The Economist.
```

La secuencia ficticia de etiquetas es la siguiente:

```
<P>
<SPAN>
<P:first-letter>
L
</P:first-letter>as primeras
</SPAN>
palabras de un artículo de The Economist.
</P>
```

Obsérvese que las etiquetas del pseudo-elemento *'first-letter'* están contiguas a su contenido (es decir, al carácter inicial), mientras que la etiqueta inicial del pseudo-elemento *'first-line'* se inserta justo a continuación de la etiqueta inicial del elemento al cual se asocia. El AU define qué caracteres están contenidos en el elemento *'first-letter'*. Normalmente deben incluirse las comillas que precedan a la primera letra:

```
|| |\ /|ás vale
| \ /|pájaro
| | |en mano
| | |que
ciento volando",
dice el refrán.
```

Cuando el párrafo comienza con otro signo de puntuación (p.ej. paréntesis o puntos suspensivos) u otros caracteres que normalmente no se consideren como letras (p.ej., dígitos y símbolos matemáticos), normalmente no se tiene en cuenta el pseudo-elemento *'first-letter'*. Algunos idiomas pueden tener reglas específicas sobre cómo tratar ciertas combinaciones de letras. En neerlandés, por ejemplo, si la combinación de letras "ij" aparece al comienzo de una palabra, deberían considerarse ambas contenidas en el pseudo-elemento *'first-letter'*. El pseudo-elemento *'first-letter'* sólo puede ser asociado a elementos en bloque.

Pseudo-elementos en selectores

En un selector contextual, los pseudo-elementos sólo están permitidos al final del selector:

```
BODY P:first-letter { color: purple }
```

En los selectores los pseudo-elementos pueden combinarse con clases:

```
P.inicial:first-letter { color: red }

<P CLASS=inicial>Primer párrafo</A>
```

En este ejemplo la primera letra de todos los elementos 'P' con 'CLASS=inicial' aparecería en rojo. Cuando se combinan con clases o con pseudo-clases, los pseudo-elementos deben situarse al final del selector. Sólo puede especificarse un pseudo-elemento por selector.

Múltiples pseudo-elementos

Pueden combinarse varios pseudo-elementos:

```
P { color: red; font-size: 12pt }
P:first-letter { color: green; font-size: 200% }
P:first-line { color: blue }
```

```
<P>Un texto que ocupará dos líneas</P>
```

En este ejemplo, la primera letra de cada elemento 'P' sería verde y tendría un tamaño de 24pt. El resto de la línea (resultante del formateo en el lienzo) sería azul, mientras que el resto del párrafo sería rojo. Suponiendo que haya un salto de línea antes de la palabra "ocupará", la secuencia ficticia de etiquetas será:

```
<P>
<P:first-line>
<P:first-letter>
U
</P:first-letter>n texto que
</P:first-line>
ocupará dos líneas
</P>
```

Obsérvese que el elemento *'first-letter'* está dentro del elemento *'first-line'*. Las propiedades establecidas en *'first-line'* serán heredadas por *'first-letter'*, pero son anuladas si las mismas propiedades se establecen en *'first-letter'*. Si un pseudo-elemento rompe un elemento real, deben regenerarse en la secuencia ficticia las etiquetas adicionales necesarias. Por ejemplo, si un elemento 'SPAN' se extiende a través de una etiqueta `</P:first-line>` debe regenerarse un par de etiquetas SPAN final e inicial, y la secuencia ficticia de etiquetas se convierte en:

```
<P>
<P:first-line>
<SPAN>
Este texto está dentro
</SPAN>
</P:first-line>
<SPAN>
de un elemento span largo
</SPAN>
```

3.8. Diseño en cascada.

En CSS, como ya hemos mencionado antes, más de una hoja de estilo puede influir simultáneamente en la presentación de un documento. Dos son las razones principales de esta característica: modularidad y equilibrio autor/lector.

Modularidad

Un diseñador de hojas de estilo puede combinar varias hojas de estilo (parciales) para reducir redundancias:

```
@import url(http://www.style.org/pastoral);
@import url(http://www.style.org/marino);

H1 { color: red }      /* prevalece sobre las hojas importadas */
```

Equilibrio autor/lector

Tanto los autores como los lectores pueden influir sobre la presentación a través de Hojas de Estilo. Para ello usan el mismo lenguaje de Hojas de Estilo, reflejando así una característica fundamental de la Web: todo el mundo puede convertirse en editor. El AU es libre de elegir el mecanismo para hacer referencia a las Hojas de Estilo personales.

Algunas veces surgirán conflictos entre las Hojas de Estilo que influyen en una presentación. La resolución de conflictos se basa en que cada regla de estilo tiene un peso. Por defecto, los pesos de las reglas del lector son menores que los pesos de las reglas de los documentos del autor. Es decir, si hay conflictos entre las Hojas de Estilo de un documento entrante y las hojas personales del lector, se usarán las reglas del autor. Tanto las reglas del autor como las del lector prevalecen sobre los valores por defecto del AU.

Las Hojas de Estilo importadas también están en cascada las unas con las otras, en el orden en que son importadas, de acuerdo con las reglas de cascada definidas más adelante. Cualquier regla especificada en la propia Hoja de Estilo prevalece sobre las reglas de las Hojas de Estilo importadas. Es decir, que las Hojas de Estilo importadas están por debajo en el orden de cascada que las reglas de la propia Hoja de Estilo. Las Hojas de Estilo importadas pueden a su vez importar y prevalecer sobre otras Hojas de Estilo, recursivamente.

En CSS1, todas las sentencias "@import" deben figurar al comienzo de una Hoja de Estilo, antes de cualquier declaración. Así es fácil ver qué reglas de la propia Hoja de Estilo prevalecen sobre las reglas de las Hojas de Estilo importadas.

Regla 'important'

Los diseñadores de Hojas de Estilo pueden aumentar el peso de sus declaraciones:

```
H1 { color: black ! important; background: white ! important }
P  { font-size: 12pt ! important; font-style: italic }
```

En este ejemplo, las primeras tres declaraciones tienen su peso incrementado, mientras que la última declaración tiene peso normal. Una regla del lector con una declaración importante prevalecerá sobre una regla del autor con una declaración normal. Una regla del autor con una declaración importante prevalecerá sobre una regla del lector con una declaración importante.

Orden de cascada

Los conflictos entre reglas son intrínsecos al mecanismo de CSS. Para hallar el valor de una combinación elemento/propiedad, debe seguirse el siguiente algoritmo:

1. Encontrar todas las declaraciones que se aplican al elemento/propiedad en cuestión. Una declaración se aplica si el selector corresponde al elemento en cuestión. Si no se aplica ninguna declaración, se usa el valor heredado. Si no hay valor heredado (este es el caso para el elemento HTML y para las propiedades que no se heredan) se usa el valor inicial.
2. Ordenar las declaraciones por pesos explícitos: las declaraciones marcadas como '!important' tienen más peso que las declaraciones no marcadas (normales).
3. Ordenar por origen: las Hojas de Estilo del autor prevalecen sobre las Hojas de Estilo del lector, que prevalecen sobre los valores por defecto del agente de usuario. Una Hoja de Estilo importada tiene el mismo origen que la Hoja de Estilo desde la cual ha sido importada.
4. Ordenar por especificidad del selector: los selectores más específicos prevalecerán sobre los más generales. Para averiguar la especificidad, se cuenta el número de atributos ID en el selector (a), el número de atributos CLASS en el selector (b), y el número de nombres de etiqueta en el selector (c). Concatenando los tres números (en un sistema numérico de base elevada) se obtiene la especificidad. A continuación figuran algunos ejemplos:

```

LI           {...} /* a=0 b=0 c=1 -> especificidad = 1 */
UL LI       {...} /* a=0 b=0 c=2 -> especificidad = 2 */
UL OL LI    {...} /* a=0 b=0 c=3 -> especificidad = 3 */
LI.red      {...} /* a=0 b=1 c=1 -> especificidad = 11 */
UL OL LI.red {...} /* a=0 b=1 c=3 -> especificidad = 13 */
#x34y      {...} /* a=1 b=0 c=0 -> especificidad = 100 */

```

Los pseudo-elementos y las pseudo-clases se cuentan como elementos y clases normales respectivamente.

5. Ordenar por orden especificado: si dos reglas tienen el mismo peso, se impone la última especificada. Se considera que las reglas de las Hojas de Estilo importadas van antes que las reglas de la propia Hoja de Estilo.

La búsqueda del valor de la propiedad puede darse por terminada una vez que una regla tenga un peso mayor que las demás reglas que se aplican a la misma combinación elemento/propiedad. Esta estrategia da al autor de las Hojas de Estilo pesos considerablemente mayores que los del lector.

Es por tanto importante que el lector tenga la capacidad de desactivar la influencia de una cierta Hoja de Estilo, por ejemplo a través de un menú desplegable. Una declaración en el atributo 'STYLE' de un elemento tiene el mismo peso que una declaración basada en un ID especificado al final de la Hoja de Estilo:

```
<STYLE TYPE="text/css">
  #x97z { color: blue }
</STYLE>

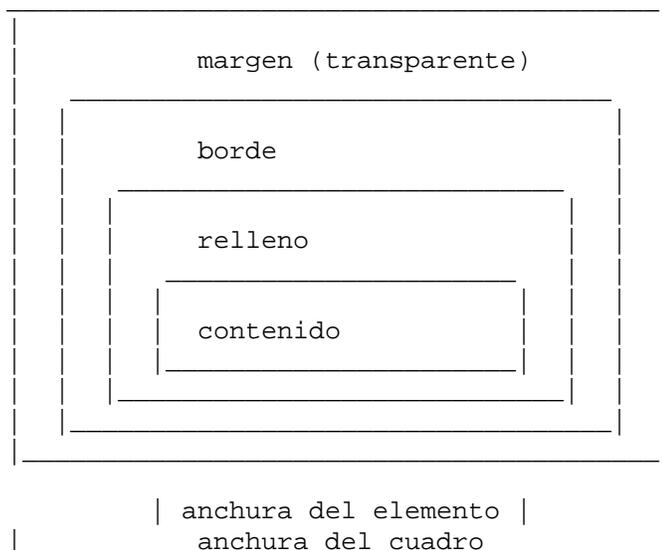
<P ID=x97z STYLE="color: red">
```

En este ejemplo, el color del elemento 'P' sería rojo. Aunque la especificidad es la misma para ambas declaraciones, la declaración en el atributo 'STYLE' prevalecerá sobre la del elemento 'STYLE', debido a la regla de cascada número 5.

El AU puede opcionalmente tener en cuenta otros atributos estilísticos de HTML, por ejemplo, 'ALIGN'. En ese caso, estos atributos se traducen a las reglas CSS correspondientes con especificidad igual a 1. Se supone que estas reglas están al comienzo de la Hoja de Estilo del autor y pueden ser anuladas por reglas de estilo subsiguientes.

3.9. Modelo de formato.

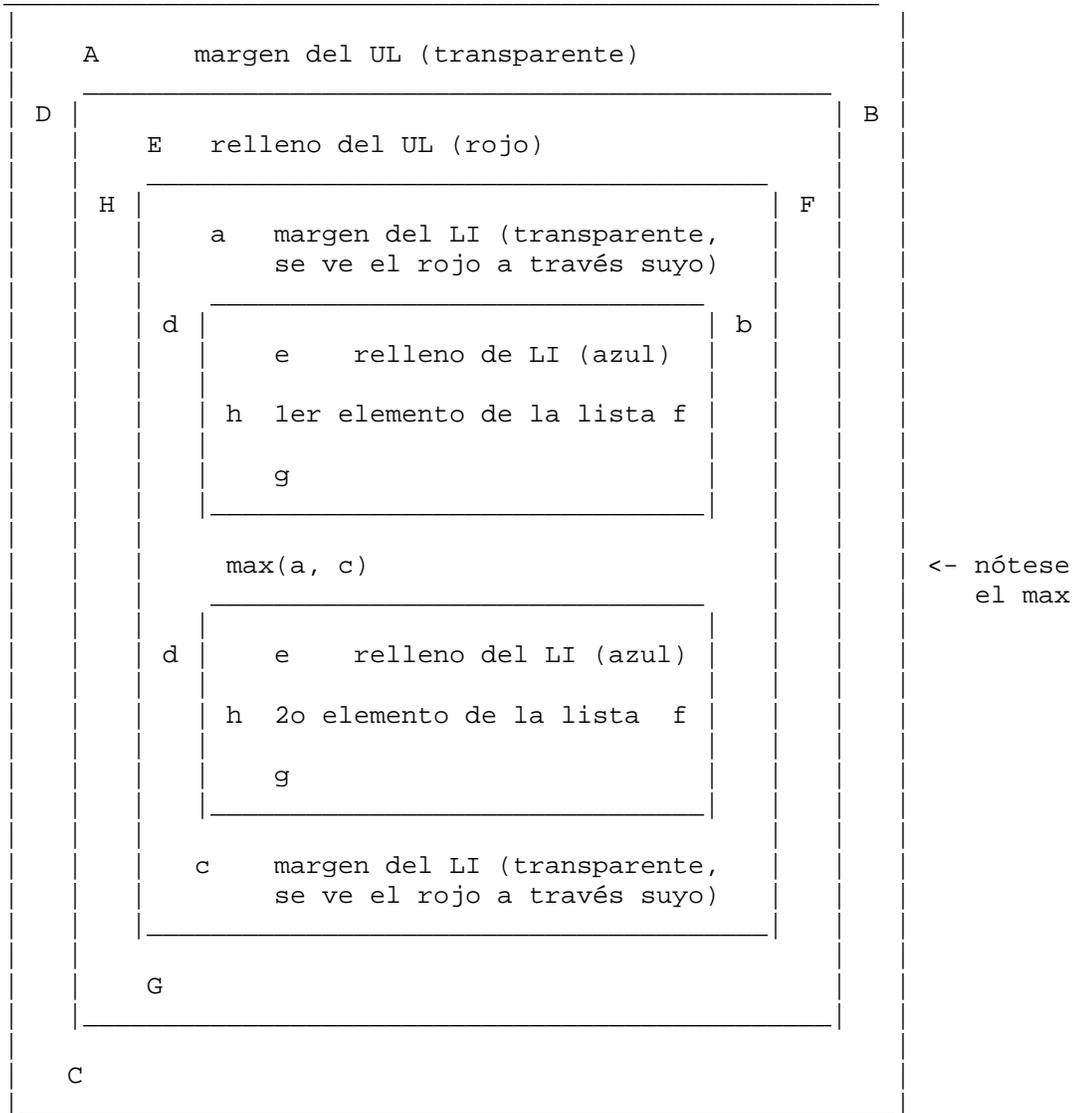
CSS asume un modelo de formato simple basado en cuadros (*boxes*), en el que cada elemento formateado da lugar a uno o más cuadros rectangulares. (Los elementos que tienen un valor de '*display*' igual a '*none*' no son formateados, y por tanto no dan lugar a un cuadro.) Todos los cuadros tienen un área central de contenido, con áreas opcionales a su alrededor de relleno, bordes y márgenes.



El tamaño de margen, relleno y borde se establece con las propiedades de margen, de relleno, y de borde respectivamente. El área de relleno usa el mismo fondo que el elemento en sí (establecido con las propiedades de fondo). El color y estilo del borde se establecen con las propiedades de borde. Los márgenes son siempre transparentes, de manera que el elemento padre se verá a través de ellos. El tamaño del cuadro es la suma de la anchura del elemento (es decir, el texto o imagen formateados) y las áreas de relleno, bordes y márgenes. Desde el punto de vista del formateador hay dos tipos principales de elementos: en bloque y en línea.

Elementos en bloque

Los elementos con un valor de 'display' igual a 'block' o a 'list-item' son *elementos en bloque*. Asimismo, los elementos flotantes (elementos con un valor de 'float' distinto a 'none') se consideran como elementos en bloque. El siguiente ejemplo muestra cómo los márgenes y el relleno formatean un elemento 'UL' con dos hijos. Para simplificar el diagrama se ha prescindido de los bordes. Por otra parte, las "constantes" literales del ejemplo no son sintaxis legal de CSS1, pero es un modo conveniente de relacionar los valores de la Hoja de Estilo con la figura.



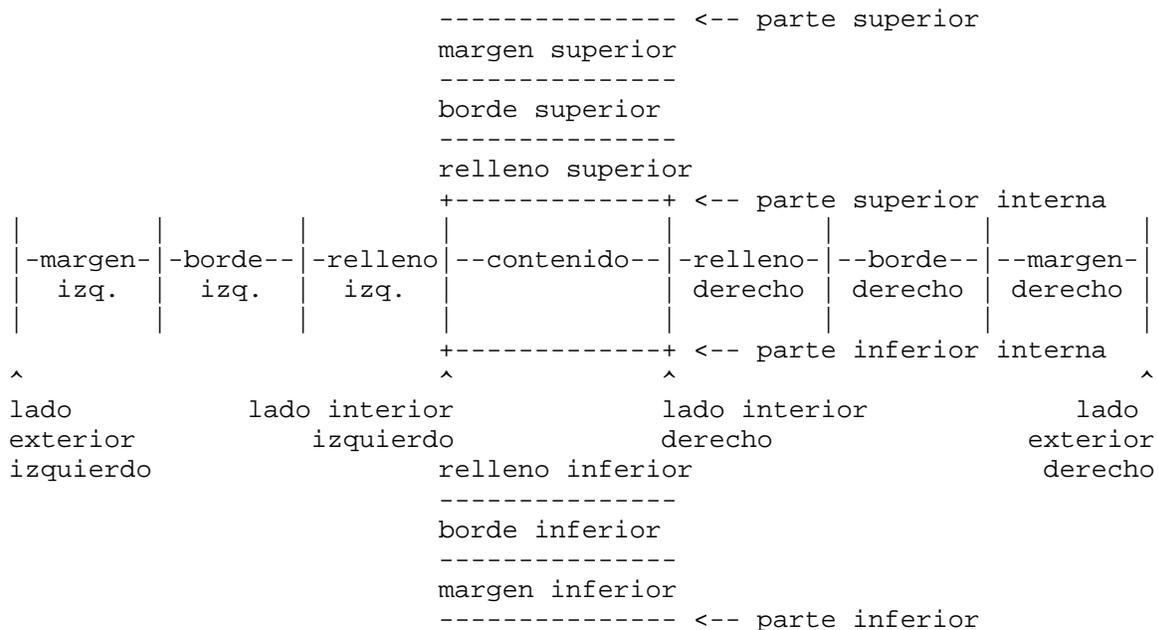
```
<STYLE TYPE="text/css">
  UL {
    background: red;
    margin: A B C D;
    padding: E F G H;
  }
```

```

LI {
  color: white;
  background: blue;      /* así el texto es blanco sobre azul */
  margin: a b c d;
  padding: e f g h;
}
</STYLE>
..
<UL>
  <LI>1er elemento de la lista
  <LI>2o elemento de la lista
</UL>

```

Técnicamente, las propiedades de relleno y margen no se heredan. Pero como muestra el ejemplo, la colocación de un elemento es relativa a los ascendientes y hermanos, de modo que las propiedades de relleno y margen de estos elementos tienen efecto en sus hijos. Si hubiera habido bordes en el ejemplo anterior, habrían aparecido entre el relleno y los márgenes. El siguiente diagrama introduce cierta terminología útil:



El *lado exterior izquierdo* es el lado de un elemento incluido su relleno, su borde y su margen. El *lado interior izquierdo* es el lado del contenido únicamente, por dentro de relleno, borde y margen. Lo mismo para los lados derechos. La *parte superior* es la parte superior del elemento incluyendo relleno, borde y margen; sólo se define para elementos en línea y elementos flotantes, no para elementos en bloque no flotantes. La *parte superior interna* es la parte superior del contenido, por dentro de relleno, borde y margen. La parte inferior es la parte inferior del elemento, por fuera de relleno, borde y margen. Sólo está definida para elementos en línea y flotantes, no para elementos en bloque no flotantes. La *parte inferior interna* es la parte inferior del elemento, por dentro de relleno, borde y margen. La *anchura* de un elemento es la anchura del contenido, es decir, la distancia entre el lado interior izquierdo y el lado interior derecho. La *altura* es la altura del contenido, es decir, la distancia entre la parte inferior interna y la parte superior interna.

Formato vertical

La anchura del margen de los elementos en bloque no flotantes especifica la distancia mínima a los bordes a los cuadros adyacentes. Dos o más márgenes verticales adyacentes (es decir, sin borde, relleno ni contenido entre ellos) se colapsan para usar el máximo de los dos valores de margen. En la mayoría de los casos, después de colapsarse los márgenes verticales, el resultado es visualmente más agradable y más parecido a lo que espera el diseñador. En el ejemplo anterior, los márgenes entre los dos elementos 'LI' se colapsan usando el máximo entre el margen inferior (*'margin-bottom'*) del primer elemento 'LI' y el margen superior (*'margin-top'*) del segundo elemento 'LI'. Análogamente, si el relleno entre el elemento 'UL' y el primer elemento 'LI' (la constante "E") hubiera sido cero, los márgenes del elemento 'UL' y del primer elemento 'LI' habrían sido colapsados. En el caso de márgenes negativos, el máximo absoluto de los márgenes negativos adyacentes se resta del máximo de los márgenes positivos adyacentes. Si no hay márgenes positivos, el máximo absoluto de los márgenes adyacentes negativos se resta de cero.

Formato horizontal

La posición y tamaño horizontales de un elemento en bloque no flotante quedan determinados por siete propiedades: *'margin-left'*, *'border-left'*, *'padding-left'*, *'width'*, *'padding-right'*, *'border-right'* y *'margin-right'*. La anchura de estas siete propiedades es siempre igual a la anchura del elemento padre. Por defecto, la anchura (*'width'*) de un elemento es *'auto'*. Si el elemento no es un elemento reemplazado, esto significa que el AU calcula la anchura de modo que la suma de las siete propiedades mencionadas en el párrafo precedente sea igual a la anchura del padre. Si el elemento es un elemento reemplazado, un valor *'auto'* de *'width'* se reemplaza automáticamente por la anchura intrínseca del elemento. Tres de las siete propiedades pueden ser establecidas a *'auto'*: *'margin-left'*, *'width'* y *'margin-right'*. Para los elementos reemplazados, un valor de *'width'* igual a *'auto'* se reemplaza por la anchura intrínseca, de modo que para ellos sólo puede haber dos valores *'auto'*. El *'width'* tiene un valor mínimo no negativo definido por el AU (que puede variar de elemento a elemento e incluso depender de otras propiedades). Si *'width'* cae por debajo de este límite, ya sea porque se ha especificado así explícitamente, o porque valiendo *'auto'* las reglas que se describen más adelante lo hacen demasiado pequeño, el valor será reemplazado por el valor mínimo.

Si *exactamente una* de las propiedades *'margin-left'*, *'width'* y *'margin-right'* es *'auto'*, el AU asignará a esa propiedad un valor que haga que la suma de las siete sea igual a la anchura del padre. Si *ninguna* de las propiedades vale *'auto'*, se asignará el valor *'auto'* al *'margin-right'*. Si *más de una* de las tres vale *'auto'*, y una de ellas es *'width'*, entonces las otras (*'margin-left'* y/o *'margin-right'*) serán puestas a cero, y *'width'* recibirá el valor necesario para que la suma de las siete sea igual a la anchura del padre. De otro modo, si tanto *'margin-left'* como *'margin-right'* son *'auto'*, se les asignará el mismo valor. Esto centrará al elemento dentro de su padre. Si se establece *'auto'* como el valor de una de las siete propiedades de un elemento en línea o flotante, se tratará como si se hubiera establecido a cero. A diferencia de los márgenes verticales, los márgenes horizontales no se colapsan.

Elementos objeto de lista

Los elementos con un valor de la propiedad *'display'* igual a *'list-item'* se formatean como elementos en bloque, pero precedidos por un marcador de objeto de lista. El tipo de marcador se determina por la propiedad *list-style*. El marcador se coloca de acuerdo con el valor de la propiedad *list-style*:

```
<STYLE TYPE="text/css">
  UL          { list-style: outside }
  UL.compact { list-style: inside }
</STYLE>

<UL>
  <LI>el primer elemento va primero
  <LI>el segundo elemento va después
</UL>

<UL CLASS=COMPACT>
  <LI>el primer elemento va primero
  <LI>el segundo elemento va después
</UL>
```

Este ejemplo puede formatearse como sigue:

```
* el primer elemento
  va primero

* el segundo elemento
  va después

* el primer elemento
  va primero

* el segundo elemento
  va después
```

Si el texto fuera de derecha a izquierda, los marcadores habrían estado en el lado derecho del cuadro.

Elementos flotantes

Por medio de la propiedad *'float'*, puede declararse que un elemento esté fuera del flujo normal de los elementos, formateándose entonces como un elemento en bloque. Por ejemplo, haciendo la propiedad *'float'* de una imagen igual a *'left'*, la imagen se mueve hacia la izquierda hasta que se alcanza el margen, relleno o borde de otro elemento en bloque. El flujo normal continuará por el lado derecho. Los márgenes, bordes y relleno del propio elemento serán respetados, y los márgenes nunca se colapsarán con los márgenes de elementos adyacentes. Un elemento es posicionado de acuerdo con las siguientes restricciones:

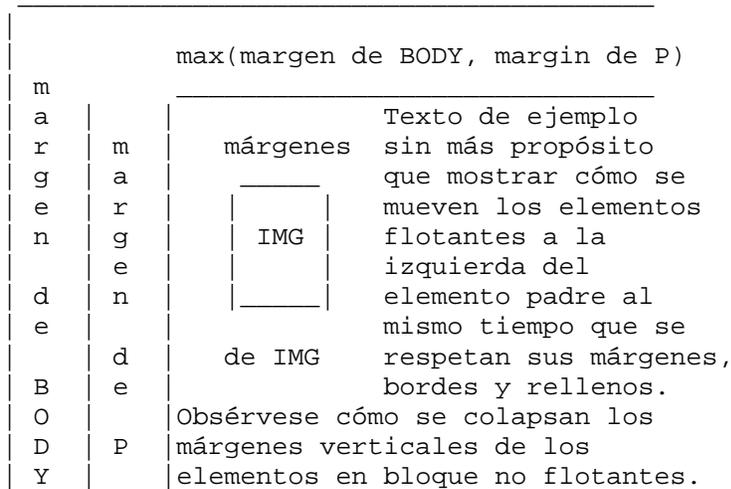
1. El lado exterior izquierdo de un elemento flotante a la izquierda no puede estar a la izquierda del lado interior izquierdo de su elemento padre. Análogamente para los elementos flotantes a la derecha.

2. El lado exterior izquierdo de un elemento flotante a la izquierda debe estar a la derecha del lado exterior derecho de todos los elementos flotantes a la izquierda anteriores (en el código fuente HTML), o la parte superior de aquél debe estar por debajo de la parte inferior de éstos. Análogamente para los elementos flotantes a la derecha.
3. El lado exterior derecho de un elemento flotante a la izquierda no puede estar a la derecha del lado exterior izquierdo de ningún elemento flotante a la derecha que esté a la derecha de aquél. Análogamente para los elementos flotantes a la derecha.
4. La parte superior de un elemento flotante no puede estar por encima de la parte superior interna de su padre.
5. La parte superior de un elemento flotante no puede estar por encima de la parte superior de ningún elemento flotante o en bloque anterior.
6. La parte superior de un elemento flotante no puede estar por encima de la parte superior de ningún *cuadro de línea* cuyo contenido preceda al elemento flotante en el código fuente HTML.
7. Un elemento flotante debe ser colocado lo más arriba que sea posible.
8. Un elemento flotante a la izquierda debe ser colocado lo más a la izquierda posible, y un elemento flotante a la derecha lo más a la derecha posible. Tiene preferencia una posición más alta que una que esté más a la izquierda/derecha.

```
<STYLE TYPE="text/css">
  IMG { float: left }
  BODY, P, IMG { margin: 2em }
</STYLE>

<BODY>
  <P>
    <IMG SRC=img.gif>
    Texto de ejemplo sin más...
  </BODY>
```

Este ejemplo podría formatearse como sigue:



Obsérvese que el margen del elemento 'P' está alrededor del elemento 'IMG' flotante. Hay dos situaciones en que un elemento flotante puede superponerse con los márgenes, bordes y rellenos de otros elementos:

- cuando el elemento flotante tiene un margen negativo: los márgenes negativos de los elementos flotantes se respetan como en otros elementos en bloque.
- cuando el elemento flotante es más ancho o más alto que el elemento en que está contenido.

Elementos en línea

Los elementos que no son formateados como elementos en bloque son *elementos en línea*. Un elemento en línea puede compartir el espacio de una línea con otros elementos. Considérese este ejemplo:

```
<P>Aquí <STRONG>aparecen</STRONG> varias <EM>palabras
enfáticas</EM>.</P>
```

El elemento 'P' es normalmente un elemento en bloque, mientras que los elementos 'STRONG' y 'EM' son elementos en línea. Si el elemento 'P' es lo suficientemente ancho como para que se formatee todo el elemento en una sola línea, habrá dos elementos en línea en esa línea:

Aquí **aparecen** varias *palabras enfáticas*.

Si no hay suficiente espacio en una sola línea, un elemento en línea será partido en varios cuadros:

```
<P>Aquí aparecen <EM>varias palabras</EM> enfáticas.</P>
```

Este ejemplo podría formatearse como sigue:

Aquí aparecen *varias*
palabras enfatizadas

Si el elemento tiene márgenes, bordes, rellenos u otras decoraciones asociadas, éstos no tendrán efecto allí donde el elemento está partido:

```

-----
Aquí aparecen |varias
-----

-----
palabras | enfatizadas
-----

```

Elementos reemplazados

Un elemento reemplazado es un elemento que es reemplazado por un contenido al que se apunta desde el elemento. P.ej., en HTML, el elemento 'IMG' es reemplazado por la imagen a la que apunta el atributo 'SRC'. Se puede suponer que los elementos reemplazados van con sus propias dimensiones intrínsecas. Si el valor de la propiedad '*width*' es '*auto*', se usará la anchura intrínseca como anchura del elemento. Si se especifica un valor diferente de '*auto*' en la Hoja de Estilo, se usará este valor, y el elemento reemplazado se escalará como corresponda (el método de escalado dependerá del tipo de medio). La propiedad '*height*' se usa de la misma manera. Los elementos reemplazados pueden ser en bloque o en línea.

La altura de las líneas

Todos los elementos tienen una propiedad '*line-height*' (altura de línea) que, en principio, da la altura total de una línea de texto. Para llegar a esa altura de línea, se añade espacio por encima y por debajo del texto de la línea. Por ejemplo, si el texto tiene 12pt de alto y el '*line-height*' está establecido en '14pt', se añade un espacio de 2pt, a saber, 1pt por encima y 1pt por debajo de la línea. Los elementos vacíos influyen en este cálculo igual que lo hacen los elementos con contenido. La diferencia entre el tamaño de la fuente y la altura de la línea se llama *interlineado*. A la mitad del interlineado se le denomina *semi-interlineado*. Después del formato, cada línea formará un *cuadro de línea* rectangular.

Si una línea de texto contiene secciones con valores diferentes de '*line-height*' (porque haya elementos en línea dentro de la línea), entonces cada una de esas secciones tiene su propio *semi-interlineado* por encima y por debajo. La altura del cuadro de línea va desde la parte superior de la sección más alta hasta la parte inferior de la sección más baja. Obsérvese que la parte superior y la parte inferior no corresponden necesariamente al elemento más alto, ya que los elementos pueden posicionarse verticalmente por medio de la propiedad '*vertical-align*'. Para formar un párrafo, cada cuadro de línea se inserta inmediatamente debajo de la línea anterior.

Obsérvese que los márgenes, bordes y rellenos por encima y por debajo de elementos en línea no reemplazados no influyen en la altura de la línea. En otras palabras: si el '*line-height*' es demasiado pequeño para el relleno o borde elegidos, se superpondrá con el texto de otras líneas.

Los elementos reemplazados de la línea (p.ej., imágenes) pueden agrandar el cuadro de línea si la parte superior del elemento reemplazado (es decir, incluyendo relleno, borde y margen) está por encima de la sección de texto más alta, o si su parte inferior está por debajo de la sección más baja. En el caso normal en que sólo haya un valor de *'line-height'* para todo el párrafo y no haya imágenes altas, la definición recién dada asegura que las líneas de base de líneas sucesivas estarán separadas exactamente por un *'line-height'*. Esto es importante cuando deben alinearse columnas de texto con fuentes distintas, por ejemplo en una tabla. Obsérvese que esto no imposibilita que los textos de dos líneas adyacentes se superpongan. El *'line-height'* puede ser menor que la altura del texto, en cuyo caso el interlineado será negativo. Esto es útil si se sabe que el texto no contiene descendentes (p.ej. porque sólo contiene letras mayúsculas), ya que entonces se pueden acercar más las líneas.

El lienzo

El lienzo es la parte de la superficie de dibujo del AU en que se representan los documentos. Ningún elemento estructural del documento corresponde al lienzo, lo cual plantea dos cuestiones a la hora de formatear un documento:

- ¿en base a qué deberían establecerse las dimensiones del lienzo?
- cuando el documento no cubre todo el lienzo, ¿cómo debería representarse esta área?

Una respuesta razonable a la primera pregunta es que la anchura inicial del lienzo se base en el tamaño de la ventana, pero CSS deja esta cuestión a merced del AU. También es razonable esperar que el AU cambie la anchura del lienzo cuando se cambie el tamaño de la ventana, pero esto también está fuera del alcance de CSS. Las extensiones HTML han sentado un precedente para la segunda cuestión: los atributos del elemento 'BODY' establecen el fondo de todo el lienzo. Con el fin de tener en cuenta las suposiciones de los diseñadores, CSS introduce una regla especial para averiguar el fondo del lienzo:

- Si el valor del *'background'* del elemento 'HTML' es distinto a *'transparent'*, entonces úsese el fondo especificado. Si no, úsese el valor del *'background'* del elemento 'BODY'. Si el valor resultante es *'transparent'*, la representación queda indefinida.

Esta regla permite lo siguiente:

```
<HTML STYLE="background: url(http://style.com/marmol.png)">
<BODY STYLE="background: red">
```

En este ejemplo, el lienzo será cubierto con "mármol". El fondo del elemento 'BODY' (que puede cubrir el lienzo completamente o no) será rojo.

Hasta que haya otros medios disponibles para hacer referencia al lienzo, se recomienda que las propiedades del lienzo se establezcan en el elemento 'BODY'.

Elementos BR

Las propiedades y valores de CSS no pueden describir el comportamiento del elemento 'BR'. En HTML, el elemento 'BR' especifica un salto de línea entre palabras. En efecto, el elemento se reemplaza por un salto de línea. Versiones futuras de CSS podrían manejar contenido añadido y reemplazado, pero los formateadores CSS deben tratar los 'BR' de forma especial.

4. Fuentes Descargables

Las fuentes cargables consisten en la posibilidad de incluir en el documento HTML la definición de las fuentes empleadas por si éstas no se encontraran disponibles en el sistema de destino. Dicha definición consiste en un fichero que se debe colocar en la máquina servidora, al igual que el documento o las imágenes. Los ficheros de fuentes, como otros muchos recursos, se pueden conseguir en Internet, pero el usuario no las puede grabar en disco (como puede hacer con el documento o las imágenes). Por otro lado, hay que tener en cuenta que las fuentes están sujetas a las "leyes de derechos de autor", por lo que, antes de utilizarlas dentro de nuestras páginas deberemos asegurarnos de tener permiso para hacerlo. Para que todo funcione correctamente, al servidor habrá que añadirle un nuevo tipo MIME [RFC2045] para que reconozca este tipo de archivos. El nuevo tipo es *application/font-tdpfr*, asociado a la extensión *.pfr* (para más información acerca de las fuentes véase [CHARSETS], [ISO10646], [UNICODE], [POSTSCRIPT]).

4.1. Cómo usar fuentes descargables.

Una vez se dispone del fichero de definición de fuentes, por ejemplo, *fuentes.pfr*, se pueden asociar al documento a través de un estilo, por medio de la palabra *fontdef*, por ejemplo:

```
<style type="text/css">
@fontdef url(http://www.gutenberg.org/fuentes.pfr)
</style>
```

o también con la etiqueta *<link>*:

```
<link rel=fontdef src = "http://www.gutenberg.org/fuentes.pfr">
```

Una vez que el fichero ha sido cargado desde el servidor donde están las páginas, imágenes, fuentes, etc., para utilizar estas fuentes, modificaremos el valor del atributo *face* de la etiqueta **, por ejemplo:

```
<font face = "Gutenberg">
Vamos a probar el tipo de letra Gutenberg
</font>
```

o bien definiendo un estilo CSS, utilizando la propiedad *font-family*:

```
<style type="text/css">
clasico { font-family: Gutenberg, sans-serif }
</style>
...
...
<div class="clasico">Este tipo de letra es la Gutenberg</div>
```

Este párrafo utilizará el tipo de fuente *Gutenberg* si está disponible, sino utilizará la *sansserif*. Podemos ver algunos ejemplos de fuentes descargables en la página de TrueDoc. He aquí una muestra de la fuente *Amelia* tomada directamente desde su servidor:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Las Fuentes dinámicas son descargadas con las páginas Web, del mismo modo que los ficheros de imágenes GIF [GIF], JPEG [JPEG] o PNG [PNG10]. Un navegador que pueda ver ficheros de fuentes *TrueDoc*, como el Netscape Communicator, renderiza las fuentes en pantalla. Por otro lado, los navegadores que no puedan visualizar este tipo de fuentes usaran una fuente alternativa presente en el sistema local del usuario.

4.2. Herramientas de diseño.

Dos programas que sirven para crear este tipo de ficheros son:

- *WebFont Wizard* de TrueDoc [TRUEDOC]
- *Typograph* de HexMac [HEXMAC]

También existe un *Plugin Font Composer* del navegador Netscape Navigator que también se puede usar. Al crear el fichero de definición de fuente, podemos especificar el dominio de la red en el que permitimos que se use dicha fuente.

5. Usabilidad y Accesibilidad con las Hojas de Estilo

Con todo lo visto hasta hora se puede concluir que el uso de las Hojas de Estilo se muestra como un método válido para dotar de altas cotas de usabilidad y accesibilidad a las páginas Web que diseñemos. Quizás falte resaltar algunos aspectos destacados en este sentido, no detallados anteriormente, ese es el propósito de este apartado.

La portabilidad es un aspecto muy destacado que se puede aproximar usando Hojas de Estilo. Diseñar interfaces flexibles y/o independientes de dispositivo nos permite maximizar la usabilidad de nuestras páginas Web. Este es el principio hacia dónde avanza el futuro de la definición de las CSS por parte del W3C.

En este sentido, iniciativas como las nuevas definiciones de perfiles específicos son el último paso en esta dirección. Por un lado se ha definido un perfil para dispositivos móviles [CSSMOBILE], basado en la definición de CSS2 pero que restringe ciertas características para que finalmente nos quede un conjunto mínimo de propiedades, valores, selectores y reglas en cascada aplicables en los dispositivos móviles o en los teléfonos de última generación.

De modo análogo se ha definido un perfil para TV [CSSTV], o mejor dicho para visualizar páginas Web directamente sobre la televisión. Este perfil se basa en la definición CSS2 y en un módulo de la definición de CSS3 (en concreto el módulo *CSS3 Module: Color* [CSS3COLOR]). Este perfil contiene la mayoría de la definición de CSS1, una parte de CSS2 y el módulo indicado de la definición de CSS3. En realidad este perfil para dispositivos de TV no es más que un subconjunto del perfil móvil.

Vislumbrando esta tendencia, vamos a considerar las Hojas de Estilo como una forma de lograr la portabilidad de nuestros contenidos a un amplio abanico de plataformas o dispositivos de naturaleza muy heterogénea. Todo ello conservando o dotando a las interfaces generadas de una gran usabilidad transversal e independiente de que plataforma se acabe usando para consumir los contenidos.



En cuanto a la accesibilidad, mencionar que el consorcio W3C inició a mediados de los 90 un área específica dedicada a la accesibilidad, creando la *Web Accessibility Initiative* (WAI). A su vez se creó la guía *Web Content Accessibility Guidelines* (WCAG) [WAI]. Entre otras pautas, las pautas 3 y 6 de esta guía promueven el uso de las Hojas de Estilo. En efecto, varias características de las Hojas de Estilo harán a la Web más usable y accesible para los usuarios con discapacidades, veamos las más destacadas:

- Permite diseñar documentos con un orden lógico en su estructura para después aplicar las Hojas de Estilo para lograr estilos de composición. Diseñando así, se permite que los documentos puedan ser leídos sin usar las Hojas de Estilo.
- Permite usar fuentes y colores del sistema para que las páginas se adapten a las necesidades y preferencias del usuario. Por ejemplo homogeneizar el uso de los colores de los enlaces usando las pseudo-clases *:link*, *:visited*, *:hover*, *:active*.
- Las propiedades para controlar la apariencia de las fuentes permiten a los autores eliminar las inaccesibles imágenes de texto renderizado.
- Las propiedades de posicionamiento permite a los autores eliminar los artilugios con el sistema de marcas (ej., imágenes invisibles) para forzar la composición.
- La semántica de las reglas *!important* indica que los usuarios con particulares requerimientos de presentación pueden suplantar las hojas de estilo del autor.
- El nuevo valor *'inherit'* (heredado) para todas las propiedades mejora la generalidad del funcionamiento en cascada y permite una más fácil y consistente sintonía en el estilo.

- El avanzado soporte de medios, incluyendo grupos de medios y los tipos de medios braille, de relieve y tty permitirán a los usuarios y autores confeccionar páginas para esos dispositivos.
- Las propiedades auditivas ofrecen control sobre la salida de voz y audio. Soportan las Hojas de Estilo auditivas que especifican cómo un documento sonará cuando es transformado a voz.
- Los selectores de atributos, la función 'attr()' y la propiedad 'content' brindan acceso al contenido alterno.
- Los contadores y la numeración de secciones y párrafos pueden mejorar la navegabilidad del documento y economizar espacio de sangrado (importante en los dispositivos braille). Las propiedades 'word-spacing' y 'text-indent' también eliminan la necesidad de usar espacios en blanco extras en el documento.

Para lograr todos los ambiciosos objetivos, respecto la usabilidad y la accesibilidad, que se propone con el uso de las Hojas de Estilo, tanto si usamos las CSS2 [W3CSTYLE], como con las CSS1, debemos seguir una serie de reglas de diseño. A continuación pasamos a detallar las principales reglas que se deberían considerar:

- **Compatibilidad hacia atrás y hacia adelante.** Las aplicaciones del usuario CSS2 serán capaces de entender las hojas de estilo CSS1. Las aplicaciones del usuario CSS1 podrán leer las Hojas de Estilo CSS2 y descartar las partes que no entienden. Además, las aplicaciones del usuario que no soporten CSS serán capaces de mostrar los documentos estilísticamente mejorados. Por supuesto, los efectos estilísticos hechos posible gracias al uso de CSS no serán procesados, pero todo el contenido será presentado.
- **Complementariedad con documentos estructurados.** Las Hojas de Estilo complementan los documentos estructurados (ej., HTML y aplicaciones XML) proveyendo información estilística del texto marcado. Debe ser fácil cambiar la hoja de estilo con poco o ningún impacto en el sistema de marcas.
- **Independencia del vendedor, la plataforma y el dispositivo.** Las Hojas de Estilo permiten a los documentos permanecer independientes del vendedor, la plataforma y el dispositivo. Las mismas Hojas de Estilo son también independientes del vendedor y la plataforma, pero CSS2 permite dirigir una Hoja de Estilo a un grupo de dispositivos (ej., impresoras).
- **Mantenibilidad.** Apuntando a una hoja de estilo desde los documentos, los responsables de los sitios en la Web pueden simplificar el mantenimiento y conservar un estilo y un efecto consistente a todo lo largo del sitio. Por ejemplo, si el color del fondo de las páginas de una organización cambia, sólo un archivo necesita ser cambiado.
- **Simplicidad.** CSS2 es más compleja que CSS1, pero sigue siendo un lenguaje de estilo simple que es humanamente legible y posible de escribir. Las propiedades CSS se mantienen independientes unas de otras en la medida de lo posible y generalmente sólo hay un modo de conseguir un efecto determinado.

- **Rendimiento de la red.** CSS proporciona una compacta codificación para presentar los contenidos. Comparado con los archivos de imágenes o de audio que son usados frecuentemente por los autores para conseguir ciertos efectos en el procesamiento, las Hojas de Estilo, la mayoría de las veces, disminuyen el tamaño del contenido. Además, menos conexiones de la red tienen que ser abiertas, lo cual ayuda a incrementar el rendimiento de la red.
- **Flexibilidad.** Las CSS pueden ser aplicadas al contenido de varias maneras. La característica clave es la capacidad de formar una cascada de estilos con la información especificada en la hoja de estilo predeterminada (aplicación del usuario), las hojas de estilo del usuario, las hojas de estilo vinculadas, el encabezamiento del documento y en los atributos de los elementos que forman el cuerpo del documento.
- **Riqueza.** Proporcionando a los autores un abundante juego de efectos de procesamiento, aumenta la riqueza de la Web como medio de expresión. Los diseñadores han estado ambicionando la funcionalidad comúnmente encontrada en los programas de edición y de presentaciones gráficas. Algunos de los efectos requeridos entran en conflicto con la independencia del dispositivo, pero CSS2 llega muy lejos satisfaciendo las demandas de los diseñadores.
- **Combinación con lenguajes alternativos.** El juego de propiedades de CSS descritas en el apéndice A conforman un sólido modelo de aplicación de formatos para presentaciones visuales y auditivas. Este modelo puede ser accedido mediante el lenguaje CSS, pero la combinación con otros lenguajes también es posible. Por ejemplo, un programa en JavaScript puede cambiar dinámicamente el valor de la propiedad 'color' de un determinado elemento.

6. Programas Asociados a las Hojas de Estilo

6.1. Editor: EDIPO.

EDIPO [EDIPO] es una herramienta creada principalmente para los usuarios de Internet que desean o requieren utilizar una hoja de estilos personal que se sobreponga a la propia de cualquier página Web, de manera que los sitios se adapten a sus necesidades de visualización.



Puede ser utilizado por cualquier sitio Web para ofrecer a sus usuarios la posibilidad de crear su propia hoja de estilos, de manera sencilla y sin tener conocimientos técnicos. Muy útil para aquellos que requieren o desean modificar la manera en que se presentan los contenidos Web para mejorar su legibilidad, como por ejemplo para las personas con deficiencias visuales o los desarrolladores que quieren revisar la accesibilidad de su sitio Web.

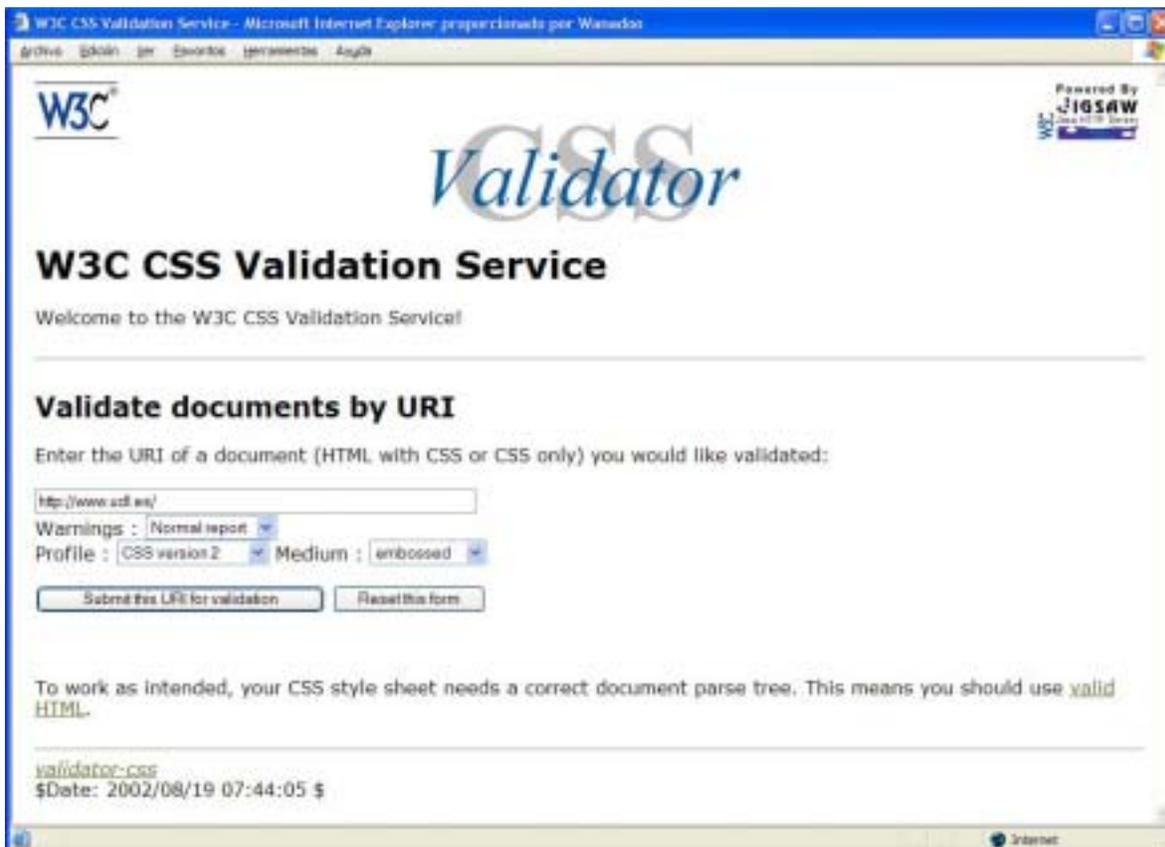
EDIPO genera una Hoja de Estilos en cascada siguiendo las indicaciones del usuario, para modificar determinadas opciones, que puede instalar en su ordenador para utilizarla como Hoja de Estilos personal.

EDIPO se presenta en dos versiones: EDIPO con marcos y EDIPO sin marcos, para aquellos usuarios cuyo navegador no soporta los marcos. Las dos versiones cuentan con una ayuda para solventar las dudas sobre su funcionamiento. Además, hay una documentación de usuario con información sobre para qué sirve, cómo se usa y algunas advertencias y recomendaciones. Una vez instalada la hoja de estilos en el ordenador del usuario, cuando navegue por Internet podrá ver las páginas Web con los colores que sean más prácticos para él, con su fuente preferida, y el tamaño de texto más adecuado a sus necesidades, etc.

EDIPO es software libre, lo que significa que cualquier persona puede redistribuirlo o modificarlo y contribuir a su desarrollo aportando mejoras, modificaciones, traducciones, corrección de errores, documentación, etc. Invitamos a todos a contribuir al desarrollo y diseño de EDIPO.

6.2. Validador: W3C CSS Validator.

El consorcio W3C dota a los diseñadores de esta herramienta de acceso público que permite validar el código CSS [W3CVAL] usado dentro de las páginas Web o en un fichero externo que defina el estilo. Está disponible directamente en la página Web escribiendo la dirección o URI del documento o descargándote una aplicación Java y ejecutando una llamada desde la línea de comandos. Por tanto es una herramienta accesible en cualquier plataforma que disponga de un navegador o de una máquina virtual de Java.



6.3. Navegadores: Compatibilidad con CSS.

Los creadores de Opera [OPERA] y por supuesto el W3C con su Amaya [AMAYA], siempre han incluido las funciones de las Hojas de Estilo. En el caso de Netscape Navigator [NETSCAPE] e Internet Explorer [EXPLORER], aunque sus versiones más nuevas se acercan a la idea de Opera o Amaya, parece que la política ha tenido un papel muy importante en la compatibilidad de dichos navegadores.

Por ejemplo, Internet Explorer siempre se ha negado incondicionalmente a permitir que el texto parpadee, y es algo muy comprensible, pero que ha creado ciertas discrepancias entre navegadores. En lo que respecta a Netscape, la secuela a la versión 4 tardó tanto en llegar que los desarrolladores se vieron estancados durante largo tiempo con un Navigator muy limitado en lo que se refiere a capacidades CSS; sin embargo, el motor Mozilla [MOZILLA] de la versión 6 ya acepta casi todas las reglas CSS. Para encontrar listas de compatibilidades de las propiedades y reglas en los diversos navegadores se pueden consultar las siguientes referencias: [W3SCHOOL], [PCCUA5].

7. Referencias

[AMAYA]

Navegador promovido por el W3C. Disponible en <http://www.w3.org/Amaya/>

[CHARSETS]

Valores registrados de juegos de caracteres. Lista de valores registrados de juegos de caracteres disponible en <http://www.iana.org/assignments/character-sets>.

[CSS3COLOR]

CSS3 Module: Color. T. Çelik and C. Lilley, 14 de Mayo de 2003. Disponible en <http://www.w3.org/TR/2003/CR-css3-color-20030514>.

[CSSMOBILE]

CSS Mobile Profile 1.0. Disponible en <http://www.w3.org/TR/css-mobile>

[CSSTV]

CSS TV Profile 1.0. Disponible en <http://www.w3.org/TR/css-tv>

[EDIPO]

Software editor de CSS EDIPO. Disponible en <http://www.sidar.org/edipo/index.php>

[EXPLORER]

Disponible en http://www.microsoft.com/windows/ie_intl/es/

[FLEX]

"FLEX: The Lexical Scanner Generator", Version 2.3.7, ISBN 1882114213

[GIF]

Disponible en <http://www.w3.org/Graphics/GIF/spec-gif87.txt>

[HEXMAC]

Disponible en <http://www.hexmac.com>

[HTML32]

"HTML 3.2 Reference Specification", Dave Raggett, Enero 14 de 1997. Disponible en <http://www.w3.org/TR/REC-html32.html>.

[HTML40]

"HTML 4.0 Specification", D. Raggett, A. Le Hors, I. Jacobs, Julio 8 de 1997. Disponible en <http://www.w3.org/TR/REC-html40/>. La recomendación define tres definiciones del tipo de documento: Strict, Transitional y Frameset, todas al alcance en la recomendación.

[HTMLDIN]

Manual de HTML Dinámico recomendado, G. Romero López. Disponible en <http://geneura.ugr.es/~gustavo/css/css.html>.

[ISO9899]

ISO/IEC 9899:1990 Lenguajes de programación -- C. Disponible en <http://www.iso.ch/cate/d17782.html>.

[ISO10646]

"Information Technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane", ISO/IEC 10646-1:1993. Disponible en <http://www.iso.ch/cate/d29819.html>

[JAVAMAN]

Manual recomendado disponible en <http://www.paratodos.8m.net/javas.htm>

[JAVASUN]

Disponible en <http://java.sun.com>

[JPEG]

Disponible en <http://www.jpeg.org/>

[MOZILLA]

Disponible en <http://www.mozilla.org/>

[NEGOT]

"Transparent Content Negotiation in HTTP", K. Holtman, A. Mutz, Marzo 9 de 1997.
Disponible en <http://gewis.win.tue.nl/~koen/conneg/draft-ietf-http-negotiation-01.html>.

[NETSCAPE]

Disponible en <http://www.aola.com/netscape/browser/index.adp>

[OPERA]

Disponible en <http://www.opera.com/>

[PCCUA5]

CSS: webs con estilo. PC-Cuadernos nº5, Marzo 2002, ref. 489. ISBN 2912954894.
Disponible en <http://pc-cuadernos.com/>

[PNG10]

"PNG (Portable Network Graphics) Specification, Version 1.0 specification", T. Boutell ed., Octubre 1 de 1996. Disponible en <http://www.w3.org/pub/WWW/TR/REC-png-multi.html>.

[POSTSCRIPT]

"The PostScript Language Reference Manual", Segunda Edición, Adobe Systems, Inc., Addison-Wesley Publishing Co., Diciembre 1990.

[RFC1630]

"Universal Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web", T. Berners-Lee, Junio 1994. Disponible en <http://www.w3.org/Addressing/rfc1630.txt>.

[RFC1738]

"Uniform Resource Locators", T. Berners-Lee, L. Masinter y M. McCahill, Diciembre 1994. Disponible en <http://www.w3.org/Addressing/rfc1738.txt>.

[RFC1766]

"Tags for the Identification of Languages", H. Alvestrand, Marzo 1995. Disponible en <http://www.ietf.org/rfc/rfc1766.txt>.

[RFC1808]

"Relative Uniform Resource Locators", R. Fielding, Junio 1995. Disponible en <http://www.w3.org/Addressing/rfc1808.txt>.

[RFC1866]

"HyperText Markup Language 2.0", T. Berners-Lee y D. Connolly, Noviembre 1995. Disponible en http://www.w3.org/MarkUp/html-spec/html-spec_toc.html.

[RFC1942]

"HTML Tables", Dave Raggett, Mayo 1996. Disponible en <http://www.ietf.org/rfc/rfc1942.txt>.

[RFC2045]

"Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed y N. Borenstein, Noviembre 1996. Disponible en <http://www.ietf.org/rfc/rfc2045.txt>. Observe que esta RFC deja obsoletas a RFC1521, RFC1522 y RFC1590.

[RFC2068]

"HTTP Version 1.1 ", R. Fielding, J. Gettys, J. Mogul, H. Frystyk Nielsen y T. Berners-Lee, Enero 1997. Disponible en <http://www.ietf.org/rfc/rfc2068.txt>.

[RFC2070]

"Internationalization of the HyperText Markup Language", F. Yergeau, G. Nicol, G. Adams, y M. Dürst, Enero 1997. Disponible en <http://www.ietf.org/rfc/rfc2070.txt>.

[RFC2318]

"The text/css Media Type", H. Lie, B. Bos, C. Lilley, Marzo 1998. Disponible en <http://www.ietf.org/rfc/rfc2318.txt>.

[RFC2396]

T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifiers (URI): Generic Syntax. IETF RFC 2396. Disponible en <http://www.ietf.org/rfc/rfc2396.txt>.

[TRUEDOC]

Disponible en <http://www.truedoc.com>

[UNICODE]

"The Unicode Standard: Version 2.0", The Unicode Consortium, Addison-Wesley Developers Press, 1996. Disponible en <http://www.unicode.org/>.

[W3C]

Disponible en <http://w3c.org/>

[W3CSTYLE]

Recurso del W3C sobre hojas de estilo en la web. Disponible en <http://www.w3.org/pub/WWW/Style>.

[W3CVAL]

Página Web de acceso al validador CSS del W3C. Disponible en <http://jigsaw.w3.org/css-validator/>

[W3SCHOOL]

Curso básico de CSS. Disponible en <http://www.w3schools.com/css/default.asp/>

[WAI]

"WAI Accessibility Guidelines: Page Authoring" para el diseño de documentos accesibles están disponibles en: <http://www.w3.org/TR/WD-WAI-PAGEAUTH>.

[XML10]

"Extensible Markup Language (XML) 1.0" T. Bray, J. Paoli, C.M. Sperberg-McQueen, editores, Febrero 10 de 1998. Disponible en <http://www.w3.org/TR/REC-xml/>.

[YACC]

"YACC - Yet another compiler compiler", S. C. Johnson, Technical Report, Murray Hill, 1975.

Apéndice A

Propiedades de CSS (Nivel 1)

Ferran Perdrix Sapiña



Grupo GRIHO (UdL)
C/Jaume II nº 69
ferran@griho.net



Índice

A1. Notación para los valores de las propiedades	A4
A2. Propiedades de fuente	A4
A2.1. Emparejamiento de fuentes.	A5
A2.2. 'font-family'.	A6
A2.3. 'font-style'.	A7
A2.4. 'font-variant'.	A7
A2.5. 'font-weight'.	A8
A2.6. 'font-size'.	A10
A2.7. 'font'.	A11
A3. Propiedades de color y fondo	A11
A3.1. 'color'.	A11
A3.2. 'background-color'.	A12
A3.3. 'background-image'.	A12
A3.4. 'background-repeat'.	A12
A3.5. 'background-attachment'.	A13
A3.6. 'background-position'.	A13
A3.7. 'background'.	A14
A4. Propiedades de texto	A15
A4.1. 'word-spacing'.	A15
A4.2. 'letter-spacing'.	A15
A4.3. 'text-decoration'.	A16
A4.4. 'vertical-align'.	A16
A4.5. 'text-transform'.	A17
A4.6. 'text-align'.	A18
A4.7. 'text-indent'.	A18
A4.8. 'line-height'.	A18
A5. Propiedades de cuadro	A19
A5.1. 'margin-top'.	A19
A5.2. 'margin-right'.	A19
A5.3. 'margin-bottom'.	A20
A5.4. 'margin-left'.	A20
A5.5. 'margin'.	A20
A5.6. 'padding-top'.	A21
A5.7. 'padding-right'.	A21
A5.8. 'padding-bottom'.	A21
A5.9. 'padding-left'.	A22
A5.10. 'padding'.	A22
A5.11. 'border-top-width'.	A22

A5.12. 'border-right-width'.	A23
A5.13. 'border-bottom-width'.	A23
A5.14. 'border-left-width'.	A23
A5.15. 'border-width'.	A23
A5.16. 'border-color'.	A24
A5.17. 'border-style'.	A24
A5.18. 'border-top'.	A25
A5.19. 'border-right'.	A26
A5.20. 'border-bottom'.	A26
A5.21. 'border-left'.	A26
A5.22. 'border'.	A26
A5.23. 'width'.	A27
A5.24. 'height'.	A27
A5.25. 'float'.	A28
A5.26. 'clear'.	A28
A6. Propiedades de clasificación	A29
A6.1. 'display'.	A29
A6.2. 'white-space'.	A29
A6.3. 'list-style-type'.	A30
A6.4. 'list-style-image'.	A30
A6.5. 'list-style-position'.	A30
A6.6. 'list-style'.	A31
A7. Referencias	A32

A1. Notación para los valores

Las hojas de estilo influyen sobre la presentación de los documentos asignando valores a propiedades de estilo. Este apéndice enumera las propiedades de estilo definidas y la lista correspondiente de sus posibles valores en CSS (nivel 1).

En todo este apéndice, los valores permitidos para cada propiedad se enumeran con una sintaxis similar a la siguiente:

Valor: N | NW | NE
Valor: [<longitud> | thick | thin]{1,4}
Valor: [<nombre-de-familia> ,]* <nombre-de-familia>
Valor: <url> ? <color> [/ <color>]?
Valor: <url> || <color>

Los valores entre "<" y ">" dan un tipo de valor. Los tipos más comunes son <longitud>, <porcentaje>, <url>, <número> y <color>; éstos se describen en el apéndice B. Los tipos más especializados (p.ej., <familia-de-fuentes> y <estilo-de-borde>) se describen bajo la propiedad correspondiente. Otras palabras son palabras clave que deben aparecer literalmente, sin comillas. La barra inclinada (/) y la coma (,) también deben aparecer literalmente.

Cuando aparezcan varias cosas yuxtapuestas, deben incluirse todas ellas en el orden especificado. Una barra (|) separa alternativas: debe especificarse una de ellas. Una barra doble (A || B) significa que debe especificarse A, B o ambas, en cualquier orden. Los corchetes ([]) se usan para agrupar. La yuxtaposición es más fuerte que la doble barra, y la doble barra es más fuerte que la barra. Así, "a b | c || d e" es equivalente a "[a b] | [c || [d e]]".

Cada tipo, palabra clave, o grupo entre corchetes, puede ir seguido por uno de los siguientes modificadores:

- Un asterisco (*) indica que el tipo, palabra o grupo precedente se repite cero o más veces.
- Un signo más (+) indica que el tipo, palabra o grupo precedente se repite una o más veces.
- Un signo de interrogación (?) indica que el tipo, palabra o grupo precedente es opcional.
- Un par de números entre llaves ({A,B}) indica que el tipo, palabra o grupo precedente, se repite al menos A veces y como mucho B veces.

A2. Propiedades de fuente

El establecimiento de propiedades de fuente será uno de los usos más frecuente de las hojas de estilo. Desgraciadamente, no existe una taxonomía bien definida y universalmente aceptada para clasificar las fuentes, y algunos términos que se aplican a una familia tipográfica pueden no ser apropiados para otras. P.ej., para referirse a texto inclinado se utiliza normalmente el término *'italic'*, pero también pueden utilizarse otros términos como *Oblique*, *Slanted*, *Incline*, *Cursive* o *Kursiv*.

Por tanto, no es un problema sencillo aplicar propiedades típicas de selección de fuentes a una fuente específica. CSS nivel 1 define las propiedades *'font-family'*, *'font-style'*, *'font-variant'* y *'font-weight'*, *'font-size'*, *'font'*.

A2.1. Emparejamiento de fuentes.

Al no haber una taxonomía universal aceptada para las propiedades de las fuentes, el emparejamiento entre propiedades y fuentes tipográficas debe hacerse con cuidado. Las propiedades se emparejan en un orden bien definido para asegurarse de que los resultados de este proceso de emparejamiento sean tan consistentes entre los diversos Agentes de Usuarios (AAUU) como sea posible (suponiendo que se ponga a la disposición de todos ellos la misma biblioteca de fuentes tipográficas).

El emparejamiento por parte de un AU sigue el siguiente algoritmo:

- El Agente de Usuario hace (o accede a) una base de datos de todas las propiedades relevantes con respecto a CSS1 de todas las fuentes de cuya existencia sabe el AU. El AU puede saber de la existencia de una fuente porque haya sido instalada localmente o porque haya sido previamente descargada de la Web. Si hay dos fuentes que tengan exactamente las mismas propiedades, una de ellas se descarta.
- En un elemento dado, y para cada carácter de ese elemento, el AU integra todas las propiedades de fuente aplicables a ese elemento. A partir del conjunto completo de propiedades, el AU utiliza la propiedad *'font-family'* para elegir una familia tipográfica tentativa. El resto de las propiedades se prueban con la familia de acuerdo con los criterios de emparejamiento descritos para cada propiedad. Si hay emparejamientos para todas las propiedades restantes, entonces ésa es la fuente emparejada con el elemento dado.
- Si no hay una fuente emparejada dentro de la familia tipográfica procesada en el paso 2, y si hay una familia tipográfica (*'font-family'*) alternativa en el conjunto de fuentes, entonces se repite el paso 2 con la siguiente familia tipográfica alternativa.
- Si hay una fuente emparejada, pero no contiene un signo para el carácter considerado, y si hay una familia tipográfica alternativa en el conjunto de fuentes, entonces se repite el paso 2 con la siguiente familia tipográfica alternativa.
- Si no hay fuente dentro de la familia seleccionada en 2, entonces se usa la familia tipográfica por defecto del AU y se repite el paso 2, usando el mejor emparejamiento que pueda lograrse con la fuente por defecto.

Este algoritmo puede optimizarse para evitar tener que comprobar las propiedades CSS1 con cada carácter. Las reglas de emparejamiento de cada propiedad mencionada en el paso (2) son las siguientes:

- 1) Se comprueba *'font-style'* en primer lugar. *'italic'* se satisfará si hay o bien una fuente en la base de datos de fuentes del AU etiquetada con la palabra clave CSS *'italic'* (preferentemente) o bien con *'oblique'*. De otro modo los valores deben coincidir exactamente o *font-style* fallará.

- 2) A continuación se comprueba *'font-variant'*. *'normal'* se empareja con una fuente que no esté etiquetada como *'small-caps'* (versalitas). *'small-caps'* se empareja (1) con una fuente etiquetada como *'small-caps'*, (2) con una fuente en la que las versalitas sean sintetizadas, o (3) por una fuente en la que todas las letras minúsculas estén reemplazadas por letras mayúsculas. Una fuente de versalitas puede ser sintetizada escalando electrónicamente las letras mayúsculas de una fuente normal.
- 3) A continuación se comprueba *'font-weight'*. No fallará nunca (ver *'font-weight'* más abajo).
- 4) *'font-size'* debe emparejarse dentro de un margen de tolerancia dependiente del AU. Normalmente, los tamaños de las fuentes escalables se redondean al píxel más cercano, mientras que la tolerancia de fuentes de mapas de bits podría ser tan grande como del 20%. Los cálculos ulteriores, p.ej., usando *'em'* en otras propiedades, se basarán en el valor del *'font-size'* realmente utilizado, no en el que ha sido especificado.

A2.2. *'font-family'*.

Valor: [[<nombre-de-familia> | <familia-genérica>],]* [<nombre-de-familia> | <familia-genérica>]

Inicial: depende del AU

Se aplica a: todos los elementos

Se hereda: sí

Valores porcentuales: N/A

Esta propiedad es una lista de nombres de familias tipográficas y/o nombres de familias genéricas ordenadas por prioridad. A diferencia de la mayoría de las demás propiedades CSS1, los valores se separan por comas para indicar que son alternativos:

```
BODY { font-family: gill, helvetica, sans-serif }
```

Hay dos tipos de valores en la lista:

<nombre-de-familia>

El nombre de una familia tipográfica para elegir. En el último ejemplo, "gill" y "helvetica" son familias tipográficas.

<familia-genérica>

En el ejemplo anterior, el último valor es un nombre de familia genérica. Se definen las siguientes familias genéricas:

- *'serif'* (p.ej. Times)
- *'sans-serif'* (p.ej. Helvetica)
- *'cursive'* (p.ej. Zapf-Chancery)
- *'fantasy'* (p.ej. Western)
- *'monospace'* (p.ej. Courier)

Se recomienda a los diseñadores de hojas de estilo ofrecer una familia tipográfica genérica como última alternativa. Los nombres de fuentes que contengan espacios en blanco deberían entrecomillarse:

```
BODY { font-family: "new century schoolbook", serif }
```

```
<BODY STYLE="font-family: 'My own font', fantasy">
```

Si se omiten las comillas, no se tienen en cuenta los caracteres de espacio que haya antes y después del nombre de la fuente, y cualquier secuencia de caracteres de espacio dentro del nombre de la fuente se convierte a un único espacio.

A2.3. 'font-style'.

Valor: normal | italic | oblique

Inicial: normal

Se aplica a: todos los elementos

Se hereda: sí

Valores porcentuales: N/A

La propiedad *'font-style'* elige entre letra normal (a veces llamada "romana" o "recta"), itálica y oblicua dentro de una familia tipográfica. Un valor *'normal'* selecciona una fuente clasificada como 'normal' en la base de datos de fuentes del AU, mientras que *'oblique'* selecciona una fuente marcada como *'oblique'*. Un valor *'italic'* selecciona una fuente que está marcada como *'italic'*, o, si ésta no está disponible, una marcada como *'oblique'*. La fuente marcada como *'oblique'* en la base de datos de fuentes del AU en realidad puede haber sido generada electrónicamente inclinando una fuente normal. Las fuentes que incluyan las palabras Oblique, Slanted o Incline en su nombre estarán normalmente marcadas como *'oblique'* en la base de datos de fuentes del AU. Las fuentes que incluyan las palabras Italic, Cursive o Kursiv en su nombre estarán normalmente marcadas como *'italic'*.

```
H1, H2, H3 { font-style: italic }
H1 EM { font-style: normal }
```

En este ejemplo, el texto enfatizado contenido en un título 'H1' aparecerá con letra normal.

A2.4. 'font-variant'.

Valor: normal | small-caps

Inicial: normal

Se aplica a: todos los elementos

Se hereda: sí

Valores porcentuales: N/A

Otro tipo de variación dentro de una familia tipográfica son las versalitas. En una fuente de letras versalitas las letras minúsculas son similares a las mayúsculas, pero de menor tamaño y con proporciones ligeramente distintas. La propiedad *'font-variant'* selecciona esa fuente. Un valor *'normal'* selecciona una fuente que no es de letras versalitas; *'small-caps'* selecciona una fuente de versalitas. En CSS1 es suficiente (pero no necesario) que la fuente de versalitas se cree tomando una fuente normal y reemplazando las letras minúsculas por caracteres de caja alta escalados. Como último recurso, se podrán utilizar letras mayúsculas como sustitutas de una fuente de versalitas. El siguiente ejemplo produce un elemento 'H3' en versalitas, con palabras enfatizadas en versalitas oblicuas:

```
H3 { font-variant: small-caps }
EM { font-style: oblique }
```

Puede haber otras variantes en la familia tipográfica, como por ejemplo fuentes con numerales de estilo antiguo, numerales versalitas, letras condensadas o expandidas, etc. CSS1 no tiene propiedades para seleccionar éstas. *CSS1 básico:* en el caso de que esta propiedad provoque que el texto se transforme a mayúsculas, se aplicarán las mismas consideraciones que para la propiedad *'text-transform'*.

A2.5. 'font-weight'.

Valor: normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900

Inicial: normal

Se aplica a: todos los elementos

Se hereda: sí

Valores porcentuales: N/A

La propiedad *'font-weight'* selecciona el peso de la fuente. Los valores '100' a '900' forman una secuencia ordenada, en la que cada número indica un peso que es al menos tan pesado como su predecesor.

La palabra clave *'normal'* es sinónima de '400', y *'bold'* (negrita) es sinónima de '700'. Se ha demostrado que otras palabras clave distintas de *'normal'* y *'bold'* se confunden a menudo con nombres de fuentes, y por ello se ha elegido una escala numérica para la lista de 9 valores.

```
P { font-weight: normal } /* 400 */
H1 { font-weight: 700 } /* bold */
```

Los valores *'bolder'* y *'lighter'* seleccionan pesos de fuente que son relativos al peso heredado del padre:

```
STRONG { font-weight: bolder }
```

Los elementos hijos heredan el peso resultante, no el valor de la palabra clave. Las fuentes (los datos de las fuentes) tienen normalmente una o más propiedades cuyos valores son nombres que describen el "peso" de una fuente. No hay un significado universal aceptado para estos nombres de pesos. Su papel principal es distinguir tipos de letra de distinto peso dentro de la misma familia tipográfica. El uso de unas familias a otras es bastante variable; por ejemplo, una fuente que uno podría pensar que es negrita (*bold*) podría ser descrita como *Regular*, *Roman*, *Book*, *Medium*, *Semi-* o *DemiBold*, *Bold* o *Black*, dependiendo de lo gruesa que sea la letra "normal" de la fuente dentro del diseño.

Al no haber un uso estándar de nombres, los valores de la propiedad de peso en CSS1 se dan según una escala numérica en la que el valor '400' (*'normal'*) se corresponde con la fuente de texto "normal" de esa familia. El nombre de peso asociado a esa fuente normalmente será *Book*, *Regular*, *Roman*, *Normal* o a veces *Medium*. Con la asociación de valores de peso numéricos a las otras denominaciones de pesos dentro de cada familia sólo se pretende preservar el orden de grosores dentro de esa familia. Sin embargo, la siguiente heurística nos dice cómo se hace la asignación en los casos típicos:

- Si la fuente tipográfica ya usa una escala tipográfica con nueve valores (como p.ej. hace OpenType), los pesos de la fuente deberían aplicarse directamente.
- Si hay al mismo tiempo una fuente marcada como *Medium* y otra marcada como *Book*, *Regular*, *Roman* o *Normal*, entonces *Medium* se asigna normalmente al valor '500'.
- La fuente marcada como "Bold" corresponderá normalmente al valor de peso '700'.

- Si hay menos de nueve pesos en la familia, el algoritmo por defecto para rellenar los "huecos" es el siguiente: si '500' está sin asignar, se le asignará la misma fuente que para '400'. Si cualquiera de los valores '600', '700', '800' o '900' sigue sin estar asignado, se asignan a la misma fuente que la siguiente palabra clave más pesada asignada, si la hay, o a la de la siguiente más ligera en caso contrario. Si cualquiera de los valores '300', '200' o '100' sigue sin estar asignado, se asigna a la misma fuente que la siguiente palabra clave más ligera, si la hay, o a la de la siguiente más pesada en caso contrario.

Los siguientes dos ejemplos ilustran el proceso. Supónganse cuatro pesos en la familia "Ejemplo 1", de más ligero a más pesado: *Regular*, *Medium*, *Bold*, *Heavy*. Y supónganse seis pesos en la familia "Ejemplo 2": *Book*, *Medium*, *Bold*, *Heavy*, *Black*, *ExtraBlack*. Obsérvese cómo en el segundo ejemplo se ha decidido dejar "Ejemplo 2 Extrablack" sin asignar.

Fuentes disponibles	Asignaciones	Relleno de huecos
"Example1 Regular"	400	100, 200, 300
"Example1 Medium"	500	
"Example1 Bold"	700	600
"Example1 Heavy"	800	900

Fuentes disponibles	Asignaciones	Relleno de huecos
"Example2 Book"	400	100, 200, 300
"Example2 Medium"	500	
"Example2 Bold"	700	600
"Example2 Heavy"	800	
"Example2 Black"	900	
"Example2 ExtraBlack"	(none)	

Ya que la intención de las palabras clave *'bolder'* y *'lighter'* es hacer más pesada o más ligera la fuente dentro de la familia, y teniendo en cuenta que una familia puede no tener fuentes asignadas a todos los valores simbólicos de peso, el emparejamiento de *'bolder'* se hace con la siguiente fuente más pesada disponible en el cliente dentro de la misma familia, y el emparejamiento de *'lighter'* se hace con la siguiente fuente más ligera dentro de la familia. Para ser más precisos, el significado de las palabras clave relativas *'bolder'* y *'lighter'* es el siguiente:

- *'bolder'* selecciona el siguiente peso que esté asignado a una fuente que sea más pesada que la heredada. Si no existe tal peso, el resultado es simplemente el siguiente valor numérico más pesado (y la fuente permanece sin cambiar), a menos que el valor heredado fuera '900', en cuyo caso el peso resultante también sería '900'.
- *'lighter'* es similar, pero funciona en el sentido opuesto: selecciona la siguiente palabra clave más ligera con una fuente diferente de la heredada, a menos que no exista tal fuente, en cuyo caso selecciona el siguiente valor numérico más ligero (y mantiene la fuente sin cambios).

No hay ninguna garantía de que vaya a haber una fuente más pesada para cada uno de los valores de *'font-weight'*; por ejemplo, algunas familias tipográficas pueden tener solamente una fuente normal y una fuente negrita (bold), otras pueden tener nueve fuentes de pesos diferentes. La única garantía es que una fuente con un valor dado no será menos pesada que fuentes con valores más ligeros.

A2.6. 'font-size'.

Valor: <tamaño-absoluto> | <tamaño-relativo> | <longitud> | <porcentaje>

Inicial: medium

Se aplica a: todos los elementos

Se hereda: sí

Valores porcentuales: relativos al tamaño de fuente del elemento padre

<tamaño-absoluto>

Una palabra clave de tipo <tamaño-absoluto> es un índice a una tabla de tamaños de fuente calculados y guardados por el AU. Los valores posibles son: [xx-small | x-small | small | medium | large | x-large | xx-large]. Para la pantalla de una computadora se sugiere un factor de escala de 1,5 entre índices sucesivos; si la fuente '*medium*' es de 10pt, la fuente '*large*' podría ser de 15pt. Medios diferentes pueden necesitar diferentes factores de escala. Por otra parte, a la hora de calcular la tabla, el AU debería tener en cuenta la calidad y disponibilidad de las fuentes. La tabla puede ser distinta de una familia a otra.

<tamaño-relativo>

Una palabra clave de tipo <tamaño-relativo> se interpreta como relativa con respecto a la tabla de tamaños de fuente y al tamaño de fuente del elemento padre. Los valores posibles son: [larger | smaller]. Por ejemplo, si el elemento padre tiene un tamaño de fuente '*medium*', un valor '*larger*' hará que el tamaño de fuente del elemento considerado sea '*large*'.

Si el tamaño del elemento padre no está cercano a una de las entradas de la tabla, el AU es libre de interpolar entre entradas de la tabla o redondear a la más cercana. El AU puede tener que extrapolar valores de la tabla si el valor numérico rebasa los límites de las palabras clave. Los valores de longitud y los porcentuales no deberían tener en cuenta la tabla de tamaños de fuente cuando se calcule el tamaño del elemento. No se permiten valores negativos.

En todas las demás propiedades, los valores de longitudes expresados en '*em*' y '*ex*' se refieren al tamaño de fuente del elemento actual. En la propiedad '*font-size*', estas unidades de longitud se refieren al tamaño de fuente del elemento padre. Obsérvese que una aplicación puede reinterpretar un tamaño explícito, dependiendo del contexto. Por ejemplo, dentro de una escena de realidad virtual una fuente puede obtener un tamaño diferente debido a la distorsión de la perspectiva.

Ejemplos:

```
P { font-size: 12pt; }
BLOCKQUOTE { font-size: larger }
EM { font-size: 150% }
EM { font-size: 1.5em }
```

Si se utiliza el factor de escala sugerido de 1.5, las tres últimas declaraciones son idénticas.

A2.7. 'font'.

Valor: [<font-style> || <font-variant> || <font-weight>]? <font-size> [/ <line-height>]? <font-family>

Inicial: no definido para propiedades abreviadas

Se aplica a: todos los elementos

Se hereda: sí

Valores porcentuales: sólo permitidos para <font-size> y <line-height>

La propiedad 'font' es una propiedad abreviada para establecer las propiedades '*font-style*', '*font-variant*', '*font-weight*', '*font-size*', '*line-height*' y '*font-family*' en un mismo lugar de una hoja de estilo. La sintaxis de esta propiedad se basa en la notación abreviada tipográfica tradicional para establecer múltiples propiedades relacionadas con los tipos de letra. Para una definición de los valores iniciales y permitidos, véanse las propiedades anteriormente definidas. Las propiedades para las cuales no se dan valores se establecen a su valor inicial.

```
P { font: 12pt/14pt sans-serif }
P { font: 80% sans-serif }
P { font: x-large/110% "new century schoolbook", serif }
P { font: bold italic large Palatino, serif }
P { font: normal small-caps 120%/120% fantasy }
```

En la segunda regla, el valor porcentual del tamaño de fuente ('80%') se refiere al tamaño de fuente del elemento padre. En la tercera regla, el porcentaje de altura de línea se refiere al tamaño de fuente del propio elemento. En las tres primeras reglas del ejemplo anterior, las propiedades '*font-style*', '*font-variant*' y '*font-weight*' no se mencionan explícitamente, lo cual significa que las tres se establecen a su valor inicial ('*normal*'). La cuarta regla establece '*font-weight*' en '*bold*', '*font-style*' en '*italic*' e, implícitamente, establece '*font-variant*' en '*normal*'. La quinta regla establece '*font-variant*' ('*small-caps*'), '*font-size*' (120% de la fuente del padre), '*line-height*' (120% del tamaño de la fuente) y '*font-family*' ('*fantasy*'). Se deduce que la palabra clave '*normal*' se aplica a las dos restantes propiedades: '*font-style*' y '*font-weight*'.

A3. Propiedades de color y fondo

Estas propiedades describen el color (a menudo llamando en inglés *foreground color*, color de primer plano) y el fondo (*background*) de un elemento (es decir, la superficie sobre la cual se representa el contenido). Se puede establecer un color de fondo y/o una imagen de fondo. También se puede establecer la posición de la imagen, si la imagen se repite y cómo se repite, y si está fija o si se mueve con respecto al lienzo (es decir, si hace *scroll*).

La propiedad '*color*' se hereda normalmente. Las propiedades de fondo no se heredan, pero el fondo del elemento padre se verá por defecto a través de los hijos, ya que el valor inicial de '*background-color*' es transparente ('*transparent*').

A3.1. 'color'.

Valor: <color>

Inicial: depende del UA

Se aplica a: todos los elementos

Se hereda: sí

Valores porcentuales: N/A

Esta propiedad describe el color del texto de un elemento (color de primer plano o *foreground color*). Hay distintas maneras de especificar el color rojo:

```
EM { color: red }           /* lenguaje natural */
EM { color: rgb(255,0,0) }  /* RGB con rango 0-255 */
```

A3.2. 'background-color'.

Valor: <color> | transparent
Inicial: transparent
Se aplica a: todos los elementos
Se hereda: no
Valores porcentuales: N/A

Esta propiedad establece el color de fondo de un elemento.

```
H1 { background-color: #F00 }
```

A3.3. 'background-image'.

Valor: <url> | none
Inicial: none
Se aplica a: todos los elementos
Se hereda: no
Valores porcentuales: N/A

Esta propiedad establece la imagen de fondo de un elemento. Cuando se establece una imagen de fondo, también debería establecerse un color de fondo que se usará en el caso de que la imagen no esté disponible. Cuando la imagen esté disponible, se superpone al color de fondo.

```
BODY { background-image: url(marmol.gif) }
P { background-image: none }
```

A3.4. 'background-repeat'.

Valor: repeat | repeat-x | repeat-y | no-repeat
Inicial: repeat
Se aplica a: todos los elementos
Se hereda: no
Valores porcentuales: N/A

Si se especifica una imagen, el valor de *'background-repeat'* determina si la imagen se repite y cómo se repite.

Un valor de *'repeat'* significa que la imagen se repite tanto horizontal como verticalmente. El valor *'repeat-x'* (*'repeat-y'*) hace que la imagen se repita horizontalmente (verticalmente), creando una sola banda de imágenes de un lado a otro (de arriba a abajo). Con un valor *'no-repeat'* la imagen no se repite.

```
BODY {
  background: red url(pendiente.gif);
  background-repeat: repeat-y;
}
```

En el ejemplo, la imagen sólo se repetirá verticalmente.

A3.5. 'background-attachment'.

Valor: scroll | fixed
Inicial: scroll
Se aplica a: todos los elementos
Se hereda: no
Valores porcentuales: N/A

Si se especifica una imagen de fondo, el valor de '*background-attachment*' determina si ésta está fija con relación al lienzo, o si se mueve (hace *scroll*) junto con el contenido.

```
BODY {
  background: red url(pendiente.gif);
  background-repeat: repeat-y;
  background-attachment: fixed;
}
```

Los AAUU pueden tratar '*fixed*' como '*scroll*'. Sin embargo, se recomienda que interpreten '*fixed*' correctamente, al menos en los elementos HTML y BODY, ya que no es posible para un autor especificar una imagen de fondo únicamente para los *browsers* que soporten '*fixed*'.

A3.6. 'background-position'.

Valor: [<porcentaje> | <longitud>]{1,2} | [top | center | bottom] || [left | center | right]
Inicial: 0% 0%
Se aplica a: elementos en bloque y elementos reemplazados
Se hereda: no
Valores porcentuales: se refieren al tamaño del propio elemento

Si se ha especificado una imagen de fondo, el valor de '*background-position*' especifica la posición inicial.

Con un par de valores '*0% 0%*', la esquina superior izquierda de la imagen se coloca en la esquina superior izquierda del cuadro que rodea al contenido del elemento (es decir, no la del cuadro que rodea a relleno, borde y margen). Un par de valores '*100% 100%*' coloca la esquina inferior derecha de la imagen en la esquina inferior derecha del elemento. Con un par de valores '*14% 84%*', el punto que está al 14% de la izquierda de la imagen y al 84% de su parte superior se coloca en el punto que está al 14% de la izquierda del elemento y al 84% de su borde superior. Con un par de valores de '*2cm 2cm*', la esquina superior izquierda de la imagen se coloca a 2 cm a la derecha y 2 cm por debajo de la esquina superior izquierda del elemento.

Si sólo se da un valor porcentual o una longitud, este valor sólo establece la posición horizontal, y la posición vertical será el 50%. Si se dan dos valores, la posición horizontal corresponde al primero. Se permite combinar longitudes con valores porcentuales, p.ej., '*50% 2cm*'. Se permiten posiciones negativas. También se pueden usar como valores palabras clave que indican la posición de la imagen de fondo. Las palabras clave no pueden combinarse con valores porcentuales ni con longitudes. Las combinaciones posibles de palabras clave y sus interpretaciones son las siguientes:

- 'top left' y 'left top' significan ambas lo mismo que '0% 0%'.
- 'top', 'top center' y 'center top' significan lo mismo que '50% 0%'.
- 'right top' y 'top right' significan lo mismo que '100% 0%'.
- 'left', 'left center' y 'center left' significan lo mismo que '0% 50%'.
- 'center' y 'center center' significan lo mismo que '50% 50%'.
- 'right', 'right center' y 'center right' significan lo mismo que '100% 50%'.
- 'bottom left' y 'left bottom' significan lo mismo que '0% 100%'.
- 'bottom', 'bottom center' y 'center bottom' significan lo mismo que '50% 100%'.
- 'bottom right' y 'right bottom' significan lo mismo que '100% 100%'.

Ejemplos:

```
BODY { background: url(banner.jpeg) right top } /* 100% 0% */
BODY { background: url(banner.jpeg) top center } /* 50% 0% */
BODY { background: url(banner.jpeg) center } /* 50% 50% */
BODY { background: url(banner.jpeg) bottom } /* 50% 100% */
```

Si la imagen de fondo está fija con relación al lienzo (véase la propiedad '*background-attachment*' más arriba), la imagen se coloca con relación al lienzo y no con relación al elemento. P.ej.:

```
BODY {
  background-image: url(logo.png);
  background-attachment: fixed;
  background-position: 100% 100%;
}
```

En este ejemplo, la imagen se coloca en la esquina inferior derecha del lienzo.

A3.7. 'background'.

Valor: <background-color> || <background-image> || <background-repeat> || <background-attachment> || <background-position>

Inicial: no definido para propiedades abreviadas

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: permitidos para <background-position>

La propiedad '*background*' es una propiedad abreviada para establecer las propiedades de fondo individuales (es decir, '*background-color*', '*background-image*', '*background-repeat*', '*background-attachment*' y '*background-position*') en un mismo lugar de una hoja de estilo. Los valores posibles de las propiedades '*background*' son el conjunto de todos los posibles para cada una de las propiedades individuales.

```
BODY { background: red }
P { background: url(chess.png) gray 50% repeat fixed }
```

La propiedad '*background*' siempre establece todas las propiedades individuales. En la primera regla del ejemplo precedente, sólo se ha dado un valor para '*background-color*', y todas las demás propiedades individuales se han establecido a su valor inicial. En la segunda regla, se han especificado todas las propiedades individuales.

A4. Propiedades de texto

A4.1. 'word-spacing'.

Valor: normal | <longitud>

Inicial: normal

Se aplica a: todos los elementos

Se hereda: sí

Valores porcentuales: N/A

La unidad de longitud indica que se añade el espacio especificado al espacio por defecto entre palabras. Los valores pueden ser negativos, pero puede haber límites específicos de cada implementación. El AU es libre de elegir el algoritmo exacto de espaciado. El espaciado de palabras también puede verse influido por el tipo de justificación (que es un valor de la propiedad *'text-align'*).

```
H1 { word-spacing: 1em }
```

Aquí, el espaciado de palabras entre cada palabra contenida en elementos 'H1' se incrementa en '1em'. Los AAUU pueden interpretar cualquier valor de *'word-spacing'* como *'normal'*.

A4.2. 'letter-spacing'.

Valor: normal | <longitud>

Inicial: normal

Se aplica a: todos los elementos

Se hereda: sí

Valores porcentuales: N/A

Una unidad de longitud indica que se añade el espacio especificado al espacio por defecto entre caracteres. Los valores pueden ser negativos, pero puede haber límites específicos de cada implementación. El AU es libre de elegir el algoritmo exacto de espaciado. El espaciado de letras también puede verse influido por el tipo de justificación (que es un valor de la propiedad *'text-align'*).

```
BLOCKQUOTE { letter-spacing: 0.1em }
```

Aquí, el espaciado de letras entre cada carácter de los elementos *'BLOCKQUOTE'* se vería incrementado en *'0.1em'*. Con un valor de *'normal'*, el AU puede modificar el espacio entre letras para justificar el texto. Esto no sucederá si *'letter-spacing'* está establecido explícitamente a un valor de tipo <longitud>:

```
BLOCKQUOTE { letter-spacing: 0 }  
BLOCKQUOTE { letter-spacing: 0cm }
```

Cuando el espacio resultante entre dos letras no sea el mismo que el espacio por defecto, los AAUU no deberían usar ligaduras. Los AAUU pueden interpretar cualquier valor de *'letter-spacing'* como *'normal'*.

A4.3. 'text-decoration'.

Valor: none | [underline || overline || line-through || blink]

Inicial: none

Se aplica a: todos los elementos

Se hereda: no, pero véase la clarificación más abajo

Valores porcentuales: N/A

Esta propiedad describe las decoraciones que se añaden al texto de un elemento. Si el elemento no tiene texto (p.ej. el elemento 'IMG' en HTML) o es un elemento vacío (p.ej. ''), esta propiedad no tiene efecto. Un valor de 'blink' hace que el texto parpadee.

El color o colores requeridos por la decoración del texto deberían derivar del valor de la propiedad 'color'. Esta propiedad no es heredada, pero los elementos deberían concordar con sus padres. P.ej., si un elemento está subrayado, la línea debería abarcar todos los elementos hijos. El color del subrayado será el mismo incluso si los elementos descendientes tienen valores distintos de 'color'.

```
A:link, A:visited, A:active { text-decoration: underline }
```

En este ejemplo se subrayarían los textos de todos los vínculos (es decir, de todos los elementos 'A' con un atributo 'HREF'). Los AAUU deben reconocer la palabra clave 'blink', pero no es necesario que soporten el efecto de parpadeo.

A4.4. 'vertical-align'.

Valor: baseline | sub | super | top | text-top | middle | bottom | text-bottom | <porcentaje>

Inicial: baseline

Se aplica a: elementos en línea

Se hereda: no

Valores porcentuales: se refieren al 'line-height' del propio elemento

Esta propiedad afecta al posicionamiento vertical del elemento. Se utiliza un conjunto de palabras clave que son relativas al elemento padre:

'baseline' (línea de base)

alinea la línea de base del elemento (o la parte inferior si el elemento no tiene línea de base) con la línea de base del padre

'middle' (medio)

alinea el punto medio vertical del elemento (normalmente una imagen) con la línea de base más la mitad de la altura x del padre

'sub'

representa el elemento como un subíndice

'super'

representa el elemento como un superíndice

'text-top' (texto superior)

alinea la parte superior del elemento con la parte superior de la fuente del elemento padre

'text-bottom' (texto inferior)

alinea la parte inferior del elemento con la parte inferior de la fuente del elemento padre

Hay otro conjunto de palabras clave que son relativas a la línea formateada a la que pertenece el elemento:

'top' (superior)

alinea la parte superior del elemento con el elemento más alto de la línea.

'bottom' (inferior)

alinea la parte inferior del elemento con el elemento más bajo de la línea.

Cuando se usa la alineación superior (*'top'*) e inferior (*'bottom'*) pueden darse situaciones irresolubles si las dependencias entre elementos crean un bucle. Los valores porcentuales se refieren al valor de la propiedad *'line-height'* del propio elemento. Elevan la línea de base del elemento (o la línea inferior, si no tiene línea de base), en la cantidad especificada, por encima de la línea de base del elemento. Son posibles valores negativos. P.ej., un valor de "-100%" bajará el elemento de modo que la línea de base del elemento acabe donde debería haber estado la línea de base de la línea siguiente. Esto permite un control preciso sobre la posición vertical de los elementos que no tienen una línea de base (como p.ej. las imágenes que se usan en lugar de letras). Se espera que en una versión futura de CSS se permita un valor de tipo <longitud> para esta propiedad.

A4.5. 'text-transform'.

Valor: capitalize | uppercase | lowercase | none

Inicial: none

Se aplica a: todos los elementos

Se hereda: sí

Valores porcentuales: N/A

'capitalize'

convierte en mayúscula el primer carácter de cada palabra

'uppercase'

convierte en mayúsculas todas las letras del elemento

'lowercase'

convierte en minúsculas todas las letras del elemento

'none'

neutraliza el valor heredado

La transformación que se lleva finalmente a cabo depende del idioma en que esté el texto. Véanse en [RFC2070] diferentes formas de encontrar el idioma de un elemento.

```
H1 { text-transform: uppercase }
```

Este ejemplo convertiría los elementos 'H1' en texto en mayúsculas.

Los AAUU pueden no tener en cuenta *'text-transform'* (es decir, tratarlo como si siempre valiera *'none'*) para los caracteres que no pertenezcan al repertorio Latin-1 y para elementos en idiomas para los que la transformación es diferente de la especificada en las tablas de conversión de caja de Unicode [UNICODE].

A4.6. *'text-align'*.

Valor: left | right | center | justify

Inicial: depende del AU

Se aplica a: elementos en bloque

Se hereda: sí

Valores porcentuales: N/A

Esta propiedad describe cómo se alinea el texto dentro del elemento. El algoritmo de justificación utilizado finalmente depende del AU y del idioma en que esté el texto.

Ejemplo:

```
DIV.center { text-align: center }
```

Como *'text-align'* se hereda, todos los elementos en bloque dentro del elemento 'DIV' con 'CLASS=center' estarán centrados. Obsérvese que las alineaciones son relativas a la anchura del elemento, no del lienzo. Si *'justify'* no está soportado, al AU lo sustituirá por otro estilo. Normalmente será *'left'* para los lenguajes occidentales. Los AAUU pueden tratar *'justify'* como *'left'* o *'right'*, según que la dirección de escritura por defecto del elemento sea de izquierda a derecha o de derecha a izquierda respectivamente.

A4.7. *'text-indent'*.

Valor: <longitud> | <porcentaje>

Inicial: 0

Se aplica a: elementos en bloque

Se hereda: sí

Valores porcentuales: se refiere a la anchura del elemento padre

Esta propiedad especifica la sangría de la primera línea formateada. El valor de *'text-indent'* puede ser negativo, pero puede haber limitaciones específicas de cada implementación. No se inserta sangría en medio de un elemento que haya sido roto por otro (como el 'BR' en HTML).

Ejemplo:

```
P { text-indent: 3em }
```

A4.8. *'line-height'*.

Valor: normal | <número> | <longitud> | <porcentaje>

Inicial: normal

Se aplica a: todos los elementos

Se hereda: sí

Valores porcentuales: relativos al tamaño de la fuente del propio elemento

Esta propiedad establece la distancia entre las líneas de base de dos líneas adyacentes. Cuando se especifica un valor numérico, la altura de línea está dada por el tamaño de fuente del elemento considerado multiplicado por el valor numérico.

La diferencia entre eso y un valor porcentual está en el modo en que se hereda el valor: cuando se especifica un valor numérico, los elementos hijos heredarán el factor en sí, no el valor resultante (como es el caso de los porcentajes y otras unidades). No se permiten valores negativos. Las tres reglas del ejemplo siguiente dan el mismo resultado de altura de línea:

```
DIV { line-height: 1.2; font-size: 10pt } /* número */
DIV { line-height: 1.2em; font-size: 10pt } /* longitud */
DIV { line-height: 120%; font-size: 10pt } /* porcentaje */
```

Un valor *'normal'* establece la altura de línea a un valor razonable para la fuente del elemento. Se sugiere que los AAUU asignen al valor *'normal'* un número entre 1.0 y 1.2.

A5. Propiedades de cuadro

Las propiedades de cuadro establecen el tamaño, el perímetro y la posición de los cuadros que representan a los elementos. Las propiedades de margen establecen el margen de un elemento. La propiedad *'margin'* establece el margen para los cuatro lados, mientras que las demás propiedades de margen sólo establecen sus lados respectivos.

Las propiedades de relleno describen cuánto espacio se inserta entre el borde y el contenido (p.ej., un texto o una imagen). La propiedad *'padding'* establece el relleno para los cuatro lados, mientras que el resto de las propiedades de relleno establecen sólo sus lados respectivos.

Las propiedades de borde establecen los bordes de un elemento. Cada elemento tiene cuatro bordes, uno en cada lado, que están definidos por su anchura, su color y su estilo. Las propiedades *'width'* y *'height'* establecen el tamaño del cuadro, y las propiedades *'float'* y *'clear'* pueden alterar la posición de los elementos.

A5.1. *'margin-top'*.

Valor: <longitud> | <porcentaje> | auto

Inicial: 0

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: se refieren a la anchura del ascendente en bloque más cercano

Esta propiedad establece el margen superior de un elemento:

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

Se permiten valores negativos, pero puede haber limitaciones específicas de cada implementación.

A5.2. *'margin-right'*.

Valor: <longitud> | <porcentaje> | auto

Inicial: 0

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: se refieren a la anchura del ascendente en bloque más cercano

Esta propiedad establece el margen derecho de un elemento:

```
H1 { margin-right: 12.3% }
```

Se permiten valores negativos, pero puede haber limitaciones específicas de cada implementación.

A5.3. 'margin-bottom'.

Valor: <longitud> | <porcentaje> | auto

Inicial: 0

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: se refieren a la anchura del ascendente en bloque más cercano

Esta propiedad establece el margen inferior de un elemento:

```
H1 { margin-bottom: 3px }
```

Se permiten valores negativos, pero puede haber limitaciones específicas de cada implementación.

A5.4. 'margin-left'.

Valor: <longitud> | <porcentaje> | auto

Inicial: 0

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: se refieren a la anchura del ascendente en bloque más cercano

Esta propiedad establece el margen izquierdo de un elemento:

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

Se permiten valores negativos, pero puede haber limitaciones específicas de cada implementación.

A5.5. 'margin'.

Valor: [<longitud> | <porcentaje> | auto]{1,4}

Inicial: no definido para propiedades abreviadas

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: se refieren a la anchura del ascendente en bloque más cercano

La propiedad *'margin'* es una propiedad abreviada para establecer las propiedades *'margin-top'*, *'margin-right'*, *'margin-bottom'* y *'margin-left'* en un mismo lugar de una hoja de estilo. Si se especifican cuatro valores de longitud, se aplican a los márgenes superior, derecho, inferior e izquierdo respectivamente. Si sólo hay un valor, se aplica a todos los lados; si hay dos o tres valores, se toma para los valores que faltan los del lado opuesto.

```
BODY { margin: 2em } /* todos los márgenes puestos a 2em */
```

```
BODY { margin: 1em 2em } /* top y bottom = 1em, right y left = 2em */
```

```
BODY { margin: 1em 2em 3em } /* top=1em, right=2em, bottom=3em, left=2em */
```

La última regla de este ejemplo equivale al ejemplo siguiente:

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

Se permiten valores negativos, pero puede haber limitaciones específicas de cada implementación.

A5.6. 'padding-top'.

Valor: <longitud> | <porcentaje>

Inicial: 0

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: se refieren a la anchura del ascendiente en bloque más cercano

Esta propiedad establece el relleno superior de un elemento.

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

Los valores de relleno no pueden ser negativos.

A5.7. 'padding-right'.

Valor: <longitud> | <porcentaje>

Inicial: 0

Se aplica a: todos los elementos

Se hereda: no

Valores de porcentaje: se refieren a la anchura del ascendiente en bloque más cercano

Esta propiedad establece el relleno derecho de un elemento.

```
BLOCKQUOTE { padding-right: 10px }
```

Los valores de relleno no pueden ser negativos.

A5.8. 'padding-bottom'.

Valor: <longitud> | <porcentaje>

Inicial: 0

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: se refieren a la anchura del ascendiente en bloque más cercano

Esta propiedad establece el relleno inferior de un elemento.

```
BLOCKQUOTE { padding-bottom: 2em }
```

Los valores de relleno no pueden ser negativos.

A5.9. 'padding-left'.

Valor: <longitud> | <porcentaje>

Inicial: 0

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: se refieren a la anchura del ascendiente en bloque más cercano

Esta propiedad establece el relleno izquierdo de un elemento.

```
BLOCKQUOTE { padding-left: 20% }
```

Los valores de relleno no pueden ser negativos.

A5.10. 'padding'.

Valor: [<longitud> | <porcentaje>]{1,4}

Inicial: no definido para propiedades abreviadas

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: se refieren a la anchura del ascendiente en bloque más cercano

La propiedad *'padding'* es una propiedad abreviada para establecer las propiedades *'padding-top'*, *'padding-right'*, *'padding-bottom'* y *'padding-left'* en un mismo lugar de una hoja de estilo. Si se especifican cuatro valores, se aplican al relleno superior, derecho, inferior e izquierdo respectivamente. Si sólo hay un valor, se aplica a todos los lados; si hay dos o tres, se toman para los valores que faltan los del lado opuesto.

La superficie del área de relleno se establece con la propiedad *'background'*:

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

En este ejemplo se establece en *'1em'* el relleno vertical (*'padding-top'* y *'padding-bottom'*), y en *'2em'* el horizontal (*'padding-right'* y *'padding-left'*). La unidad *'em'* es relativa al tamaño de fuente del elemento: *'1em'* es igual al tamaño de la fuente en uso. Los valores de relleno no pueden ser negativos.

A5.11. 'border-top-width'.

Valor: thin | medium | thick | <longitud>

Inicial: 'medium'

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Esta propiedad establece la anchura del borde superior de un elemento. La anchura de los valores correspondientes a las palabras clave depende del AU, pero siempre se cumple que *'thin'* <= *'medium'* <= *'thick'*.

Las anchuras correspondientes a las palabras clave son constantes a lo largo de un mismo documento:

```
H1 { border: solid thick red }
P { border: solid thick blue }
```

En este ejemplo, los elementos 'H1' y 'P' tendrán la misma anchura de borde independientemente del tamaño de fuente. Para conseguir anchuras relativas, puede utilizarse la unidad 'em':

```
H1 { border: solid 0.5em }
```

Las anchuras de los bordes no pueden ser negativas.

A5.12. 'border-right-width'.

Valor: thin | medium | thick | <longitud>

Inicial: 'medium'

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Esta propiedad establece la anchura del borde derecho de un elemento. Por lo demás es análoga a la propiedad 'border-top-width'.

A5.13. 'border-bottom-width'.

Valor: thin | medium | thick | <longitud>

Inicial: 'medium'

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Esta propiedad establece la anchura del borde inferior de un elemento. Por lo demás es análoga a la propiedad 'border-top-width'.

A5.14. 'border-left-width'.

Valor: thin | medium | thick | <longitud>

Inicial: 'medium'

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Esta propiedad establece la anchura del borde izquierdo de un elemento. Por lo demás es análoga a la propiedad 'border-top-width'.

A5.15. 'border-width'.

Valor: [thin | medium | thick | <longitud>]{1,4}

Inicial: no definido para las propiedades abreviadas

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Esta propiedad es una propiedad abreviada para establecer las propiedades 'border-top-width', 'border-right-width', 'border-bottom-width' y 'border-left-width' en un mismo lugar de una hoja de estilo. Puede haber entre uno y cuatro valores, interpretándose de la siguiente manera:

- un valor: las cuatro anchuras de borde se establecen a ese valor
- dos valores: las anchuras de borde superior e inferior se establecen al primer valor, y las de los bordes derecho e izquierdo al del segundo
- tres valores: el primero es la anchura del borde superior, el segundo la del derecho e izquierdo, y el tercero la del inferior
- cuatro valores: superior, derecho, inferior e izquierdo respectivamente

En los ejemplos que siguen, los comentarios indican las anchuras resultantes de los bordes superior, derecho, inferior e izquierdo respectivamente:

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

Los bordes no pueden ser negativos.

A5.16. 'border-color'.

Valor: <color>{1,4}

Inicial: el valor de la propiedad 'color'

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

La propiedad '*border-color*' establece el color de los cuatro bordes. '*border-color*' puede tener de uno a cuatro valores, asignándose estos valores a los diferentes lados de manera análoga a la explicada anteriormente para la propiedad '*border-width*'.

Si no se especifica ningún valor de color, se utilizará en su lugar el valor de la propiedad '*color*':

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

En este ejemplo, el borde será una línea continua (*solid*) negra.

A5.17. 'border-style'.

Valor: none | dotted | dashed | solid | double | groove | ridge | inset | outset

Inicial: none

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

La propiedad '*border-style*' establece el estilo de los cuatro bordes. Puede tener entre uno y cuatro valores, estableciéndose los valores de los cuatro lados de manera análoga a la explicada anteriormente para la propiedad '*border-width*'.

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

En este ejemplo, los bordes horizontales serán continuos (*'solid'*) y los bordes verticales punteados (*'dotted'*). Al ser el valor inicial de los estilos de los bordes *'none'* (ninguno), ningún borde será visible a menos que se establezca el estilo del borde. El significado de los posibles estilos de borde es el siguiente:

none (ninguno)

no se dibuja el borde (independientemente del valor de la propiedad 'border-width')

dotted (punteado)

el borde es una línea punteada dibujada encima del fondo del elemento

dashed (a trazos)

el borde es una línea a trazos dibujada encima del fondo del elemento

solid (continuo)

el borde es una línea continua

double (doble)

el borde es una línea doble dibujada encima del fondo del elemento. La suma de cada una de las dos líneas más el espacio entre ellas es igual al valor `<anchura-de-borde>`.

groove (canal)

se dibuja un canal o acanaladura 3D con colores basados en el valor `<color>`.

ridge (cresta)

se dibuja una cresta o saliente 3D con colores basados en el valor `<color>`.

inset (bajorrelieve)

se dibuja un bajorrelieve 3D con colores basados en el valor `<color>`.

outset (altorrelieve)

se dibuja un altorrelieve 3D con colores basados en el valor `<color>`.

Los AAUU pueden interpretar cualquier valor *'dotted'*, *'dashed'*, *'double'*, *'groove'*, *'ridge'*, *'inset'* y *'outset'* como *'solid'*.

A5.18. 'border-top'.

Valor: `<anchura-de-borde-superior> || <estilo-de-borde> || <color>`

Inicial: no definido para propiedades abreviadas

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Esta es una propiedad abreviada para establecer la anchura, estilo y color del borde superior de un elemento.

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

La regla de este ejemplo establecerá la anchura, estilo y color del borde inferior del elemento H1. Si hay valores omitidos se igualarán a sus valores iniciales:

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

Al haberse omitido el valor del color en este último ejemplo, el color del borde será el mismo que el valor de *'color'* del propio elemento.

Obsérvese que mientras que la propiedad *'border-style'* acepta hasta cuatro valores, esta propiedad sólo acepta un valor de estilo.

A5.19. **'border-right'**.

Valor: <anchura-de-borde-derecho> || <estilo-de-borde> || <color>

Inicial: no definido para propiedades abreviadas

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Esta es una propiedad abreviada para establecer la anchura, estilo y color del borde derecho de un elemento. Por lo demás es equivalente a la propiedad *'border-top'*.

A5.20. **'border-bottom'**.

Valor: <anchura-de-borde-inferior> || <estilo-de-borde> || <color>

Inicial: no definido para propiedades abreviadas

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Esta es una propiedad abreviada para establecer la anchura, estilo y color del borde inferior de un elemento. Por lo demás es equivalente a la propiedad *'border-top'*.

A5.21. **'border-left'**.

Valor: <anchura-de-borde-izquierdo> || <estilo-de-borde> || <color>

Inicial: no definido para propiedades abreviadas

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Esta es una propiedad abreviada para establecer la anchura, estilo y color del borde izquierdo de un elemento. Por lo demás es equivalente a la propiedad *'border-top'*.

A5.22. **'border'**.

Valor: <anchura-de-borde> || <estilo-de-borde> || <color>

Inicial: no definido para propiedades abreviadas

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

La propiedad *'border'* es una propiedad abreviada para establecer la misma anchura, color y estilo en los cuatro bordes de un elemento. Por ejemplo, la primera regla del ejemplo siguiente es equivalente al conjunto de cuatro reglas que siguen a continuación:

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

A diferencia de las propiedades abreviadas *'margin'* y *'padding'*, la propiedad *'border'* no puede establecer valores distintos para cada borde. Para ello debe utilizarse una o más de las otras propiedades de borde. Como las propiedades tienen hasta cierto punto funcionalidades comunes, el orden en que deben especificarse las reglas es importante.

Consideremos este ejemplo:

```
BLOCKQUOTE {
  border-color: red;
  border-left: double;
  color: black;
}
```

En este ejemplo, el color del borde izquierdo será negro. Esto es debido a que *'border-left'* establece anchura, estilo y color. Y como el valor del color no está especificado en la propiedad *'border-left'*, se tomará de la propiedad *'color'*. El hecho de que la propiedad *'color'* se establezca después de la propiedad *'border-left'* no es relevante.

Obsérvese que si bien la propiedad *'border-width'* acepta hasta cuatro valores de longitud, esta propiedad sólo acepta uno.

A5.23. 'width'.

Valor: <longitud> | <porcentaje> | auto

Inicial: auto

Se aplica a: elementos en bloque y elementos reemplazados

Se hereda: no

Valores porcentuales: se refieren a la anchura del elemento padre

Esta propiedad puede aplicarse a elementos que contienen texto, pero es más útil para elementos reemplazados tales como imágenes. La anchura especificada debe ser respetada, escalando la imagen si es necesario. Cuando se escale, si la propiedad *'height'* es *'auto'*, deben preservarse las proporciones de la imagen.

Ejemplo:

```
IMG.icono { width: 100px }
```

Si las dos propiedades *'width'* y *'height'* de un elemento reemplazado valen *'auto'*, estas propiedades serán establecidas a las dimensiones intrínsecas del elemento. No se permiten valores negativos.

A5.24. 'height'.

Valor: <longitud> | auto

Inicial: auto

Se aplica a: elementos en bloque y elementos reemplazados

Se hereda: no

Valores porcentuales: N/A

Esta propiedad puede aplicarse elementos que contienen texto, pero es más útil para elementos reemplazados, tales como imágenes. La altura debe ser respetada, debiéndose escalar la imagen si es necesario. Al escalar, si el valor de la propiedad *'width'* es *auto*, deben mantenerse las proporciones de la imagen.

Ejemplo:

```
IMG.icono { height: 100px }
```

Si las dos propiedades *'width'* y *'height'* de un elemento reemplazado valen *'auto'*, estas propiedades serán establecidas a las dimensiones intrínsecas del elemento. Cuando se aplique a un elemento textual, la altura puede ser respetada, por ejemplo mediante una barra de scroll. No se permiten valores negativos. Los AAUU pueden no tener en cuenta la propiedad *'height'* (es decir, tratarla como si valiera *'auto'*) si el elemento no es un elemento reemplazado.

A5.25. *'float'*.

Valor: left | right | none

Inicial: none

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Con el valor *'none'*, el elemento será mostrado donde aparezca en el texto. Con un valor *'left'* el elemento será movido a la izquierda y el texto continuará por el lado derecho del elemento. Con un valor *'right'* el elemento será movido a la derecha y el texto cambiará de línea por el lado izquierdo del elemento. Con un valor de *'left'* o *'right'* se considera al elemento como un elemento en bloque (es decir, no se tiene en cuenta la propiedad *'display'*).

```
IMG.icono {  
  float: left;  
  margin-left: 0;  
}
```

Este ejemplo colocará todos los elementos *'IMG'* con *'CLASS=icono'* junto al lado izquierdo del elemento padre. Esta propiedad se usa normalmente con imágenes en línea, pero también se aplica a elementos de texto.

A5.26. *'clear'*.

Valor: none | left | right | both

Inicial: none

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Esta propiedad especifica si un elemento permite que haya elementos flotantes a sus lados. Más concretamente, el valor de esta propiedad enumera los lados en los que no acepta elementos flotantes. Con *'clear'* igual a *'left'* el elemento será movido hasta quedar por debajo del elemento flotante a la izquierda. Con *'clear'* igual a *'none'*, se permiten elementos flotantes a ambos lados.

Ejemplo:

```
H1 { clear: left }
```

A6. Propiedades de clasificación

Estas propiedades, más que especificar parámetros visuales específicos, lo que hacen es clasificar elementos en categorías. Las propiedades de estilo de lista describen cómo dar formato a los objetos de una lista (es decir, a los elementos con un valor de *'display'* igual a *'list-item'*). Las propiedades de estilo de lista pueden establecerse para cualquier elemento, y normalmente se heredarán de padres a hijos. Sin embargo, sólo tendrán efecto en aquellos elementos que tengan un valor de *'display'* igual a *'list-item'*. En HTML éste normalmente el caso de los elementos 'LI'.

A6.1. 'display'.

Valor: block | inline | list-item | none

Inicial: block

Se aplica a: todos los elementos

Se hereda: no

Valores porcentuales: N/A

Esta propiedad describe si un elemento se muestra en el lienzo (ya sea una página impresa, una pantalla de computadora, etc.) y cómo se muestra. Un elemento con un valor de *'display'* igual a *'block'* abre un nuevo cuadro. El cuadro se posiciona relativamente a los cuadros adyacentes de acuerdo al modelo de formato de CSS. Normalmente, elementos tales como 'H1' y 'P' son del tipo *'block'*. El valor *'list-item'* es similar a *'block'*, excepto que se añade un marcador de objeto de lista. En HTML, 'LI' tendrá normalmente este valor.

Un elemento con un valor de *'display'* igual a *'inline'* da como resultado un nuevo cuadro de línea en la misma línea que el contenido previo. El cuadro se dimensiona de acuerdo con el tamaño del contenido una vez formateado. Si el contenido es textual, puede abarcar varias líneas, y habrá un cuadro en cada línea. Las propiedades de margen, borde y relleno se aplican a los elementos *'inline'*, pero no tendrán ningún efecto en los saltos de línea.

Un valor de *'none'* desactiva la presentación del documento, incluyendo los elementos hijos y el cuadro circundante.

```
P { display: block }
EM { display: inline }
LI { display: list-item }
IMG { display: none }
```

La última regla desactiva la presentación de las imágenes.

El valor inicial de *'display'* es *'block'*, pero un AU tendrá por lo general valores por defecto para cada elemento HTML de acuerdo con la representación de elementos sugerida por la especificación HTML [HTML40]. Los AAUU pueden no tener en cuenta el valor de *'display'* y usar solamente los valores por defecto del AU.

A6.2. 'white-space'.

Valor: normal | pre | nowrap

Inicial: normal

Se aplica a: elementos en bloque

Se hereda: sí

Valores porcentuales: N/A

Esta propiedad declara cómo se manipulan los espacios existentes en el interior del elemento: con el modo *'normal'* (según el cual los espacios en blanco se colapsan), *'pre'* (que se comporta como el elemento 'PRE' de HTML) o *'nowrap'* (según el cual los cambios de línea sólo ocurren por la presencia de elementos BR):

```
PRE { white-space: pre }
P   { white-space: normal }
```

El valor inicial de *'white-space'* es *'normal'*, pero un AU tendrá normalmente valores por defecto para cada elemento HTML de acuerdo con la representación de elementos sugerida por la especificación HTML [HTML40]. Los AAUU pueden no tener en cuenta la propiedad *'white-space'* en las hojas de estilo del autor y del lector, y usar en su lugar los valores por defecto del AU.

A6.3. 'list-style-type'.

Valor: disc | circle | square | decimal | lower-roman | upper-roman | lower-alpha | upper-alpha | none

Inicial: disc

Se aplica a: elementos con valor de 'display' igual a 'list-item'

Se hereda: sí

Valores porcentuales: N/A

Esta propiedad se utiliza para determinar la apariencia del marcador de objetos de lista si *'list-style-image'* es *'none'* o si la imagen a la que apunta su URL no puede ser mostrada.

```
OL { list-style-type: decimal }      /* 1 2 3 4 5 etc. */
OL { list-style-type: lower-alpha } /* a b c d e etc. */
OL { list-style-type: lower-roman } /* i ii iii iv v etc. */
```

A6.4. 'list-style-image'.

Valor: <url> | none

Inicial: none

Se aplica a: elementos con valor de 'display' igual a 'list-item'

Se hereda: sí

Valores porcentuales: N/A

Esta propiedad establece la imagen que se usará como marcador de objetos de lista. Si la imagen está disponible reemplazará al marcador establecido con la propiedad *'list-style-type'*.

```
UL { list-style-image: url(http://png.com/ellipse.png) }
```

A6.5. 'list-style-position'.

Valor: inside | outside

Inicial: outside

Se aplica a: elementos con valor de 'display' igual a 'list-item'

Se hereda: sí

Valores porcentuales: N/A

El valor de *'list-style-position'* determina cómo se dibuja el marcador de objetos de lista con relación al contenido.

A6.6. 'list-style'.

Valor: [disc | circle | square | decimal | lower-roman | upper-roman | lower-alpha | upper-alpha | none] || [inside | outside] || [<url> | none]

Inicial: no definida para propiedades abreviadas

Se aplica a: elementos con valor de 'display' igual a 'list-item'

Se hereda: sí

Valores porcentuales: N/A

La propiedad '*list-style*' es una notación abreviada para establecer las tres propiedades '*list-style-type*', '*list-style-image*' y '*list-style-position*' en un mismo lugar de una hoja de estilo.

```
UL { list-style: upper-roman inside }
UL UL { list-style: circle outside }
LI.square { list-style: square }
```

Establecer '*list-style*' directamente en un elemento 'LI' puede dar lugar a resultados inesperados. Considérese el siguiente ejemplo:

```
<STYLE TYPE="text/css">
  OL.alpha LI { list-style: lower-alpha }
  UL LI      { list-style: disc }
</STYLE>
<BODY>
  <OL CLASS=alpha>
    <LI>nivel 1
    <UL>
      <LI>nivel 2
    </UL>
  </OL>
</BODY>
```

Al ser mayor la especificidad (definida en el orden de cascada) de la primera regla de la hoja de estilo del ejemplo, prevalecerá sobre la segunda regla en todos los elementos 'LI' y sólo se utilizarán estilos de lista '*lower-alpha*'. Por ello se recomienda establecer la propiedad '*list-style*' únicamente en los elementos de tipo lista:

```
OL.alpha { list-style: lower-alpha }
UL       { list-style: disc }
```

En este ejemplo, la herencia transmitirá los valores de '*list-style*' de los elementos 'OL' y 'UL' a los elementos 'LI'. Un valor de tipo URL puede combinarse con cualquier otro valor:

```
UL { list-style: url(http://png.com/ellipse.png) disc }
```

En este ejemplo, se usará '*disc*' si la imagen no está disponible.

A7. Referencias

[HTML40]

"HTML 4.0 Specification", D. Raggett, A. Le Hors, I. Jacobs, Julio 8 de 1997. Disponible en <http://www.w3.org/TR/REC-html40/>. La recomendación define tres definiciones del tipo de documento: Strict, Transitional y Frameset, todas al alcance en la recomendación.

[RFC2070]

"Internationalization of the HyperText Markup Language", F. Yergeau, G. Nicol, G. Adams, y M. Dürst, Enero 1997. Disponible en <http://www.ietf.org/rfc/rfc2070.txt>.

[UNICODE]

"The Unicode Standard: Version 2.0", The Unicode Consortium, Addison-Wesley Developers Press, 1996. Disponible en <http://www.unicode.org/>.

[W3CSTYLE]

Recurso del W3C sobre hojas de estilo en la web. Disponible en <http://www.w3.org/pub/WWW/Style>.

Apéndice B

Unidades

Ferran Perdrix Sapiña



Grupo GRIHO (UdL)
C/Jaume II n° 69
ferran@griho.net



Índice

B1. Unidades de longitud	B3
B2. Unidades porcentuales	B4
B3. Unidades de color	B4
B4. URL	B5
B5. Referencias	B5

B1. Unidades de longitud

El formato de una unidad de longitud es un carácter opcional de signo ('+' o '-', siendo '+' el valor por defecto) seguido inmediatamente de un número (con o sin punto decimal) seguido inmediatamente de un identificador de unidad (una abreviación de dos letras). Después del número '0' el identificador de unidad es opcional.

Algunas propiedades permiten unidades de longitud negativas, pero esto puede complicar el modelo de formato y pueden existir limitaciones específicas de cada implementación. Si en algún caso no se puede dar soporte a un valor negativo, debería truncarse al valor más cercano que pueda aceptarse.

Hay dos tipos de unidades de longitud: relativas y absolutas. Las unidades relativas especifican una longitud relativa a otra propiedad de longitud. Las hojas de estilo que usen unidades relativas serán más fácilmente escalables de un medio a otro (p.ej. de una pantalla de ordenador a una impresora láser). Las unidades porcentuales (descritas más adelante) y los valores de palabras clave (p.ej. 'x-large') ofrecen ventajas similares.

Estas son las unidades relativas soportadas:

```
H1 { margin: 0.5em }      /* ems, la altura de la fuente del elemento */
H1 { margin: 1ex }      /* altura-x, aprox. la altura de la letra 'x' */
P { font-size: 12px }   /* píxeles, relativos al lienzo */
```

Las unidades relativas '*em*' y '*ex*' son relativas al tamaño de fuente del propio elemento. La única excepción a esta regla en CSS1 es la propiedad '*font-size*', donde tanto '*em*' como '*ex*' se refieren al tamaño de fuente del elemento padre.

Las unidades en píxeles, como las de la última regla, son relativas a la resolución del lienzo, o lo que es lo mismo en la mayoría de los casos, a la resolución de la pantalla de la computadora. Si la densidad de píxeles del dispositivo de salida es distinta a la de una pantalla normal de computadora, el AU debería reescalar los valores dados en píxeles. El *píxel de referencia* que se sugiere es el ángulo visual de un píxel en un dispositivo con una densidad de píxeles de 90 puntos por pulgada y a una distancia del lector igual a la longitud del brazo. Tomando una longitud nominal de brazo de 28 pulgadas, el ángulo visual es de unos 0,0227 grados. Los elementos hijos heredan el valor calculado, no el valor relativo.

```
BODY {
  font-size: 12pt;
  text-indent: 3em; /* es decir, 36pt */
}
H1 { font-size: 15pt }
```

En este ejemplo, el valor '*text-indent*' de los elementos 'H1' será de 36pt, no de 45pt.

Las unidades absolutas de longitud sólo son útiles cuando se conocen las propiedades físicas del medio de salida. Estas son las unidades absolutas soportadas:

```
H1 { margin: 0.5in }      /* pulgadas (inches), 1in = 2.54cm */
H2 { line-height: 3cm }  /* centímetros */
H3 { word-spacing: 4mm } /* milímetros */
H4 { font-size: 12pt }   /* puntos, 1pt = 1/72 in */
H4 { font-size: 1pc }    /* picas, 1pc = 12pt */
```

En aquellos casos en que la longitud especificada no pueda ser soportada, los AAUU deberían intentar aproximar. En todas las propiedades CSS1, los cálculos y herencias ulteriores deberían basarse en el valor aproximado.

B2. Unidades porcentuales

El formato de una unidad porcentual es un carácter de signo opcional ('+' o '-', siendo '+' el valor por defecto), inmediatamente seguido de un número (con o sin punto decimal), inmediatamente seguido de '%'.

Los valores porcentuales son siempre relativos a otro valor, por ejemplo una unidad de longitud. Todas las propiedades que permiten unidades porcentuales definen a qué valor se refiere el porcentaje. Normalmente será el tamaño de fuente del propio elemento:

```
P { line-height: 120% } /* 120% del 'font-size' del elemento */
```

En todas las propiedades CSS1 heredadas, cuando el valor se especifique como un porcentaje, los elementos hijos heredarán el valor resultante, no el valor porcentual.

B3. Unidades de color

Un color es o bien una palabra clave o bien una especificación numérica RGB.

La lista sugerida de nombres de palabras clave de color es: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, y yellow. Estos 16 colores están tomados de la paleta VGA de Windows, y sus valores RGB no se definen en esta especificación.

```
BODY { color: black; background: white }
H1 { color: maroon }
H2 { color: olive }
```

Para especificaciones numéricas de color se utiliza el modelo de colores RGB. Todos estos ejemplos especifican el mismo color:

```
EM { color: #f00 } /* #rgb */
EM { color: #ff0000 } /* #rrggbb */
EM { color: rgb(255,0,0) } /* rango de enteros 0 - 255 */
EM { color: rgb(100%, 0%, 0%) } /* rango de reales (float) 0.0% - 100.0% */
```

El formato de un valor RGB en notación hexadecimal es un '#' inmediatamente seguido de o bien 3 o bien 6 caracteres hexadecimales. La notación de 3 dígitos RGB (#rgb) se convierte a notación de 6 dígitos replicando dígitos, no añadiendo ceros. Por ejemplo, #fb0 se convierte en #ffbb00. Con esto se asegura que el blanco (#ffffff) puede especificarse con la notación abreviada (#fff) y elimina cualquier dependencia de la profundidad de color del dispositivo de salida.

El formato de un valor RGB en la notación funcional es 'rgb(', seguido de una lista de tres valores numéricos separados por coma (ya sean tres valores numéricos entre 0 y 255, o tres valores porcentuales entre 0,0% y 100,0%), seguida de ')'. Se permite espacio en blanco alrededor de los valores numéricos.

Los valores que se encuentran fuera de los rangos numéricos deberían truncarse. Las tres reglas siguientes son por lo tanto equivalentes:

```
EM { color: rgb(255,0,0) } /* rango de enteros 0 - 255 */
EM { color: rgb(300,0,0) } /* truncado a 255 */
EM { color: rgb(110%, 0%, 0%) } /* truncado a 100% */
```

Los colores RGB se especifican en el espacio de colores sRGB [SRGB]. Cada AU puede tener distinta fiabilidad a la hora de representar estos colores, pero el uso de sRGB proporciona una definición inequívoca y medible objetivamente de lo que debería ser el color, y que puede ser contrastada con estándares internacionales [COLOR].

Los AAUU pueden limitar sus esfuerzos de representación de colores a la aplicación de una corrección gamma sobre ellos. sRGB especifica un valor gamma de 2,2 bajo unas condiciones de visión específicas. Los AAUU ajustan los colores dados en CSS de tal modo que, combinados con el valor gamma "natural" del dispositivo de salida, se obtenga un valor gamma efectivo de 2,2. El apéndice D da más detalles a este respecto. Obsérvese que esto sólo afecta a los colores especificados en CSS; p.ej. las imágenes se supone que llevan su propia información de color.

B4. URL

Un Localizador Uniforme de Recursos (*Uniform Resource Locator*, URL) se identifica con una notación funcional:

```
BODY { background: url(http://www.bg.com/rosado.gif) }
```

El formato de un valor URL es 'url(', seguido de espacio blanco opcional, seguido de un carácter opcional de comilla simple (') o doble ("), seguido del URL en sí (como se define en [RFC1738]), seguido de un carácter opcional de comilla simple (') o doble ("), seguido de espacio en blanco opcional, seguido de ')'. Las comillas que no sean parte del URL en sí deben cerrarse. Los paréntesis, comas, espacios en blanco, las comillas simples (') y las comillas dobles (") que aparezcan en una URL deben ser transformados en secuencias de escape mediante una barra invertida: '\(', '\)', '\,', '\,'.

Los URLs parciales se interpretan como relativos al fuente de la hoja de estilo, no relativos al documento:

```
BODY { background: url(amarillo) }
```

B5. Referencias

[COLOR]

"Colorimetry, Second Edition", Publicación CIE 15.2-1986, ISBN 3-900-734-00-3.

[RFC1738]

"Uniform Resource Locators", T. Berners-Lee, L. Masinter y M. McCahill, Diciembre 1994. Disponible en <http://www.w3.org/Addressing/rfc1738.txt>.

[SRGB]

"Proposal for a Standard Color Space for the Internet - sRGB", M. Anderson, R. Motta, S. Chandrasekar, M. Stokes. Disponible en <http://www.w3.org/Graphics/Color/sRGB.html>.

[W3CSTYLE]

Recurso del W3C sobre hojas de estilo en la web. Disponible en <http://www.w3.org/pub/WWW/Style>.

Apéndice C

Gramáticas

Ferran Perdrix Sapiña



Grupo GRIHO (UdL)
C/Jaume II nº 69
ferran@griho.net



Índice

C1. Notación para los valores de las propiedades	C3
C2. Gramática de CSS1	C3
C3. Analizador léxico CSS1	C5
C4. Gramática de CSS2	C7
C5. Analizador léxico CSS2	C8
C6. Comparación de los comandos en CSS2 y CSS1	C10
C7. Referencias	C11

C1. Notación para los valores de las propiedades

La gramática mínima de CSS (esto es, de cualquier versión de CSS) que necesita soportar cualquier implementación está definida a continuación:

Los valores permitidos para cada propiedad se enumeran con una sintaxis similar a la siguiente:

```
Valor: N | NW | NE
Valor: [ <longitud> | thick | thin ]{1,4}
Valor: [<nombre-de-familia> ,]* <nombre-de-familia>
Valor: <url> ? <color> [ / <color> ]?
Valor: <url> || <color>
```

Los valores entre "<" y ">" dan un tipo de valor. Los tipos más comunes son <longitud>, <porcentaje>, <url>, <número> y <color>. Los tipos más especializados (p.ej., <familia-de-fuentes> y <estilo-de-borde>) se describen bajo la propiedad correspondiente. Otras palabras son palabras clave que deben aparecer literalmente, sin comillas. La barra inclinada (/) y la coma (,) también deben aparecer literalmente.

Cuando aparezcan varias cosas yuxtapuestas, deben incluirse todas ellas en el orden especificado. Una barra (|) separa alternativas: debe especificarse una de ellas. Una barra doble (A || B) significa que debe especificarse A, B o ambas, en cualquier orden. Los corchetes ([]) se usan para agrupar. La yuxtaposición es más fuerte que la doble barra, y la doble barra es más fuerte que la barra. Así, "a b | c || d e" es equivalente a "[a b] || [c || [d e]]".

Cada tipo, palabra clave, o grupo entre corchetes, puede ir seguido por uno de los siguientes modificadores:

- Un asterisco (*) indica que el tipo, palabra o grupo precedente se repite cero o más veces.
- Un signo más (+) indica que el tipo, palabra o grupo precedente se repite una o más veces.
- Un signo de interrogación (?) indica que el tipo, palabra o grupo precedente es opcional.
- Un par de números entre llaves ({A,B}) indica que el tipo, palabra o grupo precedente, se repite al menos A veces y como mucho B veces.

C2. Gramática de CSS1

La gramática que figura a continuación define un lenguaje mucho más pequeño, un lenguaje que define la sintaxis de CSS1. En cierto sentido sigue siendo, sin embargo, un superconjunto de CSS1: hay restricciones semánticas adicionales que no están expresadas en esta sintaxis. Un AU conforme debe adherirse, además, a las reglas de interpretación compatibles con versiones futuras, a la notación de propiedades y valores y a la notación de unidades. Además HTML impone restricciones, p.ej., en los valores posibles para el atributo 'CLASS'.

La gramática que sigue es LL(1) (obsérvese no obstante que los AAUU no deberían usarla directamente, ya que no expresa convenciones de interpretación, sino únicamente la sintaxis de CSS1). El formato de las producciones está optimizado para su utilización por personas, y se utilizan ciertas notaciones abreviadas no contempladas en *yacc* [YACC]:

```
* : 0 ó más
+ : 1 ó más
? : 0 ó 1
| : separa alternativas
[] : agrupa
```

Las producciones son:

```
stylesheet
: [CDO|CDC]* [ import [CDO|CDC]* ]* [ ruleset [CDO|CDC]* ]*
;
import
: IMPORT_SYM [STRING|URL] ';'          /* P.ej., @import url(fun.css); */
;
unary_operator
: '-' | '+'
;
operator
: '/' | ',' | /* empty */
;
property
: IDENT
;
ruleset
: selector [ ',' selector ]*
  '{' declaration [ ';' declaration ]* '}'
;
selector
: simple_selector+ [ pseudo_element | solitary_pseudo_element ]?
| solitary_pseudo_element
;
/* Un "id" es un ID que está unido a un tipo de elemento
** por su izquierda, como en: P#p007
** Un "solitary_id" es un ID que no está unido así,
** como en: #p007
** Análogamente para clases y pseudo-clases.
*/
simple_selector
: element_name id? class? pseudo_class?          /* p.ej.: H1.subject */
| solitary_id class? pseudo_class?              /* p.ej.: #xyz33 */
| solitary_class pseudo_class?                  /* p.ej.: .author */
| solitary_pseudo_class                          /* p.ej.: :link */
;
element_name
: IDENT
;
pseudo_class
: LINK_PSCLASS_AFTER_IDENT                       /* como en: A:link */
| VISITED_PSCLASS_AFTER_IDENT
| ACTIVE_PSCLASS_AFTER_IDENT
;
solitary_pseudo_class
: LINK_PSCLASS                                   /* como en: :link */
| VISITED_PSCLASS
| ACTIVE_PSCLASS
;
```

```

class                               /* como en:  P.note */
  : CLASS_AFTER_IDENT
  ;
solitary_class                       /* como en:  .note */
  : CLASS
  ;
pseudo_element                       /* como en:  P:first-line */
  : FIRST_LETTER_AFTER_IDENT
  | FIRST_LINE_AFTER_IDENT
  ;
solitary_pseudo_element              /* como en:  :first-line */
  : FIRST_LETTER
  | FIRST_LINE
  ;
    /* Tanto en el id como en el solitary id existe la restricción de que
    ** la parte que sigue al "#" debe ser un valor ID HTML válido;
    ** p.ej., "#x77" está bien, pero "#77" no.
    */

id
  : HASH_AFTER_IDENT
  ;
solitary_id
  : HASH
  ;
declaration
  : property ':' expr prio?
  | /* empty */                               /* Evita errores de sintaxis... */
  ;
prio
  : IMPORTANT_SYM                             /* !important */
  ;
expr
  : term [ operator term ]*
  ;
term
  : unary_operator?
    [ NUMBER | STRING | PERCENTAGE | LENGTH | EMS | EXS
    | IDENT | hexcolor | URL | RGB ]
  ;
    /* En el color existe la restricción de que debe
    ** tener 3 ó 6 dígitos hexadecimales (es decir, [0-9a-fA-F])
    ** después del "#"; p.ej., "#000" está bien, pero "#abcd" no.
    */

hexcolor
  : HASH | HASH_AFTER_IDENT
  ;

```

C3. Analizador léxico CSS1

A continuación figura el analizador léxico (*tokenizer*), escrito en notación *flex* [FLEX]. Obsérvese que asume una implementación de *flex* de 8 bits. El analizador léxico no distingue entre mayúsculas y minúsculas (opción `-i` de la línea de comandos de *flex*).

```

unicode      \\[0-9a-f]{1,4}
latin1       [ı-ÿ]
escape       {unicode}|\\[ -~ı-ÿ]
stringchar   {escape}|{latin1}|[ !#$%&(-~]
nmstrt       [a-z]|{latin1}|{escape}

```

```

nmchar      [-a-z0-9]|{\latin1}|{\escape}
ident       {nmstrt}{nmchar}*
name        {nmchar}+
d           [0-9]
notnm       [^-a-z0-9\\]|{\latin1}
w           [ \t\n]*
num         {d}+|{d}*\.{d}+
string      \"({stringchar}|\\'|\\\"|\\'({stringchar}|\\\"))*\'

%x COMMENT
%s AFTER_IDENT

%%
"/*"          {BEGIN(COMMENT);}
<COMMENT>"/" {BEGIN(0);}
<COMMENT>\n   {/* ignore */}
<COMMENT>.    {/* ignore */}
@import       {BEGIN(0); return IMPORT_SYM;}
"!"{w}important {BEGIN(0); return IMPORTANT_SYM;}
{ident}       {BEGIN(AFTER_IDENT); return IDENT;}
{string}      {BEGIN(0); return STRING;}

{num}         {BEGIN(0); return NUMBER;}
{num}"%"      {BEGIN(0); return PERCENTAGE;}
{num}pt/{notnm} {BEGIN(0); return LENGTH;}
{num}mm/{notnm} {BEGIN(0); return LENGTH;}
{num}cm/{notnm} {BEGIN(0); return LENGTH;}
{num}pc/{notnm} {BEGIN(0); return LENGTH;}
{num}in/{notnm} {BEGIN(0); return LENGTH;}
{num}px/{notnm} {BEGIN(0); return LENGTH;}
{num}em/{notnm} {BEGIN(0); return EMS;}
{num}ex/{notnm} {BEGIN(0); return EXS;}

<AFTER_IDENT>":"link          {return LINK_PSCLASS_AFTER_IDENT;}
<AFTER_IDENT>":"visited {return VISITED_PSCLASS_AFTER_IDENT;}
<AFTER_IDENT>":"active {return ACTIVE_PSCLASS_AFTER_IDENT;}
<AFTER_IDENT>":"first-line  {return FIRST_LINE_AFTER_IDENT;}
<AFTER_IDENT>":"first-letter {return FIRST_LETTER_AFTER_IDENT;}
<AFTER_IDENT>:"#"{name}     {return HASH_AFTER_IDENT;}
<AFTER_IDENT>"."{name}     {return CLASS_AFTER_IDENT;}

":"link          {BEGIN(AFTER_IDENT); return LINK_PSCLASS;}
":"visited       {BEGIN(AFTER_IDENT); return VISITED_PSCLASS;}
":"active        {BEGIN(AFTER_IDENT); return ACTIVE_PSCLASS;}
":"first-line    {BEGIN(AFTER_IDENT); return FIRST_LINE;}
":"first-letter  {BEGIN(AFTER_IDENT); return FIRST_LETTER;}
"#"{name}       {BEGIN(AFTER_IDENT); return HASH;}
"."{name}        {BEGIN(AFTER_IDENT); return CLASS;}

url\({w}{string}{w}\)      |
url\({w}([^\n\'\\"])|\\\\ |\\\\\'|\\\\\"|\\\\\\\\))+{w}\) | {BEGIN(0); return
URL;} |
rgb\({w}{num}%?{w}\,{w}{num}%?{w}\,{w}{num}%?{w}\) | {BEGIN(0); return
RGB;} |

[-/+{}; , # :] {BEGIN(0); return *yytext;}
[ \t]+         {BEGIN(0); /* ignore whitespace */}
\n            {BEGIN(0); /* ignore whitespace */}
\<!\-\\-     {BEGIN(0); return CDO;}
\\-\\->       {BEGIN(0); return CDC;}
.              {fprintf(stderr, "%d: Illegal character
(%d)\n", lineno, *yytext);}

```

C4. Gramática de CSS2

La gramática siguiente define la sintaxis de CSS2. Sin embargo, es en cierto sentido un superconjunto de CSS2 puesto que impone restricciones semánticas adicionales no expresadas en esta gramática. Una aplicación del usuario conformada debe también adherir a las reglas de análisis con compatibilidad futura, la notación de propiedades y valores y la unidad de notación. Además, el lenguaje del documento puede imponer restricciones, por ej., HTML impone restricciones a los posibles valores del atributo "class".

La gramática es LL(1) (pero observe que la mayoría de la AU no debe utilizarla directamente, debido a que no expresa las convenciones de análisis, sólo la sintaxis de CSS2). El formato de las producciones está optimizado para el uso humano y se usa alguna notación abreviadas más allá de *yacc* (Ver [YACC]):

```

*: 0 o más
+: 1 o más
?: 0 o 1
|: separa alternativas
[ ]: agrupa

```

Las producciones son:

```

stylesheet
: [ CHARSET_SYM S* STRING S* ';' ]?
  [S|CDO|CDC]* [ import [S|CDO|CDC]* ]*
  [ [ ruleset | media | page | font_face ] [S|CDO|CDC]* ]*
;
import
: IMPORT_SYM S*
  [STRING|URI] S* [ medium [ ',' S* medium]* ]? ';' S*
;
media
: MEDIA_SYM S* medium [ ',' S* medium ]* '{' S* ruleset* '}' S*
;
medium
: IDENT S*
;
page
: PAGE_SYM S* IDENT? pseudo_page? S*
  '{' S* declaration [ ';' S* declaration ]* '}' S*
;
pseudo_page
: ':' IDENT
;
font_face
: FONT_FACE_SYM S*
  '{' S* declaration [ ';' S* declaration ]* '}' S*
;
operator
: '/' S* | ',' S* | /* vacío */
;
combinator
: '+' S* | '>' S* | /* vacío */
;
unary_operator
: '-' | '+'
;
property
: IDENT S*
;

```

```

ruleset
  : selector [ ',' S* selector ]*
    '{' S* declaration [ ';' S* declaration ]* '}' S*
  ;
selector
  : simple_selector [ combinator simple_selector ]*
  ;
simple_selector
  : element_name? [ HASH | class | attrib | pseudo ]* S*
  ;
class
  : '.' IDENT
  ;
element_name
  : IDENT | '*'
  ;
attrib
  : '[' S* IDENT S* [ [ '=' | INCLUDES | DASHMATCH ] S*
    [ IDENT | STRING ] S* ]? ']'
  ;
pseudo
  : ':' [ IDENT | FUNCTION S* IDENT S* ')' ]
  ;
declaration
  : property ':' S* expr prio?
  | /* vacío */
  ;
prio
  : IMPORTANT_SYM S*
  ;
expr
  : term [ operator term ]*
  ;
term
  : unary_operator?
    [ NUMBER S* | PERCENTAGE S* | LENGTH S* | EMS S* | EXS S* | ANGLE S* |
      TIME S* | FREQ S* | function ]
  | STRING S* | IDENT S* | URI S* | RGB S* | UNICODERANGE S* | hexcolor
  ;
function
  : FUNCTION S* expr ')' S*
  ;
/*
 * Hay una restricción en el color que debe tener
 * 3 o 6 dígitos hexadecimales (es decir, [0-9a-fA-F])
 * después de "#"; ej., "#000" está bien, pero "#abcd" no.
 */
hexcolor
  : HASH S*
  ;

```

C5. Analizador léxico CSS2

A continuación se presenta la definición de los comandos (tokenizer) escrito en notación *flex* (ver [FLEX]). Es sensible a la diferencia entre mayúsculas y minúsculas.

Las dos apariciones de "\377" representan el número de carácter más alto que la versión actual de *flex* puede manejar (decimal 255). Debe leerse como "\4177777" (decimal 1114111), que es el punto de código más alto posible en Unicode/ISO-10646.

```

%option case-insensitive

h          [0-9a-f]
nonascii  [\200-\377]
unicode   \\{h}{1,6}[ \t\r\n\f]?
escape    {unicode}|\|[ -~\200-\377]
nmstart   [a-z]|\{nonascii}|\{escape}
nmchar    [a-zA-Z0-9]|\{nonascii}|\{escape}
string1   \"([\t !#$%&(-~)|\\{nl}|\'|\\{nonascii}|\{escape})*\"
string2   \'([\t !#$%&(-~)|\\{nl}|\\"|\{nonascii}|\{escape})*\'

ident     {nmstart}{nmchar}*
name      {nmchar}+
num       [0-9]+|[0-9]*\".[0-9]+
string    {string1}|\{string2}
url       ([!#$%&*~]|\\{nonascii}|\{escape})*
w        [ \t\r\n\f]*
nl       \n|\r\n|\r|\f
range     \?{1,6}|\{h}(\?{0,5}|\{h}(\?{0,4}|\{h}(\?{0,3}|\{h}(\?{0,2}|\{h}(\?|\{h})))
))

%%

[ \t\r\n\f]+          {return S;}

\/\*[\^]**\*+([\^/][\^]*\*+)*\/ /* ignore comments */

"<!--"                {return CDO;}
"-->"                 {return CDC;}
"~="                  {return INCLUDES;}
"|="                  {return DASHMATCH;}

{string}              {return STRING;}

{ident}               {return IDENT;}

"#" {name}            {return HASH;}

"@import"             {return IMPORT_SYM;}
"@page"               {return PAGE_SYM;}
"@media"              {return MEDIA_SYM;}
"@font-face"          {return FONT_FACE_SYM;}
"@charset"            {return CHARSET_SYM;}
"@" {ident}           {return ATKEYWORD;}

"!{w}important"      {return IMPORTANT_SYM;}

{num}em               {return EMS;}
{num}ex               {return EXS;}
{num}px               {return LENGTH;}
{num}cm               {return LENGTH;}
{num}mm               {return LENGTH;}
{num}in               {return LENGTH;}
{num}pt               {return LENGTH;}
{num}pc               {return LENGTH;}
{num}deg              {return ANGLE;}
{num}rad              {return ANGLE;}
{num}grad             {return ANGLE;}
{num}ms               {return TIME;}
{num}s                {return TIME;}
{num}Hz               {return FREQ;}

```

```

{num}kHz           {return  FREQ;}
{num}{ident}       {return  DIMEN;}
{num}%             {return  PERCENTAGE;}
{num}              {return  NUMBER;}

"url("{w}{string}{w}")" {return  URI;}
"url("{w}{url}{w}")"    {return  URI;}
{ident}"("           {return  FUNCTION;}

U\+{range}         {return  UNICODERANGE;}
U\+{h}{1,6}-{h}{1,6} {return  UNICODERANGE;}

.                  {return  *yytext;}

```

C6. Comparación de los comandos en CSS2 y CSS1

Hay algunas diferencias en la sintaxis especificada en la recomendación CSS1 (Ver [W3CSTYLE]) y la definida para CSS2. La mayoría de éstas se deben a los nuevos comandos en CSS2 que no existían en CSS1. Otras son debido a que la gramática ha sido reescrita para que sea más legible. Sin embargo, hay algunos cambios incompatibles, en lo que se consideró eran errores en la sintaxis de CSS1. Éstos se detallan a continuación:

- Las hojas de estilo CSS1 sólo podían estar en una codificación de 1 byte por carácter, tal como ASCII e ISO-8859-1. CSS2 no tiene tal limitación. En la práctica, había algunas pequeñas dificultades en extrapolar los comandos de CSS1, y algunas AU han aceptado la codificación de 2 bytes.
- CSS1 sólo permitía cuatro dígitos hexadecimales después de la barra invertida (\) para referirse a los caracteres de Unicode, CSS2 permite seis. Además, CSS2 permite una carácter de espacio en blanco para delimitar la secuencia de escape. Ej., de acuerdo a CSS1, la cadena "\abcdef" tiene 3 letras (\abcd, e, y f), de acuerdo a CSS2 tiene solamente una (\abcdef).
- El carácter de tabulador (ASCII 9) no estaba permitido en las cadenas. Sin embargo, como las cadenas en CSS1 sólo eran usadas para los nombres de las fuentes y para los URL, la única forma de que esto pueda conducir a la incompatibilidad entre CSS1 y CSS2 es que una hoja de estilo contenga una familia de fuentes con un tabulador en su nombre.
- De modo similar, los saltos de línea (escapadas con una barra invertida) no estaban permitidas en las cadenas en CSS1.
- CSS2 analiza un número seguido inmediatamente por un identificador como un comando DIMEN (es decir, una unidad desconocida), CSS1 lo analizaba como un número y un identificador. Eso significa que en CSS1, la declaración *font: 10pt/1.2serif* era correcta, como lo era *font: 10pt/12pt serif*; en CSS2, se requiere un espacio antes de "serif". (Algunas AU han aceptado el primer ejemplo, pero no el segundo.)
- En CSS1, un nombre de clase podía comenzar con un dígito (".55ft"), a menos que fuera una dimensión (".55in"). En CSS2, tales clases son analizadas como dimensiones desconocidas (para permitir futuros agregados de nuevas unidades). Para convertir ".55ft" en una clase válida, CSS2 requiere que el primer dígito sea escapado (".\35 5ft").

C7. Referencias

[FLEX]

"FLEX: The Lexical Scanner Generator", Version 2.3.7, ISBN 1882114213

[W3CSTYLE]

Recurso del W3C sobre hojas de estilo en la web. Disponible en <http://www.w3.org/pub/WWW/Style>.

[YACC]

"YACC - Yet another compiler compiler", S. C. Johnson, Technical Report, Murray Hill, 1975.

Apéndice D

Glosario de términos

Ferran Perdrix Sapiña



Grupo GRIHO (UdL)
C/Jaume II nº 69
ferran@griho.net



Glosario de Términos

A lo largo de este glosario de términos las comillas simples ('...') se utilizan para encerrar código HTML y CSS. Veamos los términos más destacados relacionados con este tutorial:

analizador CSS (*CSS parser*)

un Agente de Usuario que lee hojas de estilo CSS

atributo (*attribute*)

atributo HTML

AU (*UA*)

Agente de Usuario, normalmente un navegador, explorador o browser

autor (*author*)

el autor del documento HTML

características avanzadas de CSS1 (*CSS1 advanced features*)

características descritas en esta especificación pero que no se consideran parte de las características básicas de CSS1

características básicas de CSS1 (*CSS1 core features*)

la parte de CSS1 que se requiere a todos los AAUU conformes con CSS1

CSS

Hojas de Estilo en Cascada (*Cascading Style Sheets*)

CSS1

Hojas de Estilo en Cascada, Nivel 1. CSS1 es un mecanismo simple de hojas de estilo para la Web.

declaración (*declaration*)

una propiedad (p.ej. *font-size*) y su correspondiente valor (p.ej. '12pt')

dimensiones intrínsecas (*intrinsic dimensions*)

la anchura y la altura tal y como las define el elemento por sí mismo, sin las restricciones de lo que le rodea. En esta especificación se da por hecho que todos los elementos reemplazados – y sólo los elementos reemplazados – van con dimensiones intrínsecas.

diseñador (*designer*)

el diseñador de una hoja de estilo

documento (*document*)

documento HTML

elemento (*element*)

elemento HTML

elemento en bloque (*block-level element*)

un elemento que tiene un salto de línea antes y otro después (p.ej., 'H1' en HTML)

elemento en línea (*inline elements*)

un elemento que no tiene un salto de línea antes y otro después (p.ej., 'STRONG' en HTML)

elemento hijo (*child element*)

un subelemento (subelement) en terminología SGML

elemento padre (*parent element*)

el elemento contenedor (containing element) en terminología SGML

elemento reemplazado (*replaced element*)

un elemento acerca del cual el formateador CSS sólo conoce sus dimensiones. En HTML, los elementos 'IMG', 'INPUT', 'TEXTAREA', 'SELECT' y 'OBJECT' pueden ser ejemplos de elementos reemplazados. Por ejemplo, el contenido del elemento 'IMG' se reemplaza normalmente por la imagen a la que apunta el atributo SRC. CSS1 no define cómo se averiguan las dimensiones intrínsecas.

extensión HTML (*HTML extension*)

códigos creados por compañías de AAUU, la mayoría de las veces para implementar ciertos efectos visuales. Los elementos "FONT", "CENTER" y "BLINK" son ejemplos de extensiones HTML, así como el atributo "BGCOLOR". Uno de los objetivos de CSS es proporcionar una alternativa a las extensiones HTML.

hoja de estilo (*style sheet*)

un conjunto de reglas

HTML

Lenguaje de Etiquetas para Hipertexto (*HyperText Markup Language*), una aplicación del SGML

lector (*reader*)

la persona para la que se representa el documento

lienzo (*canvas*)

la parte de la superficie de dibujo del AU en la cual se representan los elementos

peso (*weight*)

la prioridad de una regla

propiedad (*property*)

un parámetro estilístico sobre el que se puede actuar a través de CSS. Esta especificación define una lista de propiedades y sus valores correspondientes.

pseudo-clase (*pseudo-class*)

las pseudo-classes se utilizan en CSS para permitir que información externa al propio documento HTML (p.ej., el hecho de que un vínculo haya sido visitado o no) sirva para clasificar elementos

pseudo-elemento (*pseudo-element*)

los pseudo-elementos se utilizan en CSS para hacer referencia a objetos tipográficos (p.ej., la primera línea de un elemento) en lugar de a elementos estructurales

regla (*rule*)

una declaración (p.ej., *'font-family : helvetica'*) y su selector (p.ej. 'H1')

secuencia ficticia de etiquetas (*fictional tag sequence*)

un artificio para describir el comportamiento de las pseudo-clases y los pseudo-elementos

selector (*selector*)

una cadena que identifica a qué elementos se aplica la regla correspondiente. Un selector puede ser o bien un selector simple (p.ej., 'H1') o bien un selector contextual (p.ej. 'H1 B'), el cual consiste en varios selectores simples.

selector contextual (*contextual selector*)

un selector que corresponde a elementos según la posición de éstos en la estructura del documento. Un selector contextual consiste en varios selectores simples. P.ej., el selector contextual 'H1.Inicial B' consiste en dos selectores simples, 'H1.Inicial' y 'B'.

selector simple (*simple selector*)

un selector que corresponde a elementos según el tipo y/o atributos de éstos, y no según su posición en la estructura del documento. P.ej., 'H1.Inicial' es un selector simple.

SGML

Lenguaje de Etiquetas Generalizado Estándar (*Standard Generalized Markup Language*), del cual es aplicación el HTML

tamaño de fuente (*font-size*)

el tamaño para el que una fuente ha sido diseñada. Normalmente, el tamaño de una fuente es aproximadamente igual a la distancia entre la parte inferior de la letra con descendente más baja hasta la parte superior de la letra con ascendente más alta y (opcionalmente) con una marca diacrítica.

tipo de elemento (*element type*)

un identificador genérico (*generic identifier*) en terminología SGML

usuario (*user*)

sinónimo de lector