Manual Rápido de CSS

Una aproximación para el estudiantes con interés en el dominio y consulta de CSS para la aplicación de estilos en el desarrollo de páginas Web

Este manual se constituye en un material de complemento a la clase magistral sobre el tema de desarrollo de páginas Web. Tiene como propósito apoyar a los estudiantes en la preparación para el día del laboratorio y para la realización de su proyecto del curso. El manual tiene 2 grandes secciones, la primera trata sobre el desarrollo de hojas de estilo CSS, la segunda sección presenta un compendio de enlaces de interés a ejemplos y otros manuales.

Beatriz Eugenia Florián Gaviria
03/10/2007

Tabla de contenido

Tabla de Figuras

1) CSS

Las Hojas de Estilo (o CSS, por Cascading StyleSheets) son un mecanismo que permiten aplicar formato a los documentos escritos en HTML ó XHTML (y en otros lenguajes estructurados, como XML) separando el contenido de las páginas de su apariencia. Para el diseñador, esto significa que la información estará contenida en la página HTML, pero este archivo no debe definir cómo será visualizada esa información, sino la hoja de estilos asociada a él. Las indicaciones acerca de la composición visual del documento estarán especificadas en el archivo de la CSS.

En un ejemplo sencillo, una página HTML podría tener uno o varios encabezados definidos con la etiqueta <h1>, se quieren decorar con tipografía Arial, de 19 puntos, en color azul y alineación central. Por un lado el archivo XHTML contendría la definición de encabezados y por otra el archivo CSS contendría la definición de los estilos a aplicar para la etiqueta de encabezados así:

Parte del contenido del archivo pagina.html

```
…
<h1>1er Título de nivel 1</h1>
…
<h1>1er Título de nivel 1</h1>
…
```

Estilo para <h1> definido en el archivo estilos.css

```
h1 {
   font-family: Arial;
   font-size: 19pt;
   color: blue;
   text-align: center;
}
```

Luego, agregamos un enlace en el documento HTML a la hoja de estilo creada. El enlace se crea dentro de la cabecera del documento. Nuestro documento pagina.html quedaría así:

```
<html lang="es">
  <head>
    <title>Introducción a las CSS</title>
    <link rel="stylesheet" type="text/css" href="estilos.css" />
  </head>
  <body>
    …
    <h1>1er Título de nivel 1</h1>
    …
    <h1>1er Título de nivel 1</h1>
    …
  </body>
</html>
```

La etiqueta ***<link>*** especifica:

- el tipo de vínculo: a una hoja de estilo ("stylesheet")
- la ubicación de la hoja de estilo a través del atributo "href"
- el tipo de hoja de estilo que se vincula: "text/css"

Ahora, todos los encabezados h1 de la página tendrán el aspecto que hemos definido. Se puede asociar un mismo archivo CSS a diferentes archivos HTML del sitio Web y así lograr consistencia de estilos entre las páginas que lo comprenden.

La aplicación de estilos toma en cuenta dos conceptos importantes, la ***herencia*** y la ***cascada***.

Herencia

Cada página HTML está compuesta por una serie de elementos (títulos, párrafos, listas, tablas, etc.) organizados en una estructura donde cada elemento está contenido por otro elemento, que a su vez puede estar contenido por otro.

En esta estructura existe un elemento raíz que es el que actúa de contenedor de todos los demás elementos. En HTML se puede considerar como elemento raíz al elemento <body> o al elemento <html>.

La importancia de este hecho es que cada elemento hereda las propiedades del elemento que lo contiene (llamado el elemento padre). Quiere decir que si especificamos la propiedad color: red para <BODY> en la hoja de estilos, todos los elementos de la página heredarán esta característica (como listas y párrafos) y no será necesario especificar nuevamente la propiedad color en cada uno de ellos.

Aquí es necesario hacer algunas precisiones:

1. No todas las propiedades son hereditables y para cada propiedad se define si ésta se hereda o no.
2. El valor *inherit* puede aplicarse a cualquier propiedad de los elementos. Este valor puede usarse para reforzar explícitamente la herencia de una propiedad o para lograr que un elemento herede de su padre una propiedad que de otro modo no sería heredada.

```
*.peligro {  color: inherit;  }
```

3. Cuando se asigna una propiedad a un elemento, el valor especificado reemplaza al valor heredado, como si se sobrescribiera el valor heredado del padre..
4. Los elementos heredan los valores computados (expresados con base a porcentajes o tamaños relativos) del padre, no los valores especificados. Todas las propiedades tienen siempre un valor asignado. El valor por defecto, el valor heredado o el valor asignado en el archivo CSS).

Cascada

Cuando varias hojas de estilos se aplican sobre un documento y hay conflictos sobre la regla que debe aplicarse. El orden de la cascada determina, en el caso en que existan reglas incompatibles, cuál de ellas tiene preponderancia.

Las hojas de estilo pueden tener tres orígenes:

- ***El autor***: como hemos visto hasta ahora, puede hacerlo a través de una hoja externa que es la forma más elegante. También se pueden hace la hoja de estilos incrustada, incluso definiendo el estilo en línea.
- ***El usuario***: también puede especificar su propia hoja de estilo. Esta posibilidad puede resultar de gran ayuda para aquellas personas con discapacidades visuales o, simplemente, para quienes deseen adaptar las páginas a sus preferencias. Cada navegador tiene su forma particular de proveer al usuario la posibilidad de especificar una hoja de estilo propia.
- ***La aplicación del usuario***: el navegador también aplica una hoja de estilo predeterminada que presenta los elementos de la página de modo que satisfagan las expectativas generales de presentación del documento.

Las siguientes son las normas que determinan la fuerza de las reglas de estilo:

1. La primera disposición se hace por el origen: las reglas de la hoja de estilo del autor tienen más fuerza que las del usuario y éstas, a su vez, prevalecen por sobre las del navegador.
2. La segunda disposición es por especificidad: los selectores más específicos tienen mayor fuerza que los selectores generales. Por ejemplo: UL {} es menos específico que UL LI {}. En la Recomendación CSS2 puede encontrar la explicación acerca de cómo se calcula la especificidad de un selector. Especificidad
3. Finalmente, se dispone por el orden especificado: si dos reglas tienen la misma fuerza, origen y especificidad, la última en ser especificada es la que vence. Las reglas en las hojas de estilo importadas se considera que están antes que cualquier regla en la propia hoja de estilo.

Reglas

Un documento CSS está compuesto por una colección de reglas. Como lo más probable es que haya más de una regla, una colección de reglas se llama frecuentemente *rule-set* o juego de reglas. Una regla consiste en un selector acompañado de un bloque de declaraciones.

El selector determina a que elementos en el árbol del documento se aplicarán las reglas. El bloque de declaración es una lista, separada por punto y coma (";"), de una o varias declaraciones (o ninguna), contenidas entre llaves ("{" y "}"). Cada declaración puede o estar vacía o estar compuesta por una pareja propiedad/valor, donde la propiedad y el valor se separan con dos puntos (":").

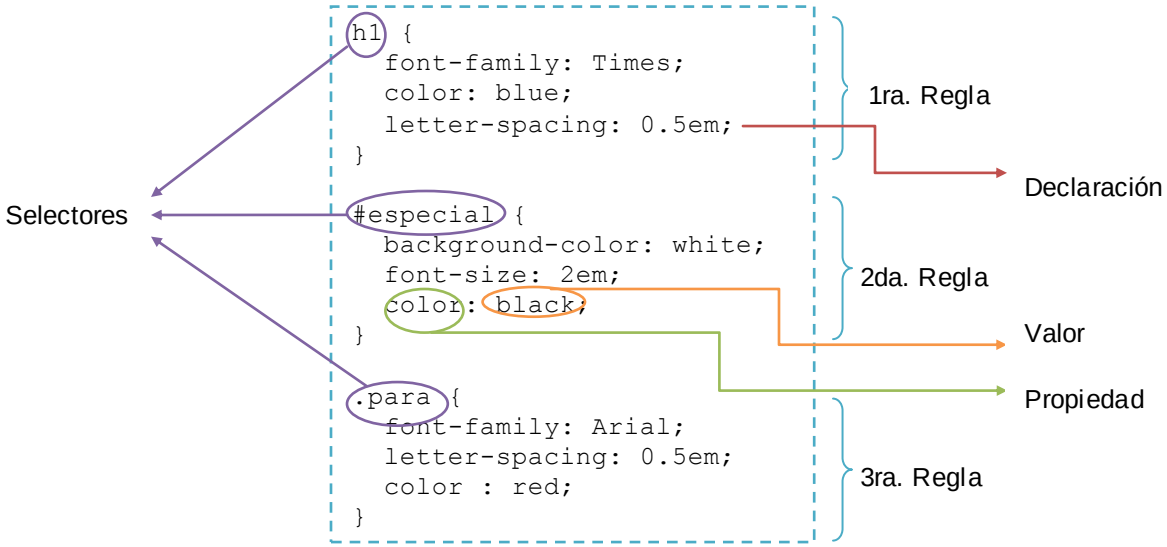

Figura 1. Elementos de un documento CSS

En el ejemplo, el documento CSS contiene tres reglas. Los selectores de estas tres reglas serian *h1, #especial* y *.para* respectivamente. Cada regla define tres declaraciones propias. En la primera regla el selector es h1 y el bloque de declaración es todo lo comprendido entre las llaves y las llaves en sí mismas. El bloque de declaración consiste en tres declaraciones separadas por punto y coma. Como el punto y coma sólo es requerido para separar declaraciones, el punto y coma final no es necesario, pero es legal. La primera declaración especifica una propiedad de *font-family* con un valor de *Times*.

Selectores de CSS

Los selectores son los nombres que identifican un estilo o regla. Estos nombres definen patrones que determinan a qué elementos del árbol del documento, donde la hoja de estilos es utilizada, serán aplicadas las declaraciones del estilo. Pueden utilizarse patrones complejos para ser más específico sobre qué elementos van a aplicarse las reglas, basados en la posición del elemento en el árbol del documento.

Selector Universal

El selector universal se marca con un asterisco "*" y abarca a todos los elementos del documento XHTML. El siguiente ejemplo aplicaría un fondo amarillo a todos los elementos que soportasen la propiedad background-color salvo que se usase un selector más específico que lo sobrescriba.

```
* {  background-color: yellow;  }
```

El siguiente ejemplo podría ser aplicado a todos los elementos del documento XHTML que tengan como nombre de clase *enfatizado* (class="enfatizado")

```
*.enfatizado {  background-color: blue;  }
```

Cuando el asterisco forma parte de un patrón, como en el ejemplo anterior, puede ser omitido tranquilamente. Un patrón de .enfatizado sería igual de efectivo que *.enfatizado

Selector por Tipo

Los selectores por tipo afectan a todas las apariciones de un elemento en el documento XHTML. El siguiente ejemplo aplica un fondo blanco a todos los elementos <p> del documento.

```
p {  background-color: white;  }
```

Cuando varios selectores van a compartir el bloque de declaraciones, los selectores pueden ser agrupados separándolos con una coma. El siguiente ejemplo aplicará un fondo negro a todos los elementos <h1> y <h2>

```
h1, h2 {  background-color: black;  }
```

Selector por Clase

El atributo class se indica con un punto y se aplica a todos los elementos del documento con esa clase. El siguiente ejemplo aplicará un fondo amarillo a todos los elementos <li> marcados por la clase *entrega* (<li class="entrega">)

```
li.entrega {  background-color: yellow;  }
```

Si la clase va a ser aplicada a un número de elementos, estos pueden definirse de forma individual, agruparse o utilizar el selector universal (*.nombreClase ó . nombreClase)

Selector por ID

El atributo id debe ser único dentro del documento web, por tanto sólo puede aplicarse a una única aparición de un elemento dentro del árbol del documento. El selector CSS para indicar los id es la

almohadilla #. El siguiente ejemplo aplicará un fondo amarillo al elemento <h2> que tenga el id único *ofertas* (<h2 id="ofertas">)

```
h2#ofertas {  background-color: yellow; }
```

Dado que el selector id debe ser único, es totalmente seguro utilizar el selector universal (*#idNombre o #idNombre). Para mejorar la legibilidad y la facilidad en el mantenimiento suele ser buena idea conservar el nombre del elemento en su lugar (h2#idNombre).

SELECTORES DESCENDENTES

El árbol del documento está organizado de tal forma que cada elemento del árbol tenga un elemento padre, exceptuando el elemento raíz. Los selectores descendentes se usan para afectar a todos los elementos que descienden de un determinado padre. Los selectores descendentes se crean a partir de selectores separados por espacios, donde cada selector subsiguiente debe estar contenido en su predecesor. El siguiente ejemplo aplicará un fondo amarillo a cualquier elemento contenido dentro de un párrafo, <p>.

```
p strong {  background-color: yellow;  }
```

Ejemplo: hoja de estilos aplicando diferentes tipos de selectores

```
/* todos los elementos 'h1' */
h1 { font-family: Times; }

/* los elementos 'h1' y 'h2' */
h1, h2 { color: blue; }

/* elementos 'em' dentro de elementos 'h1'/
h1 em { color : red; }

/* elementos con clase 'para' */
.para { font-family: Arial; }

/* elementos 'h1' con id 'z98y' */
h1#z98y { letter-spacing: 0.5em; }
```

Investigar sobre: Selectores Hijos, Selectores Hermanos Adyacentes y Selectores Atributo

Pseudo-Selectores de CSS

Todos los selectores vistos hasta el momento se refieren a elementos que podemos encontrar dentro del código HTML. Los pseudo selectores se utilizan en escenarios donde la posición del elemento dentro del árbol del documento no sea suficiente como selector, como en la primera línea de un párrafo.

Los pseudo elementos y las pseudo clases son utilizados para aplicar formato basado en información fuera del árbol del documento y están precedidos de "dos puntos" :

PSEUDO ELEMENTOS

Son llamados pseudo-elementos porque en realidad no existen en el documento fuente (ninguna marca identifica la primer letra de un párrafo, por ejemplo) pero son muy útiles para seleccionar elementos importantes dentro de la composición.

CSS 2.1 tiene cuatro pseudo elementos:

- ***:first-line*** Permite aplicar un estilo determinado a la primera línea de un párrafo.
- ***:first-letter*** Permite seleccionar la primer letra de un párrafo, generalmente para utilizarla como capitular (en un tamaño mayor que el resto del párrafo)
- ***:before*** Permite insertar un contenido antes de un elemento determinado y definir el estilo del contenido insertado
- ***:after*** Permite insertar un contenido después de un elemento determinado y definir el estilo del contenido insertado

Los pseudo-elementos :first-letter y :first-line permiten conseguir el efecto de colocar la primer letra o la primer línea de un párrafo (o ambas cosas a la vez) en un formato diferenciado.

Por otro lado, :before (antes) y :after (después) permiten insertar un contenido antes o después de un elemento determinado y definir el estilo del contenido insertado. La propiedad 'content', junto con estos pseudo-elementos, especifican lo que se inserta.

```
p:first-line { text-transform: uppercase; }
```

Con la regla anterior conseguiríamos que todas las letras de la primera línea de los párrafos fueran convertidas en mayúsculas. Por supuesto, la medida de esta primera línea será determinada por el tamaño de la fuente, el ancho de la ventana, etc.

Sólo las siguientes propiedades se aplican al pseudo-elemento :first-line: propiedades de la fuente, propiedades del color, propiedades del fondo, 'word-spacing', 'letter-spacing', 'text-decoration', 'vertical-align', 'text-transform', 'line-height', 'text-shadow' y 'clear'.

```
p {
  font-size: 12pt;
  line-height: 12pt;
}

p:first-letter {
  font-size: 200%;
  font-weight: bold;
}
```

Las reglas anteriores determinan que los párrafos tengan una capitular en negritas del doble de tamaño que la fuente del párrafo.

El siguiente ejemplo aplica un font-size de 2em a la primera letra de un párrafo.

```
p:first-letter {  font-size: 2em; }
```

Podemos lograr que antes de cada elemento <h3> aparezca el texto "Tema:" sin necesidad de tener que escribirlo en cada título. También podemos hacer que cada párrafo termine con un pequeño ícono o poner "Fin" al pie de cada página usando las siguientes reglas.

```
h3:before {  content: "Tema: ";  }
p:after {  content: url("icono.gif");  }
body:after {
  content: "Fin";
  display: block;
}
```

En la última regla hemos especificado también "display: block" para que la palabra "Fin" comience en una nueva línea (como si fuese un nuevo párrafo).

PSEUDO CLASES

CSS 2 tiene las siguientes pseudo clases:

- ***:first-child***,
- ***:link*** y ***:visited*** Permiten definir un estilo para los enlaces y otro para los enlaces visitados
- ***:active***, ***:hover*** y ***:focus*** Cambian de acuerdo a las acciones del usuario
  - :hover se aplica cuando el cursor del mouse señala el elemento.
  - :active se aplica cuando el elemento es activado (por ejemplo, cuando el usuario presiona el botón del mouse).
  - :focus se aplica cuando el elemento recibe el foco.
- ***:lang*** Permite seleccionar elementos en base a su idioma. (una pseudo clase para lenguaje). Como cada idioma tiene sus propias convenciones con respecto al formato (uso de itálicas, sangrías o comillas, por ejemplo), esta pseudo-clase nos permite describir cómo debe aparecer un elemento según el idioma usado. El código para identificar el idioma consta generalmente de dos letras: "es" español, "en" inglés, "de" alemán, "fr" francés, etc.

El siguiente ejemplo aplicaría un color amarillo a todos los elementos de ancla que sean el primer hijo de cualquier elemento

```
a:first-child {  color: yellow;  }
```

El siguiente ejemplo da color rojo a los enlaces, color gris a los enlaces visitados, color azul al enlace sobre el cual se posa el ratón y color fucsia al enlace activado.

```
a:link {  color: red;  }
a:visited {  color: gray;  }
a:hover {  color: blue;  }
a:active {  color: fuchsia;  }
```

El siguiente ejemplo aplicaría un fondo amarillo pálido a los párrafos con un atributo lang de Francés

```
p:lang(fr) {  background-color: #ffc;  }
```

La siguiente regla especifica el tipo de comillas que debe usar un elemento Q en francés:

```
Q:lang(fr) { quotes: '« ' ' »'; }
```

Ejemplo: Hoja de estilos con definición de pseudo clases y pseudo elementos

```
/*** Pseudo clases para enlaces ***/
a:link {  color: red;  } /* enlaces sin visitar */
a:visited {  color: gray;  } /* enlaces visitados */
a:active {  color: black;  } /* enlaces activos */

/*** Pseudo elementos para tipografía ***/
p:first-line{text-transform:uppercase;} /*1ra. línea párrafo
*/
p:first-letter{font-size: 2em;} /* 1ra. letra parrafo */
```

Propiedades Iniciales de CSS

En esta sección las propiedades se han agrupado en categorías adecuadas. Para cada categoría se especifica la siguiente información:

- Nombre de la propiedad
- Conjunto de posibles valores de la propiedad
- Valor por defecto (en negrita o especificado separadamente)
- Elementos a los cuales se aplica la propiedad (todos a menos de que se indique algo específico)
- El símbolo ¥ indica que la propiedad no es heredada

Las propiedades relativas al manejo de cajas se explicarán en la siguiente sección de manejo de cajas. Las propiedades se definen con una gramática que parte de la notación y unidades a continuación.

NOTACIÓN

a b	a seguido de b
[a b]	agrupación de a y b
a | b	a ó b (excluyente, solo a o solo b)
a || b	alguno de los dos, o ambos en cualquier orden
a?	a es opcional
a*	ninguna o más ocurrencias de a
a+	una o más ocurrencias de a
a{1, 4}	a ocurre al menos una vez y máximo 4 veces

UNIDADES

Unidades de Longitud

<longitud>	(+ | -)? <numero> <unidad>
<numero>	<digito>+[. <digito>*]?
<digito>	0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<unidad>	<unidad-absoluta> | <unidad-relativa>
<unidad-absoluta>	mm | cm | in | pt | pc
<unidad-relativa>	em | ex | px

Para las unidades absolutas

- ***mm*** representa milímetros,
- ***cm*** representa centímetros,
- ***in*** representa pulgadas (1in=2.54cm)
- ***pt*** representa puntos (1pt=1/72in)
- ***pc*** representa picas (1pc=12pt)

Para las unidades relativas

- ***1em*** representa el alto de la letra M mayúscula de la fuente utilizada
- ***ex*** representa el alto de la letra X de la fuente utilizada y
- ***px*** representa pixeles.

Unidades de Porcentaje

<porcentaje>	<numero>%

URL

<url>	url(text)

Unidades de Color

<color>	<nombre-color> | <rgb-color>
<nombre-color>	aqua | black | blue | fuchsia | gray | green | lime | maroon | navy | olive | purple | red | silver | teal | white | yellow
<rgb-color>	#<hex><hex><hex> | #<hex><hex><hex><hex><hex><hex> | rgb(<numero>, <numero>, <numero>) | rgb(<porcentaje>, <porcentaje>, <porcentaje>)

En CSS hay varias maneras de indicar un color. La forma más sencilla es estableciendo el nombre del color directamente. Existen 16 nombres de color en inglés válidos, como lo indica el elemento <nombre-color> en la definición de unidades de color anterior. La desventaja de este método es la limitación de colores, puede que exista algún color que queramos establecer y no tenga nombre. Además, debemos aprendernos los nombres. Debido a estos inconvenientes la notación que más se usa es la rgb por números hexadecimales ó la de rgb por números decimales. En nuestra definición de unidades de color estas notaciones están representadas por el elemento <rgb-color>.

Los colores en nuestro monitor está formados por tres haces de luz: rojo, verde y azul. Se llama sistema RGB (Red Green Blue). Podemos elegir un color indicando el valor de sus componentes rojo, verde y azul por separado. Este valor puede variar de 0 a 255 (se usan 8 bits para representar este rango). Por ejemplo, si queremos un amarillo puro (yellow), ponemos 255 para el rojo, 255 para el verde y 0 para el azul. Si queremos un verde oscuro (sin nombre en inglés asociado) podríamos utilizar 0 para el rojo, 161 para el verde y 32 para el azul. Si queremos un naranja podríamos utilizar 255 para el rojo, 128 para el verde y 0 para el azul. Si queremos un azul podríamos utilizar 10 para el rojo, 10 para el verde y 245 para el azul.

```
/* Enlaces no visitados en amarillo puro */
a:link { color: rgb(255,255,0); }

/* Enlaces visitados en un verde oscuro */
a:visited { color: rgb(0,161,32); }

/* Enlaces con ratón sobre él en un naranja*/
a:hover {  color: rgb(255,128,0);   }

/* Enlaces activos en un azul */
a:active {  color: rgb(10,10,245);   }
```

El rango de valores (0..255) puede ser representado con dos dígitos hexadecimales, que van desde el 00 hasta el FF. La forma de escribir un color completo en notación hexadecimal es #RRGGBB, donde la almohadilla indica que se trata de un color hexadecimal, y RR, GG y BB son los dígitos correspondientes al rojo, verde y azul, respectivamente. Entonces el amarillo de nuestro ejemplo anterior sería #FFFF00, el verde #00A120, el naranja #FF8000 y el azul #0A0AF5.

```
/* Enlaces no visitados en amarillo puro */
a:link { color: #FFFF00; }

/* Enlaces visitados en un verde oscuro */
a:visited { color: #00A120; }

/* Enlaces con ratón sobre él en un naranja*/
a:hover {  color: #FF8000; }

/* Enlaces activos en un azul */
a:active {  color: #0A0AF5;   }
```

La mayoría de los colores que usaremos tendrán por cada componente los mismos dígitos. Por ejemplo: #FF0000 (rojo), #0000AA (azul marino) o #000000 (negro). Estos colores los podemos abreviar y dejarlos en tres cifras. Por ejemplo, #F00 es equivalente a #FF0000. Sin embargo, colores como #A9A9A9 no pueden ser abreviados.

Por tanto, para conseguir el color blanco, podemos hacerlo de cinco formas diferentes.

- color: white;
- color: rgb (255,255,255);
- color: rgb (100%, 100%, 100%);
- color:#FFFFFF;
- color:#FFF;

PROPIEDADES DE FONDO

background-color (¥)

Valor: <color> | **transparent**

background-image (¥)

Valor: <url> | **none**

background-repeat (¥)

Valor: **repeat** | repeat-x | repeat-y | no-repeat

background-attachment (¥)

Valor: **scroll** | fixed

background-position (¥)

Valor: [<porcentaje> | <longitud>]{1,2} | [top | center] | [left | center | right]
Por defecto: 0% 0%
Se aplica a: bloques y elementos remplazados
Valor por porcentaje: calculado de acuerdo al tamaño del elemento

background (¥)

Valor: <background-color> || <background-image>
<background-repeat> || <background-attachment>
<background-position>
Por defecto: no definido para las propiedades resumidas
Valor por porcentaje: permitido sobre <background-position>

PROPIEDADES DE FUENTE

color

Valor: <color>
Por defecto: UA specific

font-family

Valor: [[<nombre-de-familia> | <familia-genérica>],]*[<nombre-de-familia> | <familia-genérica>]
Por defecto: UA specific

<familia-genérica> serif | sans-serif | cursive | fantasy | monospace

font-style

Valor: **normal** | italic | oblique

font-variant

Valor: **normal** | small-caps

font-weight

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

font-size

Valor:	<tamaño-absoluto> | <tamaño-relativo> | <longitud> | <porcentaje>
Valor por porcentaje:	relativo al tamaño de fuente del elemento padre
<tamaño-absoluto>	xx-small | x-small | small | medium | large | x-large | xx-large
<tamaño-relativo>	larger | smaller

font

Valor:	[<font-style> || <font-variant> || <font-weight>]? <font-size> [/ <line-height>]? <font-family>
Por defecto:	no definido para las propiedades resumidas
Valor por porcentaje:	permitido sobre <font-size> y <line-height>

PROPIEDADES DE TEXTO

word-spacing

Valor:	normal | <longitud>

letter-spacing

Valor:	normal | <longitud>

text-transform

Valor:	capitalize | uppercase | lowercase | none

text-decoration (¥)

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

text-align

Valor:	left | right | center | justify
Por defecto:	específicos americanos
Se aplica a:	elementos de bloque

vertical-align (¥)

Valor:	baseline | sub | super | top | text-top | middle | bottom | text-bottom | <porcentaje>
Se aplica a:	elementos de línea
Valor por porcentaje:	con respect al valor 'line-height' del elemento

white-space

Valor:	normal | pre | nowrap

PROPIEDADES DE LISTAS

list-style-type

Valor:	disc | circle | square | decimal | lower-roman | upper-roman | lower-alpha | upper-alpha | none
Se aplica a:	elementos con valor de 'display' igual a 'list-item'

list-style-image

Valor: <url> | none
Se aplica a: elementos con valor de 'display' igual a 'list-item'

list-style-position

Valor: inside | outside
Se aplica a: elementos con valor de 'display' igual a 'list-item'

list-style

Valor: <list-style-type> || <list-style-position> || <list-style-image>
Por defecto: no definido para las propiedades resumidas
Se aplica a: elementos con valor de 'display' igual a 'list-item'

EJEMPLOS DE USO DE PROPIEDADES INICIALES

```
/* Cuerpo de la página con una imagen de fondo repetida */
body {
      background-image: url(images/fondo.gif);
      background-repeat: repeat;
}
/* Enlaces sin visitar sin decorado y de un tono verde  */
a:link {
      color: #00A120;
      text-decoration: none;
}
/* Enlaces bajo el ratón, subrayados y de color negro */
a:hover {
      color: #000000;
      text-decoration: underline;
}

/*Enlaces visitados sin decorado y de un tono verde claro*/
a:visited {
      color: #B6DEB3;
      text-decoration: none;
}

/* Listas con class=listasConImagen, con una imagen como
viñeta */
li.listasConImagen {
      list-style-image: url(images/vinneta.jpg);
}

/*Listas con class=listasRomanas, con numeración romana*/
li.listasRomanas {
      list-style-type: lower-roman;
}

/* Encabezados <h1> con id=tituloPagina en Arial, a 23
pixeles, en negrilla, con un tono de verde y en mayúsculas*/
h1#tituloPagina {
      font-family: Arial, Helvetica, sans-serif;
      font-size: 23px;
      font-weight: bold;
      color: #097900;
      text-transform: capitalize;
}
```

```
/* Titulos, subtitulos y texto normal de paginas */

/* Una clase para asignar a todos los títulos*/
.tituloPagina {
	font-family: Arial, Helvetica, sans-serif;
	font-size: 23px;
	font-style: normal;
	font-weight: bold;
	font-variant: normal;
	color: #097900;
	letter-spacing: 1pt;
	text-align: left;
	vertical-align: middle;
	word-spacing: normal;
	text-transform: capitalize;
}

/* Una clase para asignar a todos los subtítulos*/
.subtitulos {
	font-family: Arial, Helvetica, sans-serif;
	font-size: 14px;
	font-style: normal;
	font-weight: bold;
	color: #097900;
}

/* Una clase para asignar a todos los párrafos normales*/
<p>.normal {
	font-family: Arial, Helvetica, sans-serif;
	font-size: 12px;
	font-style: normal;
	font-weight: normal;
	color: #636466;
	text-align: justify;
	padding-left: 20px;
	padding-right: 20px;
}

/* Un estilo especial para el mensaje de copyright */

#copyright {
	font-family: Arial, Helvetica, sans-serif;
	font-size: 9px;
	font-style: normal;
	font-weight: normal;
	color: #636466;
	text-align: right;
	vertical-align: middle;
}
```

El Modelo de Cajas de CSS

El modelo de caja de CSS describe las cajas rectangulares que son generadas por los elementos. El elemento raíz del documento (<html> o, mejor aún, <body> en (X)HTML) genera una caja que actúa como bloque de contención de las cajas generadas subsecuentemente. A su vez, cada caja puede actuar como bloque de contención de otras cajas generadas por sus elementos descendientes.

Esta estrategia permite tener el control no solo de las propiedades del elemento en sí (su color, la fuente usada para el texto, etc.), sino también de las propiedades de la caja generada por el elemento (sus márgenes, sus bordes, la posición dentro del documento). Esto nos permita componer visualmente la página de un modo mucho más rico y flexible que con tablas de (X)HTML.

Este modelo de caja es el que permite, por ejemplo, que cualquier elemento de la página pueda recibir todas de las propiedades definidas en CSS. Por eso, a diferencia de lo que sucede con (X)HTML, todos los elementos pueden tener una imagen de fondo o un borde (sin necesidad de usar una tabla para eso); también es posible darle una ubicación precisa a cualquier elemento dentro de la página, ya sea con respecto a la pantalla o a otros elementos. Mejor aún, nos da la posibilidad de usar un lenguaje como JavaScript para conseguir efectos muy interesantes modificando las propiedades de las cajas (moverlas de su posición, mostrarlas o esconderlas, cambiar su tamaño, etc.).

Cada caja tiene un área de contenido (por ejemplo, texto, una imagen, etc.) y las áreas circundantes opcionales de padding (relleno), border (borde) y margin (margen).

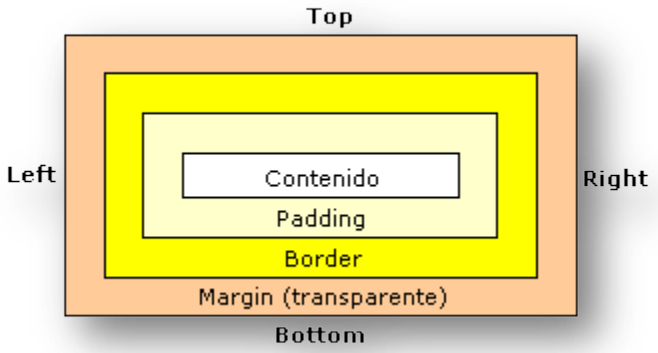

Figura 2. Elementos de Cajas en CSS

En esta sección veremos cómo se especifican las propiedades para cada una de estas áreas:

Las áreas de padding, border y margin se dividen en cuatro segmentos: top (superior), bottom (inferior), left (izquierdo) y right (derecho). De este modo podemos distinguir border-left, border-right, border-top y border-bottom (lo mismo para padding y margin).

El estilo del fondo de las distintas áreas de una caja es determinado del siguiente modo:

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

PROPIEDADES DE MARGEN PARA CAJAS

margin-top, (¥), margin-right (¥), margin-bottom (¥), margin-left (¥)

Las propiedades margin-top, margin-right, margin-bottom y margin-left determinan los márgenes superior, derecho, inferior e izquierdo de una caja respectivamente.

Valor:	<longitud> | <porcentaje> | auto
Por defecto:	0
Valor por porcentaje:	con relación al ancho de los elementos del padre

El valor auto es determinado por el navegador. Se pueden usar valores negativos (por ejemplo, `margin-top: -8px`).

margin (¥)

La propiedad 'margin' es una propiedad resumida que se utiliza para definir los cuatro márgenes a la vez.

Valor:	<longitud> | <porcentaje> | auto {1, 4}
Por defecto:	no definido para las propiedades resumidas
Valor por porcentaje:	se calcula con relación al ancho del bloque de contención

Los signos `{1,4}` significan que pueden especificarse de 1 a 4 valores para el ancho del margen:

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

PROPIEDADES DE RELLENO PARA CAJAS

padding-top (¥), padding-right (¥), padding-bottom (¥), padding-left (¥)

Las propiedades 'padding-top', 'padding-right', 'padding-bottom', 'padding-left' determinan el relleno superior, derecho, inferior e izquierdo de una caja respectivamente.

Valor:	<longitud> | <porcentaje>
Por defecto:	0
Valor por porcentaje:	se calcula con respecto al ancho del bloque de contención

A diferencia de las propiedades del margen, los valores para el relleno no pueden ser negativos.

padding (¥)

La propiedad 'padding' es una propiedad resumida que se utiliza para definir los cuatro rellenos a la vez.

Valor:	<longitud> | <porcentaje> {1, 4}
Por defecto:	no definido para las propiedades resumidas
Valor por porcentaje:	se calcula con respecto al ancho del bloque de contención

Los signos `{1,4}` significan que pueden especificarse de 1 a 4 valores para el ancho del relleno:

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

- Si hay cuatro valores se aplican al relleno superior, derecho, inferior e izquierdo, respectivamente.

PROPIEDADES DE BORDE PARA CAJAS

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

border-top-width (¥), border-right-width (¥), border-bottom-width (¥), border-left-width (¥)

Las propiedades 'border-top-width', 'border-right-width', 'border-bottom-width', 'border-left-width' determinan el ancho de los bordes superior, derecho, inferior e izquierdo de una caja respectivamente.

Valor: thin | **medium** | thick | <lenght>

El valor thin representa un borde fino, el valor medium representa un borde medio y el valor thick representa un borde grueso. Si se expresa el valor en forma explícita, las dimensiones del borde no pueden ser negativas. La interpretación de los tres primeros valores puede variar de un navegador a otro, pero no la relación entre ellos.

border-width (¥)

La propiedad 'border-width' es una propiedad resumida que se utiliza para definir el ancho de los cuatro bordes a la vez.

Valor: thin | medium | thick | <lenght> {1, 4}
Por defecto: no definido para las propiedades resumidas

Los signos `{1,4}` significan que pueden especificarse de 1 a 4 valores para el valor del ancho del borde.

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

border-top-color (¥), border-right-color (¥), border-bottom-color (¥), border-left-color (¥)

Estas propiedades determinan el color de los bordes superior, derecho, inferior e izquierdo de una caja respectivamente.

Valor: <color> | inherit
Por defecto: el valor de la propiedad 'color'

Si se especifica el color transparent el borde es transparente (no obstante, puede tener grosor). Si el color del borde de un elemento no es especificado, el navegador toma el valor de la propiedad 'color' del elemento para el color del borde.

border-color (¥)

La propiedad 'border-color' es una propiedad resumida que se utiliza para definir el color de los cuatro bordes a la vez.

Valor: <color>{1,4}
Por defecto: el valor de la propiedad 'color'

Esta propiedad puede tener de uno a cuatro valores que son aplicados a los distintos lados como en 'border-width'.

border-top-style (¥), border-right-style (¥), border-bottom-style (¥), border-left-style (¥)

Estas propiedades determinan el estilo de los bordes superior, derecho, inferior e izquierdo de una caja respectivamente.

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

El significado de los tipos de estilos es:

- ***none*** Ningún borde
- ***hidden*** Igual a 'none'
- ***dotted*** El borde es una serie de puntos
- ***dashed*** El borde es una serie de pequeños segmentos de línea
- ***solid*** El borde es un único segmento de línea
- ***double*** El borde son dos líneas sólidas
- ***groove*** El borde luce como si estuviese tallado en la página
- ***ridge*** Lo opuesto a 'grove': el borde parece que estuviera sobresaliendo de la página
- ***inset*** El borde hace que toda la caja luzca como si estuviera empotrada en la página
- ***outset*** Lo opuesto a 'inset': el borde hace que toda la caja parezca sobresalir

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

border-style (¥)

La propiedad 'border-style' es una propiedad resumida que se utiliza para definir el estilo de los cuatro bordes a la vez.

Valor:	**none** | dotted | dashed | solid | double | groove | ridge | inset | outset {1,4}

Esta propiedad puede tener de uno a cuatro valores que son aplicados a los distintos lados como en 'border-width'.

border-top (¥), border-right (¥), border-bottom (¥), border-left (¥)

Estas son propiedades resumidas para definir el ancho, estilo y color del borde superior, derecho, inferior e izquierdo de una caja.

Valor:	thin | medium | thick | <lenght> || <border-style> || <color>
Por defecto:	no definido para las propiedades resumidas

Por ejemplo:

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

La regla anterior define un borde inferior para los párrafos (un línea gruesa entera de color rojo). Los valores que no se especifican son colocados en sus valores iniciales.

border (¥)

La propiedad 'border' es una propiedad resumida para colocar el mismo ancho, color y estilo a los cuatro bordes de una caja.

Valor:	<border-width> || <border-style> || <color>
Por defecto:	no definido para las propiedades resumidas

A diferencia de las propiedades resumidas 'margin' y 'padding', la propiedad 'border' no puede definir diferentes valores para los cuatro bordes. Para eso, deben usarse una a más de las otras propiedades del borde.

FORMATO VISUAL DE CAJAS

El modelo de formato visual rige el comportamiento de las cajas generadas por los elementos de la página. Hemos visto hasta ahora algunas propiedades (bordes, márgenes, rellenos) que pueden aplicarse a las cajas y también las propiedades que definen su color/imagen de fondo. Ahora vamos a ver cómo se puede definir el tipo y dimensiones de esas cajas, su comportamiento y relación con las otras cajas en la estructura del documento.

Para comenzar debemos saber que en HTML existen tres tipos de elementos:

- ***Elementos de bloque***: Son aquellos tratados visualmente como bloques separados de los elementos que lo rodean (por ejemplo: <p> o). Para hacerlo más sencillo, podemos decir que son aquellos que comienzan una nueva línea dentro del documento. Los elementos a nivel de bloque generan una caja de bloque principal que sólo contiene otras cajas de bloque.
- ***Elementos de línea***: Son aquellos que no forman nuevos bloques de contenido; el contenido es distribuido a nivel de las líneas (por ejemplo: <b> o).
- ***Elementos de lista***: Son elementos de bloque que generan una caja principal y otras cajas adicionales (generalmente contienen una viñeta o caracteres alfanuméricos) que se agregan al costado del elemento.

display (¥)

Valor:	inline | **block** | list-item | run-in | compact | marker | table | inline-table | table-row-group | table-header-group | table-footer-group | table-row | table-column-group | table-column | table-cell | table-caption | none | inherit
Por defecto	inline

A continuación se explica el significado de algunos de estos valores.

- ***block***: Este valor provoca que un elemento genere una caja de bloque principal
- ***inline***: Este valor provoca que un elemento genere una o más cajas de línea
- ***list-item***: Este valor provoca que un elemento genere una caja de bloque principal y una caja de línea list-item (por detalles ver la sección sobre listas)
- marker: Este valor declara que el contenido generado antes o después de una caja será un marcador (por detalles ver la sección sobre marcadores)
- ***none***: Este valor provoca que un elemento no genere ninguna caja (es decir, el elemento no tiene ningún efecto sobre la composición) y los elementos descendientes tampoco generan cajas. Hay que destacar que este valor no crea una caja invisible sino que hace que el elemento desaparezca por completo. Esa es la diferencia con las propiedades sobre visibilidad que provocan que un elemento pueda ser invisible pero siga ocupando un espacio en la página
- ***run-in y compact***: Estos valores crean cajas de bloque o de línea según el contexto y tienen un comportamiento como el que conocemos para las listas de definiciones
- ***table, inline-table, table-row-group, table-column, table-column-group, table-header-group, table-footer-group, table-row, table-cell y table-caption***: Estos valores provocan que un elemento se comporte como un elemento tabla (por detalles ver la sección sobre tablas)

En este punto es necesario hacer algunas aclaraciones. Teóricamente, estos valores nos permiten alterar el comportamiento visual de los elementos de la página, por ejemplo, haciendo que un elemento a nivel de línea en HTML (supongamos <CODE>) se transforme en un elemento de bloque, de modo que cualquier texto marcado con <CODE></CODE> comenzaría en una nueva línea. De igual modo, podríamos hacer que cualquier elemento se comporte como una tabla o como parte de una tabla. Con HTML y XHTML no tiene mucho sentido ya que existen elementos

como las tablas, columnas y celdas, pero otros lenguajes (XML, por ejemplo) no tienen dichos elementos.

position (¥)

Valores:	static | relative | absolute | fixed | **inherit**
Por defecto:	static
Se aplica a:	todos los elementos, no al contenido generado

Con CSS los autores pueden ubicar los elementos de la página usando uno de estos valores:

- ***static***: Esta es la posición predeterminada y la que conocemos en HTML. La caja se sitúa dentro del flujo normal de la página y las propiedades 'top', 'right', 'bottom' y 'left' no se aplican.
- ***relative***: La posición de la caja se ajusta en relación a su posición normal dentro de la página. Cuando una caja X se posiciona relativamente, la caja siguiente se sitúa como si X no se hubiera desplazado.
- ***absolute***: Las cajas son quitadas del flujo normal de la página y su posición se especifica con las propiedades 'left', 'right', 'top', y 'bottom'. Estas propiedades especifican los desplazamientos con respecto al bloque de contención de la caja por lo que los elementos posicionados absolutamente no tienen ninguna influencia sobre la posición de las cajas siguientes. El bloque de contención para una caja posicionada es establecido por el antepasado posicionado más cercano o, si no existe, por el bloque de contención inicial (la esquina superior izquierda de la página, en el modelo visual).
- ***fixed***: El posicionamiento fijo es una subcategoría del posicionamiento absoluto. La única diferencia es que para una caja posicionada de modo fijo, el bloque de contención es establecido por el acceso visual (la pantalla del monitor) y el elemento no se mueve cuando se realiza un desplazamiento. Esto significa que cuando se hace un scroll en la página los elementos con position: fixed mantienen su posición en la pantalla.

top (¥), bottom (¥)

Estas propiedades especifican la distancia que se desplaza el elemento por debajo del borde superior del bloque de contención de la caja y por sobre de borde inferior del bloque de contención de la caja respectivamente.

Valor:	<longitud> | <porcentaje> | **auto** | inherit
Se aplica a:	los elementos posicionados
Valor por porcentaje:	Se calcula con respecto a la altura del bloque de contención. Si la altura del bloque de contención no está especificada explícitamente (es decir, depende de la altura del contenido), el valor del porcentaje es interpretado como 'auto'.

right (¥), left (¥)

Estas propiedades especifican la distancia que se desplaza el elemento hacia la izquierda del borde derecho del bloque de contención de la caja y hacia la derecha del borde izquierdo del bloque de contención de la caja respectivamente.

Valor:	<longitud> | <porcentaje> | **auto** | inherit
Se aplica a:	los elementos posicionados
Valor por porcentaje:	se calcula con respecto al ancho del bloque de contención

Para las cajas con `position:absolute`, el desplazamiento es con respecto al bloque de contención de la caja. Para las cajas con `position:relative`, el desplazamiento es con respecto al borde externo de la propia caja (es decir, la caja de desplaza de su posición normal dentro de la página de acuerdo a estas propiedades).

z-index (¥)

En algunos casos, ciertos elementos se superpongan visualmente con otros; en esas situaciones, este orden dentro de la pila de elementos puede adjudicarse explícitamente con la propiedad `z-`

`index` utilizando un valor entero (cuanto más alto el entero, más cercano al lector o más arriba en la pila).

Valor: **auto** | <numero> | inherit
Se aplica a: elementos con propiedad position definida

Los valores tienen los siguientes significados:

- ***<numero>*** Este entero es el nivel de pila de la caja. La caja también establece un contexto de pila local en el cual su nivel de pila es '0'.
- ***auto*** El nivel de pila de la caja generada es el mismo que el de la caja de su padre. La caja no establece un nuevo contexto de pila local.

Todos los descendientes del elemento ubicado más al frente se ubiquen por sobre todos los descendientes del elemento ubicado más atrás. Si en una página encontramos dos elementos (A y B), uno con z-index=5 (A) y el otro con z-index=3 (B), A será procesado al frente de B. Supongamos ahora que A tiene un descendiente (C) con z-index=0: el elemento C, por pertenecer al contexto de pila del elemento ubicado más arriba, también será procesado por encima de B (aunque B tenga z-index=3). Todos los elementos descendientes de A se ubicarán por encima de B (y también por encima de todos los descendientes de B).

float (¥)

Valor: left | right | **none**

Un flotante es una caja que es desplazada a la izquierda o a la derecha en la línea actual. Esta propiedad especifica si una caja debe flotar a la izquierda, derecha o no debe flotar en absoluto. Los valores de esta propiedad tienen los siguientes significados:

- ***left*** El elemento genera una caja de bloque que flota a la izquierda. El contenido fluye sobre el costado derecho de la caja, comenzando en la parte superior.
- ***right*** Igual que 'left', pero el contenido fluye sobre el costado izquierdo de la caja, comenzando en la parte superior.
- ***none*** La caja no es flotante.

clear (¥)

Valor: **none** | left | right | both

Esta propiedad indica cuál de los lados de la(s) caja(s) de un elemento no puede quedar adyacente a una caja flotante anterior. Esta propiedad sólo puede especificarse para elementos a nivel de bloque (incluyendo también a los elementos flotantes). Los valores tienen los siguientes significados:

- ***left*** El margen superior de la caja generada se aumenta lo suficiente para que su borde superior quede debajo del borde inferior de cualquier caja flotante a la izquierda que aparezca antes en el documento fuente.
- ***right*** El margen superior de la caja generada se aumenta lo suficiente para que su borde superior quede debajo del borde inferior de cualquier caja flotante a la derecha que aparezca antes en el documento fuente.
- ***both*** La caja generada se mueve debajo de todas las cajas flotantes que aparecen antes en el documento fuente.
- ***none*** No existe ninguna restricción a la posición de la caja con respecto a los flotantes.

visibility (¥)

Esta propiedad especifica si las cajas generadas por un elemento son procesadas. Aunque el elemento sea invisible sigue afectando la composición de la página, es decir, continúa ocupando su espacio.

Valor: visible | hidden | collapse | **inherit**

width (¥)

Esta propiedad especifica el ancho del contenido de las cajas generadas por elementos a nivel de bloque y elementos reemplazados (por ejemplo, IMG).

Valor: <longitud> | <porcentaje> | **auto**
Se aplica a: de bloque and elementos remplazados
Valor por porcentaje: con relación al ancho de los elementos del padre

- ***<longitud>*** Especifica un ancho fijo.
- ***<porcentaje>*** Especifica el ancho según un porcentaje. El porcentaje es calculado con respecto al ancho del bloque de contención de la caja.
- ***auto*** El ancho depende de los valores de otras propiedades. Las reglas para determinar este valor pueden consultarse en la sección [Computando anchos y márgenes] de la Especificación CSS2.

height (¥)

Esta propiedad especifica la altura del contenido de las cajas generadas por elementos a nivel de bloque y elementos reemplazados (por ejemplo, IMG).

Valor: <longitud> | <porcentaje> | **auto** | inherit
Se aplica a: de bloque y elementos remplazados

- ***<longitud>*** Especifica una altura fija.
- ***<porcentaje>*** Especifica la altura según un porcentaje. El porcentaje es calculado con respecto a la altura del bloque de contención de la caja. Si la altura del bloque de contención no se especifica explícitamente (es decir, depende de la altura del contenido) el valor es interpretado como 'auto'.
- ***auto*** La altura depende de los valores de otras propiedades. Las reglas para determinar este valor pueden consultarse en la sección [Computando alturas y márgenes] de la Especificación CSS2.

overflow (¥)

Esta propiedad especifica si el contenido de un elemento a nivel de bloque es recortado cuando desborda la caja del elemento. Generalmente el contenido de una caja de bloque se mantiene dentro de los límites de la caja, pero puede suceder que el contenido desborde esos límites y quede parcial o totalmente fuera de la caja.

Esto puede suceder si especificamos un tamaño determinado para un elemento (con las propiedades 'width' y 'height', por ejemplo) y su contenido resulta demasiado grande para las medidas adjudicadas. También puede darse al utilizar márgenes negativos o posiciones absolutas para el elemento.

Cuando se produce un desbordamiento, la propiedad 'overflow' especifica si la caja es recortada y (en caso afirmativo) cómo será recortada. La propiedad 'clip' especifica el tamaño y la forma de la zona de recorte.

Valor: **visible** | hidden | scroll | auto | inherit
Se aplica a: de bloque and elementos remplazados

Los valores tienen los siguientes significados:

- ***visible*** Este valor indica que el contenido no es recortado, es decir, puede ser procesado fuera de la caja de bloque.
- ***hidden*** Este valor indica que el contenido es recortado y los usuarios no tendrán acceso al contenido recortado. El tamaño y forma de la zona de recorte son especificados por la propiedad 'clip'.
- ***scroll*** Este valor indica que el contenido es recortado y el navegador debe proporcionar un mecanismo de desplazamiento que permanecerá siempre visible (aunque la caja no tenga parte de su contenido recortado).
- ***auto*** El comportamiento del valor 'auto' depende del navegador, pero significa

clip (¥)

La propiedad 'clip' se aplica a elementos que tienen una propiedad 'overflow' con un valor diferente a 'visible'.

La propiedad clip define una zona de recorte, es decir, qué porción del contenido de un elemento es visible. De manera predeterminada, la zona de recorte tiene el mismo tamaño y forma que la caja del elemento. Sin embargo, la zona de recorte puede ser modificada por la propiedad 'clip'.

Valores:	<forma> | auto | inherit
Valor inicial:	auto
Se aplica a:	los elementos a nivel de bloque y reemplazados
<forma>	rect (<arriba>,<derecha>,<abajo>,<izquierda>)
<arriba>	<longitud> | <auto>
<derecha>	<longitud> | <auto>
<abajo>	<longitud> | <auto>
<izquierda>	<longitud> | <auto>

Los valores tienen los siguientes significados:

- ***auto*** La zona de recorte tiene el mismo tamaño y ubicación que la(s) caja(s) del elemento.
- ***<forma>*** En CSS2, el único valor permitido es: rect (<arriba>, <derecha>, <abajo>, <izquierda>). Los valores para <arriba><derecha><abajo> e <izquierda>pueden ser una <medida> o 'auto'. Las medidas negativas están permitidas y 'auto' significa lo mismo que '0' (cero).

2) Enlaces Interesantes

Especificidad http://www.w3.org/TR/CSS21/cascade.html#specificity

Lista de todas las propiedades de CSS 2.1 http://www.w3.org/TR/CSS21/propidx.html

Manual sencillo y completo de CSS (casi todo el manual anterior está basado en este) http://www.sidar.org/recur/desdi/mcss/manual/indice.php

Wikipedia CSS http://es.wikipedia.org/wiki/Hojas_de_estilo_en_cascada

Manual de la w3School para CSS http://www.w3schools.com/css/default.asp

Tutorial de selectores de CSS http://css.maxdesign.com.au/selectutorial/index.htm

Tutorial para manejo de objetos flotantes http://css.maxdesign.com.au/floatutorial/index.htm

Tutoriales para manejo de listas con CSS

http://css.maxdesign.com.au/listamatic/index.htm

http://css.maxdesign.com.au/listamatic2/index.htm

http://css.maxdesign.com.au/listutorial/index.htm

Ejemplos de diagramación con base a 3 columnas http://css-discuss.incutio.com/?page=ThreeColumnLayouts

El proyecto Camaleón http://www.camaleoncss.com/1/p1/camaleon.html