Manual CSS Básico

- Introducción
- Sintaxis CSS
  - Definición en la propia etiqueta html
  - Novedades
- Colocando las hojas de estilo
  - Dentro del documento HTML
  - En un fichero .CSS
- Propagación de estilos
  - Herencia de estilos
  - Estilos en función del contexto
  - El concepto de cascada
- Definición de los estilos
  - Mediante clases
  - Mediante ID
  - Diferencias entre clases e ID
- Pseudoclases y pseudoelementos
  - Pseudoclases
  - Pseudoelementos
- Propiedades de las hojas de estilo
- Propiedades de texto
  - Declarando longitudes
  - Ejemplos
- Propiedades de las fuentes
  - Ejemplos
- Propiedades de los colores y fondos
  - Sobre la utilización del color
  - Ejemplos
- Propiedades de margenes y padding
  - Ejemplos
- Propiedades de los bordes
  -

1. Introducción

Con el HTML se intento desde un principio la definición de estilos lógicos que se centrasen más en el contenido de la información que en su presentación. Sin embargo el gran exito de Internet y por tanto la aparición de importantes intereses comerciales exigían un control cada vez más rígido sobre la presentación final del documento.

Esto motivo una **evolución del HTML centrada en mejorar su presentación.** Aunque las intenciones han sido buenas, los efectos secundarios han sido ha menudo desafortanudos y hemos llegado a un HTML demasido complejo para sus objetivos iniciales y en muchos casos **incompatible** entre los principales navegadores:

- Uso de **extensiones propietarias del HTML** (soluciones propias de Netscape y de Microsoft)
- **Conversión de texto en imágenes** de texto para hacerle mantener el mismo aspecto en cualquier navegador.
- Uso de **imágenes** (**transparentes**, de tamaños concretos, ...) para controlar los espacios en blanco.
- **Uso de tablas para controlar el diseño global** de la página.
- **Uso de programas en lugar del marcado HTML**...

Estas técnicas aumentan considerablemente la complejidad de las páginas Web, tienen escasa flexibilidad y problemas de interoperabilidad.

Debido a estas causas y con la intención de volver a la idea original de separar el contenido de la presentación y de ofrecer mayores y más faciles posibilidades de presentación el W3 Consortium, **empezo a discutir a principios de 1995 la utilización de las Hojas de Estilo (CSS) en la Web.** La intención era incluirlas en las especificaciones 3.0 del HTML pero las discusiones se extendieron tanto que al final se impusieron las extensiones propuestas por Netscape dando lugar a la versión 3.2 del HTML, aunque la versión 3.0 del Explorer ya las soportaba en parte.

En la actualidad las especificaciones CSS se encuentran en el nivel 2 (CSS2), que es una recomendación del W3C del 12 de Mayo de 1998, y se están escribiendo actualmente las especificaciones del nivel 3 (CSS3). Sin embargo, **en este curso vamos a estudiar la recomendación de nivel 1** (CSS1), que es la que está implementada por los navegadores actuales (Explorer 5 y superiores, Mozilla y Opera) y en parte por las antiguas versiones 4 tanto del Explorer como del Netscape.

2. Sintaxis CSS

En el momento de definir los estilos debemos tener en cuenta que estos se pueden declarar directamente sobre la etiqueta HTML o que podemos definirlos en su conjunto para toda la página HTML.

Definición en la propia etiqueta html

La sintaxis básica para especificar un estilo en una etiqueta determinada es:

```
<etiqueta STYLE="propiedad:valor;....">... </etiqueta>
```

Donde `etiqueta` representa un TAG estandar del HTML y la palabra STYLE es el atributo que indica que a dicha etiqueta le vamos a asociar un estilo. El resto ya es la definición propia del estilo que viene definido por un par `propiedad:valor` separados entre si por dos puntos y del resto de pares por un punto y coma.

En `propiedad` indicamos que caracteristica (tipo de fuente, color, etc...) queremos cambiar y en valor el `valor` que le damos.

Por ejemplo si a un parrafo le queremos dar un tamaño de fuente 10 y un margen izquierdo de 20 pts.

```
<P STYLE="font-size:10pt; margin-left:20pt;">
    Mi Primer parrafo con Estilo
</P>
```

Pero la asignación individual a cada etiqueta de los estilos puede resultar un poco pesada también contamos con la posibilidad de definirlos de forma global para toda la página.

Dentro del documento HTML

Cuando hacemos una defincón en grupo de una hoja de estilos utilizando la etiqueta <STYLE> esta tiene que estar colocada en la cabecera del documento HTML es decir dentro de la etiqueta <HEAD>.

```
<HTML>
<HEAD>
<TITLE>Esta es mi primera hoja de estilos</TITLE>
  <STYLE TYPE="text/css">
  <!-
        H1 {color:blue;}
        P {font-size:10pt; marginleft:20pt;}
  -->
</STYLE>
</HEAD>
<BODY>
<H1>Texto Azul</H1>
<P>Fuente de tamaño 10 ptos. y margen izquierdo 20</P>
</BODY>
</HTML>
```

En un fichero .CSS

Pero también podemos definirlas en un fichero externo y luego referenciarlas desde el código HTML. Esta es sin duda alguna una de sus mayores virtudes, ya que nos permite crear una misma hoja de estilos para toda la Web que luego llamamos desde cada una de las páginas. De esta manera, cambiando el fichero que contiene la hoja de estilos podemos cambiar el aspecto de toda la web.

Existen dos maneras de enlazar las hojas de estilo.

Haciendo uso de la etiqueta `<LINK>`

```
<HTML>
<HEAD>
   <LINK REL="stylesheet" TYPE="text/css" HREF="hoja_estilos.css">
</HEAD>
......
</HTML>
```

O utilizando la etiqueta `@IMPORT`

```
<HTML>
<HEAD>
  <STYLE TYPE="text/css">
     @IMPORT URL("hoja_estilo.css")
 </STYLE>
</HEAD>
......
</HTML>
```

El fichero hoja_estilos.css es un simple fichero de texto con extensión .css en el que definimos la hoja de estilo en función de la sintaxis que hemos descrito anteriormente.

```
/*Fichero hoja_estilos.css*/
        <!-
        H1 {color:blue;}
        P {font-size:10pt; marginleft:20pt;}
  -->
/*Fin del fichero hoja_estilo.css*/
```

3. Propagación de estilos

Herencia de estilos

Las **etiquetas** de un documento **HTML** estan organizadas de manera que **unas engoblan a otras**. Por ejemplo todas las etiquetas se encuentran dentro del TAG <BODY> y otras como <EM> la podemos encontrar entre <P>, <H1>, <UL>, etc...

Esta organización permite una relación **padre-hijo** de manera que los estilos definidos para etiquetas padres seran heredados por los hijos.

Por ejemplo si definimos:

```
<STYLE TYPE="text/css">
  <!-
   BODY {font-family:Arial; color:blue;}
  -->
</STYLE>
```

en principio todo el documento estara con una fuente Arial de color azul.

Esta caracteristica puede **resultar muy útil** ya que **evita tener que volver a definir los mismos estilos** para diferentes etiquetas. Es decir con el ejemplo anterior nos evitamos (si nos interesa) tener que volver a definir en la etiqueta <P> la fuente y su color.

El mecanismo de herencia de estilos siempre funciona a menos que un elemento hijo tenga definido su propio estilo, el cual prevalece sobre el heredado e incluso en algunos casos se complementan.

En el siguiente caso:

```
<HTML>
<HEAD>
<TITLE>Esta es mi primera hoja de estilos</TITLE>
  <STYLE TYPE="text/css">
  <!-
    P {color:blue; font-size:12pt;}
    EM {color:red; text-transform:uppercase;}
  -->
</STYLE>
</HEAD>
<BODY>
<P>Este es un parrafo donde <B> la etiqueta <B> </B>hereda
 el estilo de <P></P>
<P>En cambio <EM> en la etiqueta <EM></EM> al estar definido
 prevalece su propio estilo<</P>
</BODY>
</HTML>
```

Observamos como la etiqueta <B> utilizada para colocar negrita si que remarca el texto pero lo deja en azul ya que hereda ese valor de <P>. En cambio <EM> aparece en mayusculas y verde a pesar de estar dentro de <P> ya que le hemos definido su propio estilo.

Otro caso es por ejemplo el de esta página. En la etiqueta <BODY> se ha definido los margenes que tendra la página y ya no hemos tenido que volver a definirlos ya que el resto de etiquetas lo heredan.

Otro aspecto importante es que esta **herencia supone algunas veces una complementación** entre los

estilos definidos. Por ejemplo como hemos dicho anteriormente en la etiqueta <BODY> hemos definido unos determinados margenes. Luego he definido un estilo para diferenciar el código del resto de los parrafos y he vuelto a definir los margenes pero teniendo en cuenta la cantidad que habia puesto anteriormente.

Aunque hay que tener en cuenta que **no todas las propiedades se heredan** y que es conveniente tener las especificaciones a mano para ver cuales se heredan y cuales no.

Estilos en función del contexto

Otra característica importante es que podemos dar estilos a un elemento dependiendo de donde se encuentre colocado en la página, es decir a una etiqueta no tiene porque corresponderle siempre el mismo **estilo** sino que podemos hacer que este **dependa de la situación en la que se encuentre**.

En el siguiente ejemplo la etiqueta <B> aparece marca texto en rojo en una situación y en azul en otro. Azul cuando se encuentra dentro de un parrafo normal y rojo cuando esta en una lista.

```
<HTML>
<HEAD>
<TITLE>Esta es mi primera hoja de estilos</TITLE>
  <STYLE TYPE="text/css">
  <!-
    P B{color:blue}
    LI B {color:red}
  -->
  </STYLE>
</HEAD>
<BODY>
<P>Aqui aparece <B> azul </B> ....</P>
<P>En cambio en la siguiente lista aparece...
<UL>
<LI>Aparece <B>rojo</B> ....</LI>
</UL>
</BODY>
</HTML>
```

El concepto de cascada

A traves del concepto de cascada **se establecen las reglas que determinan la forma en que una definición de estilo puede sustituir a otra**.

Como ya hemos visto los estilos pueden definirse en varios sitios:

- En un fichero externo .css que luego referenciamos a traves de la etiqueta <LINK> o @IMPORT.
- A traves de un bloque de estilos definidos en la etiqueta <STYLE>.
- O directamente sobre la etiqueta HTML.

La utilización conjunta de estos métodos no es incompatible, sino que en muchos casos necesaria y recomendable. Es decir en una página HTML nos puede interesar utilizar la .css global que utilizamos para toda la Web, y luego otras particulares que definimos para esta página. Esto va a dar lugar a que muchas definiciones de estilo se solapen y entonces la reglas de cascada nos garantizan que manda la última especificación de estilo definida siguiendo el orden jerarquico definido en la lista anterior.

4. Definición de los estilos

Hasta el momento siempre que hemos definido los estilos estos han ido asociado a una etiqueta HTML.

El problema es que una misma etiqueta tendra que presentar el mismo aspecto en toda la página y esto evidentemente no nos interesa. Hemos visto que utilizando las reglas que determinan la cascada se puede evitar en parte este problema , pero **mediante el uso de clases** y de la **definición de estilos con ID** lo solucionaremos completamente.

A traves de ellos **asociaremos a un identificador un estilo de manera que luego podamos utilizar este identificador con la etiqueta HTML que nos interese.**

Mediante clases

Definiendo los estilos mediante la utilización de clases creamos un estilo, un formato que en principio no esta asociado a ninguna etiqueta HTML y que nosotros podemos asociar con la que queramos a traves del atributo CLASS.

Para definir una clase que identifica un estilo se hace como hasta ahora, lo que cambiamos el nombre de la etiqueta por el nombre de la clase.

```
.NombreClase {color: blue}
```

que luego asociaremos a la etiqueta HTML con el atributo CLASS.

```
<p class="NombreClase">Este parrafo ira en azul.</p>
<h2 class="NombreClase">Y este tambien</p>
```

pudiendo ser las etiquetas diferentes.

Conviene tener bien clara la utilización de las clases ya que posiblemente sea la forma más habitual de definir los estilos dada la libertad de acción que nos permiten al no obligarnos a presentar una misma etiqueta con el mismo aspecto en toda la página.

Mediante ID

Los ID funcionan de forma similiar a las clases pero están limitados a su utilización con un sólo elemento. Los ID se definen utilizando el signo "#" seguido por el mombre del identificador y luego las propiedades del estilo.

```
#NombreID {color: blue}
```

que luego asociamos a la etiqueta HTML mediante el atributo ID.

```
<p id="NombreID">Este parrafo ira en azul</p>
```

Lo importante de definir estilos utilizando ID es que de esa manera identificamos de forma univoca un elemento de la página HTML.

Diferencias entre clases e ID

La principal diferencia entre utilizar un identificador o una clase para definir un estilo es que **mediante un ID estamos identificando algún elemento de la página de forma univoca y por tanto sólo lo podemos utilizarlo con ese elemento**. Esto se suele hacer porque luego posiblemente querremos realizar alguna acción sobre ese elemento. Mientras que con una **clase estamos definiendo un estilo genérico que luego podremos utilizar sobre cualquier elemento del HTML**.

5. Pseudoclases y pseudoelementos

En algunos casos el **HTML ya proporciona a ciertas etiquetas un estilo especial propio**. El hecho de que los enlaces se marquen y se subrayen es un ejemplo de estilo que ya existe incluso antes de ser especificado en nuestra hoja de estilo. **A estos elementos los denominaremos pseudoclases**.

Por **pseudoelementos nos referiremos** a algunos elementos del documento que **no** estan reconocidos como parte del **HTML estándar** pero que **el navegador va a ser capaz de identificar** y por lo tanto proporcionarles un estilo determinado. Son, por ejemplo, **la primera letra o la primera linea de un bloque de texto**.

Pseudoclases

Por el momento sólo podemos definir pseudoclases para los enlaces, es decir sólo se utiliza con la etiqueta A.

La forma de utilizarlas es la siguiente:

```
etiqueta:Pseudoclase {definición de estilo}
```

Se dispone de las siguientes pseudoclases:

- **:link**, nos indica el estilo con el que debe aparecer el **enlace antes de ser visitado**.
- **:visited**, nos indica el estilo con el que aparece **un enlace visitado**.
- **:active**, nos indica el estilo con el que aparece el **enlace en el momento de activarlo**, es decir de pulsar sobre él.
- **:hover**, nos indica el estilo en el que aparece el **enlace al pasar por encima el ratón**.

En el siguiente ejemplo, el enlace aparece en primer lugar de color verde. Cuando pasamos por encima el cursor se vuelve de color amarillo. Cuando lo pulsamos se vuelve gris. Y si se trata de un enlace ya visitado se vuelve azul.

```
<HTML>
<HEAD>
<TITLE>Esta con pseudoclases</TITLE>
  <STYLE TYPE="text/css">
  <!-
        A:link {COLOR: green}
        A:visited {COLOR: blue}
        A:active {COLOR: gray}
        A:hover {COLOR: yellow}
  -->
</STYLE>
</HEAD>
<BODY>
<P>Un <a href="http://www.programacion.net/html">enlace</a>
       de prueba.</P>
</BODY>
</HTML>
```

Las pseudoclases se pueden combinar con las clases. Es decir podemos aplicar estas pseudoclases sólo a la clase de enlace que nos interese.

En el siguiente ejemplo sólo las aplicaremos a los enlaces definidos con la clase external.

```
A.external:link {color:blue}
A.external:visited {color:purple}
```

```
<A class="external" href="ejem.htm">enlace</A>
```

Y también **se pueden utilizar en función del contexto**.

```
A:link IMG {border-color:blue}
```

Pseudoelementos

Los pseudolementos se utilizan por el momento para crear efectos de texto tales como letras capitales o destacar la primera linea de ciertos bloques de texto.

La forma de utilizarlas es la siguiente:

```
etiqueta:Pseudoelemento {definición de estilo}
```

Por el momento disponeros de los siguientes pseudoelementos:

- **:first-line**, selecciona y **aplica estilo a la primera linea** de un bloque de texto. La cantidad de texto que compone la primera línea depende del tamaño y familia de la fuente, tamaño de la ventana, tamaño del bloque, existencia de objetos flotantes, etc... Sólo se pueden aplicar algunas características de estilo.
- **:first-letter**, selecciona **la primera letra** del texto de un bloque de texto para aplicarle determinados efectos tipográficos. Sólo se pueden aplicar algunas características de estilo.

En el siguiente ejemplo la primera línea de un parrafo debe mostrarse en mayusculas.

```
P:first-line {text-transform:uppercase}
```

En el siguiente ejemplo los párrafos del documento van en negro y con un tamaño de fuente de 12 puntos, en cambio la primera letra del párrfo debe ser verde y de doble tamaño.

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

Anexos: Propiedades de las hojas de estilo

Ya hemos visto que la defínción de estilos consiste en la sucesión de pares **propiedad:valor** separados por ; en los que indicamos que característica del texto que va entre la etiqueta HTML queremos controlar.

Todas estas **propiedades** estan **divididas** en **5 grandes grupos** que facilitan su utilización y documentación:

- Propiedades de texto
- Propiedades de las fuentes
- Propiedades de los colores y fondos
- Propiedades de margenes y padding
- Propiedades de los bordes

En las proximas paginas encontrareis una lista de estas propiedades, y páginas de ejemplos de como utilizarlas.

Para unificar el aspecto de las páginas de ejemplos he creado una hoja de estilos **"estilo1.css"** muy sencilla que ya teneis que empezar a entender.

```
/*Fichero hoja de estilos. Estilos1.css*/

 body { background-color: #000000;
       font-family: Verdana, Arial, Helvetica, sans-serif;
       font-size: 10pt;
       color: #FFFFFF;
       margin-right: 100px;
       margin-left: 100px}

 .titulo { font-size: 36pt;
      font-weight: bolder;
      color: #FFCC00;
      text-align: left;
      margin-left: -50px}

  /*Fin del fichero */
```

En la que como podeis observar definimos el color del fondo, el tipo y tamaño de letra que aparecera en la página, los margenes y definimos una clase título, que utilizaremos para presentar el titulo. Luego desde cada página la referenciamos de la siguiente manera:

```
<link rel="stylesheet" href="estilos1.css">
```

A1. Propiedades de texto

Son propiedades que afectan a la presentación visual de caracteres, espacios, palabras y parrafos.

Las propiedades que podemos utilizar son las siguientes:

text-transform

> Especifica, por medio de palabras reservadas, **si las letras del texto deben transformarse en mayúsculas, minúsuculas**, sólo la primera letra de cada palabra en mayúsculas, o si ha de dejarse como está. **Se hereda**.
>
> `uppercase | lowercase | capitalize |` **`none`**

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

En el ejemplo anterior todas las cabeceras de nivel 1 apareceran en mayúsculas.

vertical-align

Alineación vertical del texto en relación con la línea base del texto. **No se hereda**.

```
baseline | sub | super | top | text-top | middle | mottom |
text-bottom | %
```

```
STRONG {vertical-align: sub}
```

text-align

Fija la **alineación del bloque**, al margen izquierdo, al derecho, centrado o a ambos. **Se hereda**.

```
left | right | center | justify
```

```
H1 {text-align: center}
```

En este caso todas las cabeceras de nivel 1 apareceran centradas.

text-indent

Fija la sangría o indentación de la primera línea del texto. **Se hereda**.

```
XX unidad | %, (Por defecto su valor es 0)
```

```
P {text-indent: 2em}
```

line-height

Indica la **distancia entre dos lineas** adyacentes. **Se hereda**.

```
normal | XX unidad | %
```

text-decoration

Fija una o más **características "decorativas" del texto**, como subrayado, lineas superiores, caracteres tachados o parpadeantes.

Si se aplica a un elemento de bloque la heredan sólo los descendientes que sean de texto (o en-linea). **Si se aplica a uno de éstos la heredan todos los descendientes**.

```
underline | overline | line-trough | blink | none
```

```
H1 {text-decoration: underline}
```

En el ejemplo anterior todas las cabeceras de nivel 1 apareceran subrayadas.

letter-spacing

Especifica el **espaciado entre letras**. En el caso de especificar una longitud esta se sumara a la normal. **Se hereda**.

```
normal | XX unidad
```

```
H1 { letter-spacing: 0.5pc}
```

word-spacing

Especifican el **espaciado entre palabras**. En el caso de especificar una longitud esta se sumara a la normal. **Se hereda**.

```
normal | XX unidad

H1 { word-spacing: 0.5pc}
```

Declarando longitudes

En algunas de las propiedades tenemos que definir una longitud. En la sintaxis CSS esta **se define con un signo opcional, un número y un identificador de unidad**. Estos tres elementos **van juntos y sin espacios entre ellos**.

En la siguiente tabla se muestran los diferentes tipos de medidas que admiten las CSS:

Abreviatura	Unidad	Equivalencia
in	pulgadas	2.54 cm
cm	centimétros	
mm	milimetros	
pt	puntos	1/72 pulgadas
pc	picas	12 puntos
em	anchura-m	
ex	altura-x	
px	píxeles	

Ademas al poner una longitud se puede especificar de forma **absoluta** o de forma relativa mediante la **unidad em o utilizando porcentajes**.

A2. Propiedades de las fuentes

Con estas propiedades controlamos el estilo de una fuente, su tamaño, su familia, su grosor, etc.

Las propiedades que podemos utilizar son las siguientes:

font-family

Indicamos la familia de la fuente a utilizar. Sus posibles valores son los nombres de las fuentes, pudiendo especificar un conjunto de familias. Se selecciona la primera que se encuentra en el sistema. **Se hereda**.

```
nombre de las familias de las fuentes | familia genérica

BODY { font-family: Verdana, Arial, Helvetica, sans-serif}
```

En el ejemplo anterior estamos diciendo que todo el contenido de la página este en Verdana, y que en caso de que no este instalada utilice Arial y si no Helvetica y si no el tipo de familia genérica sans-serif.

font-style

Especifica el estilo de la fuente, es decir, si los caracteres iran en normal, en itálica (cursiva) o en oblicua (un poco más inclinada que la itálica). **Se hereda**.

```
normal | italic | oblique

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

En este ejemplo estamos indicando que las cabeceras de hasta nivel 3 iran en itálica. Pero que el texto emfatizado de las cabeceras de nivel 1 iran en normal.

font-variant

Especifica la variante de la fuente, es decir, si los caracteres serán tomados de la **fuente normal o de small-caps** (o versalitas, donde las minúsculas son parecidas a las mayusculas pero de menor tamaño) dentro de una familia de fuentes. **Se hereda**.

```
normal | small-caps
```

font-weight

Especifica el peso, **densidad o grosor** (cantidad de espacio ocupado en relación con el rectángulo en que se inscribe el carácter) de la fuente de caracteres dentro de una família de fuentes, con **valores entre 100 y 900 siendo cada vez más densos.** El valor 400 equivale a normal, y el 700 a bold.

El **valor bolder** indica que los **caracteres deben ser más densos que el valor heredado** excepto en el que caso de que ya valga 900.

El **valor lighter** funciona a la inversa de bolder **indicando que tiene que ser de menos densidad** excepto en el caso de que ya valga 100.

Esta propiedad se hereda.

```
lighter | normal | bold | bolder | 100 | 200 | 300 | ..... |
900

STRONG {font-weight: bolder}
```

font-size

Especifica el tamaño de la fuente. Se hereda.

Puede ponerse como una longitud en unidades (absoluta o relativa), aunque también admite palabras reservadas que **indican tamaños absolutos predefinidos (de menor a mayor xx-small, x-small, small, medium, large, x-large, xx-large)**.

Los valores **larger y smaller especifican un tamaño, en relación con los anteriores,** respectivamente un grado mayor o menor que el heredado.

Para facilitar el funcionamiento en cascada, es conveniente **utilizar longitudes relativas (como em) o porcentajes**.

```
XX units | % | larger | smaller | xx-small | x-small | medium |
large | x-large | xx-large
```

A3. Propiedades de los colores y fondos

Son las propiedades que permiten fijar el color del primer plano y fondo de un elemento.

Las propiedades que podemos utilizar son las siguientes:

color

Especificamos el color del primer plano del contenido del texto. **Se hereda**.

```
Nombre Color | Valor HEX | Rgb (R%,g%,B%) | Rgb(R, G,B)

EM {color: red}
```

En el anterior ejemplo cualquier texto enfatizado de la página aparecera en rojo.

background-color

Especifica el color del fondo del elemento, que puede ser un color o la palabra reservada **transparent** (el fondo del elemento padre se ve).

En general, las propiedades del fondo **no se heredan**, **pero al ser su valor inicial transparent, el fondo del elemento padre se ve en los hijos.**

Esta propiedad **no se hereda.**

```
transparent | Nombre Color | Valor HEX | Rgb (R%,g%,B%) | Rgb(R, G,B)

H1 {background-color: #0000FF}
```

En el anterior ejemplo todos los encabezados de nivel 1 aparaceran con un fondo azul.

background-image

Designa una imagen para rellenar el fondo del elemento por medio de una URL o la palabra reservada none para indicar que no se utilizara ninguna imagen.

Puede fijarse también un color que se colocará debajo de la imagen, y que asomará si la imagen no está disponible o, si lo está, detrás de sus zonas transparentes.

Esta propiedad **no se hereda**.

```
none | url(dirección)

H1 {background-image: white url("fondo.gif")}
```

En el ejemplo anterior cualquier encabezado de nivel 1 tendra como fondo la imagen "fondo.gif" y si esa imagen no esta disponible el fondo aparecera en color blanco.

background-repeat

Establece si la imagen de fondo (establecida con background-image) **se repetirá** horizontal y verticalmente para rellenar todo el fondo del elemento (**repeat)**, sólo horizontalmente (**repeat-x**), sólo verticalmente (**repeat-y**), o no se repetirá (**no-repeat**). **No se hereda**.

```
repeat | repeat-x | repeat-y | no-repeat

H1 {background-image: white url("fondo.gif")}
```

En este ejemplo al no especificarse la propiedad `background-repeat` por defecto se toma el valor **repeat** por lo que la imagen "fondo.gif" se repetira tanto horizontal como verticalmente tantas veces como sea necesario hasta ocupar toda la zona determinada por el encabezado.

background-attachment

Establece si la imagen debe desplazarse (**scroll**) **o permanecer fija** con respecto a la ventana (**fixed**) cuando el usuario hace scroll. En cualquier caso sólo asoma en el área de contenido y relleno del elemento). **No se hereda**.

```
scroll | fixed
```

background-position

Fija la posición de la imagen en el área de relleno del elemento por medio de:

Coordenadas, expresadas como **longitud** (distancia hasta la esquina superior izquierda del límite de relleno) o **porcentaje** sobre el tamaño de la caja (el punto n% de la imagen se sitúa en el punto n% del area de relleno), de manera que "0% 0%" indica un ajuste a la esquina superior izquierda, "100% 100%" a la inferior derecha y "50% 50%" centrada. **Si se expresa una sola se entiende como "x" dándose a "y" el valor 50%**.

Palabras reservadas, poniendo una o dos en cualquier orden. Si falta una se le da el valor "center" , de manera que por ejemplo "left" es equivalente a "center left".

Esta propiedad **no se hereda**.

```
XX unidad | % | top | center | bottom | left | right (el valor por defecto es la esquina superior izquierda "0% 0%")

P.nota {background-image: white url("fondo.gif");
        background-repeat : repeat-y;
        background-position: top center;}
```

background

Esta propiedad en sí representa un metodo abreviado para especificar todos los valores en un solo marcador.

```
transparent | color | url | repeat | scroll | position
```

Sobre la utilización del color

El color puede ser especificado mediante nombre o mediante especificación RGB.

Los nombres son los pertenecientes a la paleta VGA de Windows: **aqua, black, blue, fucshia, gray, green, lime, marron, navy, olive, purple, red, silver, teal, white, y yellow.**

La especificación RGB se puede efectuar de tres formas. La primera es mediante tres números hexadecimales, donde el primero corresponde a la cantidad de rojo, el segundo a la cantidad de verde y el tercero a la cantidad de azul.

Por ejemplo.

```
#FF0000 - Sera el color rojo
#00FF00 - Sera el color verde
#0000FF - Sera el color azul
```

y con la combinación de estos podemos conseguir los colores que queramos.

```
#FFFF00 - Amarillo
```

El mismo color puede ser especificado mediante la funcion `rgb()` indicando el código de color o el porcentaje:

```
rgb(0,255,0) --color verde
rgb(0%,100%,0) --color verde
```

A4. Propiedades de margenes y padding

Con estas propiedades especificamos los margenes de cualquier elemento y con el padding controlamos la distancia entre el borde y el contenido.

Las propiedades que podemos utilizar son las siguientes:

margin-top

Fijamos el margen superior.

Con la propiedad **auto** el navegador un valor adecuado dependiendo del tipo de elemento.

```
XX unidades | % | auto (el valor por defecto es 0)
```

```
BODY {margin-top: 1cm}
```

margin-bottom

Fijamos el margen inferior. No se hereda.

```
XX unidades | % | auto (el valor por defecto es 0)
```

margin-left

Fijamos el margin izquierdo. No se hereda.

```
XX unidades | % | auto (el valor por defecto es 0)
```

margin-right

Fijamos el margen derecho. No se hereda.

```
XX unidades | % | auto (el valor por defecto es 0)
```

margin

Mediante esta propiedad **podemos especificar todos los valores de los margenes de una sola vez**.

Se pueden poner hasta cuatro valores, para especificar cada uno de los margenes.

```
margin: ancho1 ancho2 ancho3 ancho4
```

Pero si sólo especificamos **ancho1** se refíere a los cuatro lados del margen. Si se especifican dos valores, **ancho1** se refíere a los lados superior e inferior y **ancho2** al izquierdo y derecho. En el caso de poner los tres primeros, **ancho1** se refiere al lado superior, **ancho2** al izquierdo y derecho y **ancho3** al inferior.

Esta propiedad **no se hereda**.

```
XX unidades | % | auto (el valor por defecto es 0)
```

```
BODY {margin: 1in 2in}
```

En el ejemplo le estamos dando a la página un margen superior e inferior de 1 pulgada e izquierdo y derecho de 2 pulgadas.

padding-top

Distancia entre el borde superior y el contenido. No se hereda.

```
XX unidades | %
```

padding-bottom

Distancia entre el borde inferior y el contenido.No se hereda.

```
XX unidades | %
```

padding-left

Distancia entre el borde izquierdo y el contenido. No se hereda.

```
XX unidades | %
```

padding-right

Distancia entre el borde derecho y el contenido. No se hereda.

```
XX unidades | %
```

padding

Mediante esta propiedad **podemos definir en una única propiedad los cuatro valores anteriores**. Funciona de la misma manera que la propiedad margin.

Esta propiedad **no se hereda**.

```
XX unidades | %
```

```
H1 {padding: 10%}
```

A5. Propiedades de los bordes

Con estas propiedades especificamos el ancho, color y estilo del area de borde.

Las propiedades que podemos utilizar son las siguientes:

border-top-width
border-bottom-width
border-left-width
border-right-width

Mediante estas cuatro propiedades **especificamos el ancho del area de borde por encima, debajo, a la izquierda y derecha del elemento**.

Se puede especificar una longitud o las palabras reservadas **thin**, **medium** y **tick** para que el navegador asigne tres valores normalizados de grosor creciente y constantes en el documento.

Estas propiedades **no se heredan**.

```
thin | medium | thick | XX unidad
```

```
H1 {border-top-width: thin}
```

border-width

Especificamos **en una única propiedad el ancho de todo el area de borde**. Funciona de la misma manera que la propiedad margin.

Esta propiedad **no se hereda**.

```
thin | medium | thick | XX unidad
```

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

border-style

Especificamos el estilo de visualización del borde.

Los posibles valores son:

none ninguno, anchura cero	**double** doble
dottet punteado	**groove** hundido
dashed discontinuo	**ridge** resaltado
solid liso	**inset** toda la caja hundida
	outset toda la caja resaltada

Esta propiedad **no se hereda**.

```
none | dottet | dashed | solid | double | groove | ridge | inset
| outset
```

border-color

Especificamos el color del borde. No se hereda.

```
nombre del color | valor HEX | Rgb (R%, G%, B%) | Rgb (R,G,B)
```

border-top
border-bottom
border-left
border-right

En cada una de las cuatro propiedades podemos **especificar en conjunto todas las características de cada uno de los lados del borde**: tamaño, estilo y color.

```
anchura | estilo | color
```

border

Mediante esta propiedad podemos **especificar de forma conjunta todas las propiedades de un borde**: tamaño, estilo y color.

```
anchura | estilo | color
```