TEMA 4: Hojas de estilos.

# 1.- Introducción a CSS.

**Las hojas de estilo en cascada (CSS - Cascading Style Sheets) son un estándar W3C que define la presentación de los documentos Web**, es decir, el modo en el que se muestra un documento en pantalla o se suministra al usuario, ya sea por el monitor, en la pantalla del teléfono móvil o leído por un lector de pantalla.

Lo más importante es que con CSS se mantienen las instrucciones de presentación separadas del contenido del documento HTML.

Las hojas de estilo en cascada como hoy las conocemos, comenzaron cuando **Håkon Lie** publicó su primer borrador de hojas de estilo HTML en cascada, al que pronto se le unió **Bert Bos,** gran impulsor de este estándar.

* **CSS Nivel 2, Revisión 1** que es simplemente una recomendación que realiza unos ajustes menores a CSS2 basándose en la experiencia de trabajo con él entre 1998 y 2004.
* **CSS Nivel 3,** basada en módulos, añade soporte para texto flotante vertical, mejor manejo de tablas, idiomas internacionales y una mejor integración con otras tecnologías XML como SVG, MathML y SMIL.

El CSS WG también está trabajando en conjuntos CSS especiales orientados a medios específicos como: CSS Mobile, CSS Print y CSS TV.

[Actividades del grupo W3C WG (Se abre en una ventana nueva)](http://www.w3.org/Style/CSS)

Los navegadores actuales implementan bastante bien CSS1 desde 1999 (tres años después de su lanzamiento) aunque, dependiendo de la marca y versión del navegador, hay algunas pequeñas diferencias de implementación. El primer navegador en dar soporte completo al CSS1 ha sido Internet Explorer 5.0 aunque hasta ese momento fue el Opera el que soportaba mejor CSS1. Después otros navegadores también lo han ido implementando.

Sin embargo, en los navegadores más recientes hay diferentes niveles de implementación de CSS2.1.

## 1.1.- Añadir estilos a un documento con CSS.

No existe ninguna desventaja por utilizar CSS en la maquetación de páginas web, son todo ventajas y, entre ellas, podemos destacar las siguientes:

* **Mayor control** en el diseño de las páginas: Se puede llegar a diseños fuera del alcance de HTML.
* **Menos trabajo**: Se puede cambiar el estilo de todo un sitio con la modificación de un único archivo.
* **Documentos más pequeños**: Las etiquetas **<font>** y la gran cantidad de tablas empleadas para dar una buena apariencia a los sitios web desaparecen ahora, por lo que se ahorra código en la configuración de la presentación del sitio.
* **Documentos mucho más estructurados**: Los documentos bien estructurados son accesibles a más dispositivos y usuarios.
* **El HTML de presentación está en vías de desaparecer**: Todos los elementos y atributos de presentación de las especificaciones HTML fueron declarados obsoletos por el W3C.
* **Tiene buen soporte**: En este momento, casi todos los navegadores soportan casi toda la especificación CSS1 y la mayoría también las recomendaciones de nivel 2 y 2.1.

Pero ¿cómo funciona CSS? El funcionamiento podemos resumirlo en tres pasos:

1. Hay que comenzar con un documento HTML. En teoría, el documento tendrá una estructura lógica y un significado semántico a través de los elementos HTML adecuados. Con HTML se crea **la estructura de la página web.**
2. Luego hay que **escribir las reglas de estilo para definir el aspecto ideal de todos los elementos.** Las reglas seleccionan el elemento en cuestión por su nombre y, a continuación, listan las propiedades (fuente, color, etc.) y los valores que se le van a aplicar.
3. Por último, hay que **vincular los estilos al documento**. Las reglas de estilo pueden reunirse en un documento independiente y aplicarse a todo el sitio, o pueden aparecer en la cabecera y aplicarse solo a ese documento.

Las hojas de estilo consisten en una o más reglas que describen cómo debería mostrarse en pantalla un elemento.

A la hora de aplicar las reglas de estilo a un documento HTML, **existen tres modos** distintos:

* Estilos en línea.
* Hojas de estilos incrustados.
* Hojas de estilos externas: vinculadas o importadas.

|  |
| --- |
| ESTILOSModos de aplicación **Estilos en línea.**  Se puede añadir información de estilo a un elemento concreto de un documento HTML o XHTML, en adelante (X)HTML, empleando el atributo ***style*** en la etiqueta del elemento. Esta operación se realiza en el propio documento. Su sintaxis general es:  <etiqueta(X)HTML style=″declaraciones de estilo″  Aunque este es un uso perfectamente válido de la información de estilo, los estilos en línea contaminan el documento con información de presentación.  Debe emplearse con moderación, solo ocasionalmente, para ignorar reglas de nivel más alto.  De hecho, el atributo style fue depreciado en XHTML 1.1 y no aparece en otros lenguajes XML.  **Hojas de estilos incrustadas.**  Un método más compacto para añadir hojas de estilo es incrustar un bloque de estilo en la parte superior (en la cabecera) del documento (X)HTML empleando el elemento ***style***.  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN">  <html>  <head>  <style type=″text/css″ media=″screen″>  /\* hoja de estilo \*/  </style>  <title>Mi primera página web con estilo</title>  </head>  /\* Aquí iría el body \*/  </html>  El elemento ***style*** debe colocarse en la cabecera del documento (dentro del head).  Es obligatorio emplear el atributo ***type*** en los documentos (X)HTML. De momento, el único estilo viable es ***text/css***.  El atributo ***media*** permite seleccionar el medio en el que se va a aplicar el estilo: pantalla (screen), impresión, dispositivo móvil, etcétera.  Los navegadores más antiguos, que ignoran el elemento ***style*** normalmente mostrarían su contenido como si fuera parte del cuerpo del documento (body), de modo que el usuario vería el texto correspondiente a las reglas de estilo en lugar de su aplicación en el contenido de la página. Para evitarlo, las reglas de estilo se ponen dentro de un comentario empleando <!-- y --> para abrir y cerrar el comentario, respectivamente.  Este método puede emplearse cuando sólo tenemos un documento web que utiliza un único estilo.  **Hojas de estilos externas importadas.**  Se puede importar una hoja de estilo con la sentencia ***@import*** de CSS. Esta sentencia puede usarse en un archivo CSS o dentro del elemento style visto anteriormente en las hojas de estilo incrustadas.  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN">  <html>  <head>  <style type=“text/css” media=“screen”>  @import url("dirección web o local del archivo css");  /\* aquí podrían ir otros @import u otras reglas de estilo \*/  </style>  <title>Mi primera página web con estilo</title>  </head>  /\* Aquí iría el body \*/  </html>  Ten en cuenta, si utilizas este método, que:   * Las sentencias ***@import*** deben estar al principio de la hoja de estilo, antes que cualquier otra regla. * El orden en el que se importan las hojas de estilo importa, prevaleciendo en caso de conflicto el del último archivo importado.   **Hojas de estilos externas vinculadas o enlazadas.**  Es el modo más potente de utilizar CSS.  Se reúnen todas las reglas de estilo en un documento de texto independiente y se crean vínculos a ese documento desde todas las páginas de un sitio.  De este modo, se pueden hacer cambios en el estilo de un modo homogéneo en todo el sitio editando la información del estilo en un único documento.  Para hacerlo se emplea con el elemento ***link*** en el documento (X)HTML.  El elemento ***link*** se coloca en la cabecera (head) del documento, tal y como se ve en el siguiente ejemplo:  <head>  <link rel="stylesheet" href= ″../estilos/miestilo.css″ type="text/css" />  </head>  El atributo ***rel*** define la relación del documento vinculado con el documento actual.  El atributo ***href*** indica la URL del documento de hoja de estilo. Puedes vincular más de una hoja de estilo en un documento, y ten en cuenta que será muy importante el orden en el que las pongas pues se aplicarán “de arriba hacia abajo”, es decir, la última hoja de estilos enlazada modificaría los estilos que coincidan con los de las anteriores. |
| Reglas de estiloSintaxis Las reglas de estilo en las hojas de estilo incrustadas y en las hojas de estilo externas. Veamos primero dos ejemplos de reglas y lo que hace cada una de ellas.  Regla de estilo 1: Establece el color gris para la etiqueta de cabecera de tamaño 1 (etiqueta h1 del HTML).  h1 {color: #EEEEEE;}  Regla de estilo 2: Configura un tipo de fuente específico de nombre “Verdana” con un tamaño de 12 píxeles de alto para todos los párrafos (etiqueta p del HTML) y, en caso de no existir esta fuente en el equipo configurará una de la familia sans serif.  p {font-size: 12px; font-family: Verdana, sans-serif;}  Con estos ejemplos vemos que hay dos partes diferenciadas en una regla: lo que está a la izquierda, fuera de las llaves ({}) y lo que está dentro de ellas.  Podemos decir que una regla se construye con un conjunto de declaraciones aplicadas a un selector, donde el selector es lo que está a la izquierda de las llaves y, el conjunto de declaraciones es lo que está dentro de ellas. Es decir:  selector {declaraciones;}  Decimos un conjunto de declaraciones porque, dentro de las llaves puede ir 1, 2 ó más declaraciones siempre que vayan separadas entre sí por el símbolo del punto y coma (;). Es decir:  selector {declaración1; declaración2; …)  Ten en cuenta que los puntos suspensivos son sólo para indicar que pueden ir más declaraciones, pero no se pondrían dentro de la regla.  Siguiendo con los dos ejemplos anteriores, podemos decir nuestras reglas tienen:  2 selectores:  h1, p  3 declaraciones:  color: #EEEEEE; font-size: 12px; font-family: Verdana, sans-serif;  Ahora diremos que una declaración se construye con dos elementos separados por el símbolo de dos puntos (:) que son: propiedad y valor. Es decir, la sintaxis de una regla con una sola declaración sería:  selector {propiedad: valor;}  En nuestros ejemplos serían:  **selector**: h1 y p  **propiedad**: color, font-size y font-family  **valor**: #EEEEEE, 12px y Verdana, sans-serif  Hay que destacar que cuando a una propiedad como font-family se le asignan dos posibles valores estos van separados por comas (,) y a la hora de establecer uno de ellos intentará siempre hacerlo de izquierda a derecha.  Podemos decir que las dos secciones principales de una regla de una hoja de estilo son:   * El **selector** (que identifica a **qué elemento** se le va a aplicar el estilo) * La **declaración** (que es un conjunto formado por pares propiedad: valor).   Las llaves permiten definir declaraciones múltiples, separadas por ";", que componen un bloque de declaración.  Un ejemplo de bloque de declaración será el siguiente:  p {font-size: 11px; font-weight: bold; color: #CC0066;}  Si tenemos en cuenta que CSS ignora tanto los espacios en blanco como los saltos de línea, la regla anterior:  p {font-size: 11px; font-weight: bold; color: #CC0066;}  podría escribirse así:  p {  font-size: 11px;  font-weight: bold;  color: #CC0066;  }  Incluso podría escribirse así:  p { font-size: 11px;  font-weight: bold;  color: #CC0066;}  Todo es cuestión de gustos y de hábitos. Pero, cuando hay una lista muy grande de reglas con múltiples declaraciones cada una,***es importante ser consistente en la forma de escribirlas y distinguir claramente el selector y las diferentes declaraciones***.***De esta forma resulta más fácil de mantener***.  **Estilos en línea.**  /\* aquí iría parte del documento (X)HTML \*/  <h1 style="color: red">Esta cabecera es de color rojo</h1>  <p style="font-size: 12px; font-family: 'Trebuchet MS', sans-serif">  /\* Este texto tiene un tamaño de 12px y un tipo de letra Trebuchet MS\*/  </p>  /\* aquí iría parte del documento (X)HTML \*/  **Hojas de estilos incrustadas.**  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN">  <html>  <head>  <style type=″text/css″ media=″screen″>  p {font-size: 12px; font-weight: bold; color: #CC0066;}  </style>  <title>Mi primera página web con estilo</title>  </head>  /\* aquí iría el body \*/  </html>  **Hojas de estilos externas incrustadas.**  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN">  <html>  <head>  <style type=″text/css″ media=″screen″>  @import url(../estilos/miestilo.css);  </style>  <title>Mi primera página web con estilo</title>  </head>  /\* Resto del documento HTML \*/  **Hojas de estilos externas enlazadas.**  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN">  <html>  <head>  <link rel="stylesheet" href=″../estilos/miestilo.css ″ type="text/css" />  </head>  <title>Mi primera página web con estilo</title>  </head>  /\* Resto del documento HTML \*/ |

## 1.2.- Hojas de estilo externas.

Modo de emplear las hojas de estilo externas: importándolas o enlazándolas. También has visto cómo se crea una regla de estilo y sus componentes: selector, propiedad y valor.

Las hojas de estilo son documentos de texto con, por lo menos, una regla. Estos archivos no contienen ninguna etiqueta HTML, ¿para qué? Al igual que en los documentos HTML, en las hojas de estilo se pueden incluir comentarios, pero en este caso, se escriben del siguiente modo:

**/\* Este es un comentario \*/**

CSS2 introduce la posibilidad de orientar las hojas de estilo a medios de presentación específicos. Para ello se emplea el atributo **media** del elemento **link**.

La siguiente tabla muestra los valores que puede tomar el atributo **media**:

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

A continuación, se muestra cómo se pueden emplear la función **@media** de la misma forma que hacíamos con la función **@import**.

|  |
| --- |
| <style type="text/css">  @import url(http://estilos/miestilo.css);  @media print {  body {  font-size: 10pt;  }  /\* Establece el tamaño de fuente para impresión \*/  }  @media screen {  body {  font-size: 13px  }  /\* Establece el tamaño de fuente para visualización \*/  }  p {  font-family: Verdana;  }  </style> |

## 1.3.- Conceptos clave de CSS.

**Estructura y herencia.**

Un documento HTML tiene una estructura determinada que es equivalente a un árbol genealógico cuando se hace referencia a la relación entre elementos:

* Un elemento es "hijo" de otro si está contenido directamente en él y este último pasa a ser su "padre". Pj: el elemento **p** es hijo del elemento **body** y el elemento **body** es padre del elemento **p**.
* Los elementos que tienen el mismo padre son "hermanos". Pj: un elemento **p** puede ser hermano de otro elemento **p** si ambos son hijos directos del elemento **body**.

Controlar la **relación padre-hijo** es fundamental para el funcionamiento de CSS. Un hijo puede "**heredar**" valores de propiedad de su padre. Con una buena planificación, la herencia puede emplearse para hacer más eficiente la especificación de los estilos.

Este principio por el que algunas reglas se ignoran y otras se heredan nos introducen un concepto muy importante: "**la cascada**".

**Reglas de estilo en conflicto: la "cascada".**

La "cascada", de las hojas de estilo en cascada, se refiere a lo que ocurre si varias fuentes de información de estilo quieren dar formato al mismo elemento de una página. Cuando un navegador encuentra un elemento para el cual hay varias declaraciones de estilo, **las ordena de acuerdo con el origen de la hoja de estilo**, la especificidad de los selectores y el orden de la regla para poder determinar cuál aplicar.

**Origen de la hoja de estilo.**

**Los navegadores** otorgan un **peso** distinto a las hojas de estilo que, ordenadas de menor a mayor peso, son:

* Hojas de estilo del navegador.
* Hojas de estilo del lector.
* Hojas de estilo de la persona que ha diseñado la página web.
* Declaraciones de estilo **!important** del lector.

Existe otra jerarquía de pesos que se aplican a las hojas de estilo creadas por la persona que ha diseñado la página web. Es importante entender esta jerarquía y tener en cuenta que las reglas de estilo que están al final de la lista ignorarán a las primeras. La siguiente lista, que como la anterior está ordenada de menor a mayor peso, muestra esta otra jerarquía:

* Hojas de estilo externas vinculadas (empleando el elemento **link** en la cabecera del documento).
* Hojas de estilo externas importadas (empleando el elemento **@import** dentro del elemento **style** en la cabecera del documento).
* Hojas de estilo incrustadas (empleando el elemento **style** en la cabecera del documento).
* Estilos en línea (empleando el atributo **style** en la etiqueta del elemento).
* Declaraciones de estilo marcadas como **!important**.

**Especificidad del selector.**

Hasta ahora se tuvieron en cuenta las distintas fuentes de la información del estilo. Pero aún puede existir algún conflicto a nivel de reglas. Por esa razón, "la cascada" continúa a nivel de reglas. Lo verás mejor con el siguiente ejemplo, que podría estar en una hoja de estilo externa o incrustada. En él se muestran dos reglas que hacen referencia al elemento **strong**.

**strong {color: red;} h1 strong {color: blue;}**

En el ejemplo anterior, todo el texto del documento (X)HTML marcado con la etiqueta **strong** aparecerá en color rojo. Sin embargo, si el texto marcado con la etiqueta **strong** aparece dentro de una cabecera de primer nivel (**h1**), su color será azul. Esto ocurre porque un elemento en un contexto determinado **es más específico** que en un contexto general y, por lo tanto, tiene **más peso.** Debes tener en claro que, cuanto más específico sea el selector se le dará más peso para ignorar las declaraciones en conflicto.

**Orden de las reglas.**

Cuando una hoja de estilo contiene varias reglas en conflicto de igual peso, sólo se tendrá en cuenta la que está en último lugar. En el siguiente ejemplo, todas las cabeceras de primer nivel del documento serían rojas porque se impone siempre la última regla:

**h1 {color: green;} h1 {color: blue;} h1 {color: red;}**

**Tipos de elementos CSS. Elementos de bloque y en línea.**

En CSS, la noción de "elemento de bloque" y "en línea" es puramente de presentación. Un elemento de bloque de CSS siempre genera saltos de línea, antes y después de él, mientras que los elementos en línea de CSS no lo hacen, aparecen en el flujo de la línea y sólo pasarán a otra línea si no tienen espacio.

En (X)HTML, los párrafos (**p**), cabeceras (como **h1**), listas (**o**l, **ul**, **dl**) y contenedores (**div**) son los elementos de bloque más comunes, mientras que, el texto enfatizado (**em**), las anclas (**a**) y los elementos **span** son los elementos en línea más comunes.

Con CSS podrás indicarle al navegador cómo quieres que se vea en el documento empleando para ello los atributos **block** e **inline** de la propiedad **display**, independientemente de que un elemento sea de bloque o en línea.

## 1.4.- El modelo de cajas de CSS.

Es un concepto fundamental para comprender el funcionamiento de las hojas de estilo. Aquí podrás ver una introducción básica a este modelo. De acuerdo con este modelo, **todos los elementos** de una página web **generan una caja** rectangular alrededor llamada **"caja del elemento".**

En estas **cajas** se pueden configurar **propiedades** como bordes, márgenes y fondos (entre otras). Las cajas también se pueden emplear para posicionar los elementos y diseñar la página.

Las cajas de elementos están hechas de cuatro componentes principales:

* **Contenido del elemento**: es lo que está en el núcleo de la caja está el.
* **Relleno** (**padding**): es el espacio que rodea al contenido.
* **Borde** (**border**): es la parte que perfila el relleno.
* **Margen** (**margin**): es el espacio que rodea al borde, la parte más externa del elemento.

Hay algunas características fundamentales del modelo de cajas que vale la pena destacar:

* El relleno, los bordes y los márgenes son opcionales, por lo que, si ajustas a cero sus valores se eliminarán de la caja.
* Cualquier color o imagen que apliques de fondo al elemento se extenderá por el relleno.
* Los bordes se generan con propiedades de estilo que especifican su estilo (por ejemplo: sólido), grosor y color. Cuando el borde tiene huecos, el color o imagen de fondo aparecerá a través de esos huecos.
* Los márgenes siempre son transparentes (el color del elemento padre se verá a través de ellos).
* Cuando definas el largo de un elemento estás definiendo el largo del área de contenido (los largos de relleno, de borde y de márgenes se sumarían a esta cantidad).
* Puedes cambiar el estilo de los lados superior, derecho, inferior e izquierdo de una caja de un elemento por separado.

[Enlace a la web de la W3C que habla sobre las propiedades CSS relacionadas con el modelo de cajas.](https://www.w3c.es/Divulgacion/GuiasReferencia/CSS21/#modeloCajas)

## 1.4.1.- Área de contenido y relleno.

**Área de contenido.**

Recuerda que el área de contenido es la parte más interna de la caja. En el ejemplo siguiente se muestra cómo se pueden modificar las propiedades que afectan al tamaño del área de contenido: su ancho (**width**) y su altura (**height**).

**div {width:100px; height:200px; }**

Existen otras propiedades interesantes que nos permiten ajustar la altura y el ancho máximo y mínimo del área de contenido de las cajas. Estas propiedades son, respectivamente: **max-height, max-width, min-height y min-width.**

**Relleno.**

El relleno es una cantidad opcional de espacio existente entre el área de contenido de un elemento y su borde. Es conveniente que establezcas un valor de relleno cuando pones borde a un elemento.

Para establecer el relleno se emplea la propiedad **padding**. Esta propiedad, como muchas otras en CSS, obliga a configurar los valores en un orden determinado. Estos valores y su orden son: **top** (arriba), **right**derecha, **bottom** (debajo) e **left** (izquierda).

El ejemplo siguiente muestra una tabla que agrupa algunos ejemplos de la asignación de valores y su interpretación por CSS. En todos los ejemplos se ha empleado como selector el elemento **div**.

| **Ejemplos de asignación de valores a la propiedad padding.** | |
| --- | --- |
| **EJEMPLO** | **INTERPRETACIÓN** |
| div {padding: 3px 20px 3px 20px;} | Establece un relleno para todos los elementos div de 3 píxeles por encima del área de contenido, 20 píxeles a su derecha, 3 píxeles por debajo y 20 píxeles a su izquierda. |
| div {padding: 3px 20px 3px;} | Al omitir un valor, asume que el valor del relleno a la izquierda es el mismo que el de la derecha. |
| div {padding: 3px 20px;} | Al omitir dos valores, asume que el primer valor corresponde al relleno por encima y por debajo del área de contenido y, el segundo valor corresponde al relleno a la derecha y a la izquierda. |
| div {padding: 3px;} | Al omitir tres valores, asume que ese valor es el mismo para todos. |

Otras características interesantes del relleno son:

* El valor del relleno se sumará al de **width** ya definido en el elemento.
* Su color es el mismo al del área de contenido.
* El relleno nunca se "colapsa". Esto lo entenderás cuando veas los márgenes que sí se colapsan.

En el siguiente ejemplo se muestra cómo configuramos el relleno y el color de fondo del área de contenido de los elementos h1 del documento. El color del área de relleno será el mismo que el del elemento.

**h1 {padding: 4px 10px; background: #ccc; }**

## 1.4.2.- Bordes.

Un borde es una línea dibujada alrededor del área de contenido de un elemento y de su relleno (**padding**), aunque ya vimos que éste último era opcional.

Los bordes funcionan, a la hora de establecer sus valores, de la misma manera que el relleno visto anteriormente, siguiendo un orden: **superior, derecho, inferior, izquierdo**, siempre en el sentido de las agujas del reloj y comenzando en las 12. Es fácil de recordar.

Se pueden establecer valores distintos para cada uno de los bordes y omitir valores, al igual que hacíamos con el relleno.

Podemos configurar el estilo del borde, su grosor y su color. Las propiedades son:

* **Border-style**: con esta propiedad configuramos el estilo del borde. Esta propiedad es, sin duda, la más importante del borde, ya que, si no está presente el borde no existirá.

La propiedad **border-style** puede tener los valores: **none**, **hidden**, **dotted**, **dashed**, **solid**, **double**, **groove**, **ridge**, **inset**, **outset**e **inherit**.

**div {border-style: solid dashed dotted double; } div {border-style: solid dashed dotted double; }**

* **Border-width**: con esta propiedad configuramos el grosor del borde.

Los valores que puede tomar esta propiedad son: **thin**, **medium**, **thick**, **inherit** o un valor concreto en píxeles. Si no se especifica esta propiedad tomará medium como valor por defecto.

**div {border-style: solid; border-width: thin medium thick 12px; }**

* **Border-color**: con esta propiedad configuramos el color del borde.

Si no especificamos el color el elemento coge el del "primer plano", es decir que si, por ejemplo, tenemos una caja en cuyo interior hay texto, el color del borde será el color del texto.

Existe el color **transparent** pero no está soportado por todos los navegadores.

**div {border-style: solid; border-width: 4px; border-color: #333 #red rgb(0,0,255) #0044AC; }**

La propiedad **border** une todas las propiedades "**border**" vistas hasta ahora. En esta propiedad, a diferencia de las anteriores, no hay que colocar los valores en ningún orden concreto. La propiedad **border**se emplea cuando se quieren configurar los cuatro lados por igual.

También tenemos las propiedades: **border-top**, **border-right**, **border-bottom** y **border-left**.

**h1 {border: .5em solid blue;}**

**h2 {border-left: solid blue .5em;}**

**h3 {border-right: solid .5em;}**

## 1.4.3.- Márgenes.

El margen es la cantidad de espacio que se puede añadir alrededor del borde de un elemento.

Esta propiedad se configura con la propiedad **margin**. Al configurar esta propiedad debemos tener en cuenta que, a la hora de establecer los valores para los márgenes, hay que emplear la misma filosofía que con la propiedad **padding**.

Los márgenes **top** y **bottom** de dos elementos que van **seguidos se "colapsan".** Es decir, se asume como margen entre ambos elementos el mayor de ellos. El siguiente ejemplo muestra lo que ocurre cuando tenemos dos elementos un **h1** y un **h2** colocados uno a continuación del otro.

**h1 {margin: 10px 20px 10px 20px;}**

**h2 {margin: 20px;}**

En el primer caso el margen superior e inferior es de 10px. En el segundo caso es de 20px. El espacio resultante entre los dos elementos será de 20px.

Por el contrario, si fuesen dos elementos colocados "uno al lado del otro" (como dos elementos **span**), deberíamos tener en cuenta que los márgenes **right** y **left** no se colapsan, **sino que se suman.**

## 2.- Selectores.

El selector es la parte de la regla de estilo que identifica el elemento concreto al que se aplicarán las instrucciones de presentación. CSS ofrece varios tipos de selectores que permiten mejorar la flexibilidad y la eficiencia en la creación de hojas de estilo.

[Reglas sintácticas de los estilos CSS.](http://www.maestrosdelweb.com/editorial/cssintaxis/)

[Patrones de selectores CSS](https://www.w3.org/wiki/CSS_/_Selectores_CSS)

## 2.1.- Selectores de elemento.

Los selectores de elemento son los más sencillos. Son aquellos que se aplican a un elemento (etiqueta) del lenguaje HTML.

**h1 {color: blue;}**

**h2 {color: blue;}**

**p {color: blue;}**

Si te fijas en los ejemplos, verás que se está definiendo la misma propiedad (**color**) en todos los elementos e incluso se está asignando el mismo valor (blue). El ejemplo siguiente muestra cómo se puede escribir una única regla aplicada a varios selectores a la vez.

**h1, h2, p {color: blue;}**

Cuando se realiza una declaración sobre varios selectores a la vez, éstos se separan por comas.

Existe un **selector** de elementos "**universal**" representado por el **asterisco (\*).** El ejemplo siguiente muestra una regla que pondrá en gris todos los elementos del documento que no tengan especificado otro color.

**\* {color: grey;}**

## 2.2.- Selectores contextuales.

Los **selectores de elemento** se aplican a todos los casos en los que se encuentre el elemento en el documento HTML. En cambio, los selectores contextuales permiten aplicar estilos a los elementos basándose en su contexto o en su relación con otro elemento.

Hay varios tipos de selectores contextuales:**descendente, hijo y hermano.**

Los **selectores descendentes** hacen referencia a elementos que están contenidos en otro elemento. Un selector descendente se pone a continuación del selector en el que está contenido separado de él por un espacio en blanco. El siguiente ejemplo especifica que los elementos **em** deben tener color azul, pero sólo si son descendientes de un elemento de lista (**li**). El resto de los elementos em no se verán afectados.

**li em {color: blue;}**

Los selectores descendentes también pueden estar anidados en varias capas de profundidad. El siguiente ejemplo pone de color amarillo sólo el texto enfatizado (**em**) de las anclas (**a**) que se encuentren en las listas ordenadas (**ol**).

**ol a em {color: yellow;}**

Si se emplea el selector descendente combinado con el selector universal, se puede restringir el alcance de un selector descendente. El siguiente ejemplo muestra dos párrafos que contienen un hipervínculo. En el primer caso el elemento ancla es descendiente directo del elemento de párrafo y, en el segundo caso, es descendiente directo del elemento **span** que, a su vez, lo es del elemento de párrafo.

**<p><a href="#">Enlace</a></p>**

**<p><span><a href="#">Enlace</a></span></p>**

Examinemos las dos reglas de estilo siguientes:

**p a { color: red; }**

**p \* a { color: red; }**

Con la primera se consigue que se muestren los dos enlaces de color rojo, mientras que con la segunda regla sólo se mostraría en rojo el segundo de los enlaces. La razón es que el selector **p \* a** se traduce como "todos los elementos de tipo **<a>** que se encuentran dentro de cualquier elemento que, a su vez, se encuentra dentro de un elemento de tipo **<p>**". Como el primer elemento **<a>** está directamente incluido en un elemento **<p>**, no se cumple la condición del selector **p \* a**.

Un **selector hijo** es un caso concreto de un selector descendente en el un selector está contenido directamente en otro, sin que existan niveles intermedios. Un selector hijo se escribe a continuación de su selector padre separándolo de él por el símbolo "mayor que" (**>**). En el siguiente ejemplo se pone en gris el fondo del texto enfatizado, pero sólo si es hijo directo de un párrafo.

**p > em {background-color: gray;}**

El **selector adyacente** se utiliza para hacer referencia a un elemento que sigue inmediatamente a otro en el código, con el que comparte el mismo elemento padre. Un selector adyacente se escribe a continuación de otro selector separándolo de él por el símbolo de suma (**+**). El siguiente ejemplo pondría en color azul el primer párrafo que sigue a una cabecera de primer nivel.

**h1 + p {color: blue;}**

## 2.3.- Selectores de clase e ID.

Para poder hacer uso de selectores más específicos, se hace necesario introducir los conceptos de identificador (**id**) y clase (**class**).

Identificadores (**id**).

Los elementos HTML disponen de un atributo llamado identificador (**id**), que tiene como finalidad identificar al de manera excluyente. De este modo, CSS u otro lenguaje podrá hacer referencia a él y distinguirlo del resto de los elementos del documento.

Un **id** debe ser único en cada documento (X)HTML.

**<p id="textocabecera">**

Se recomienda que el valor del id sea un nombre que caracterice o clarifique, de forma breve y esquemática al elemento y que, además, sea fácilmente reconocible por el programador. Se utilizan con frecuencia para identificar las secciones principales de las páginas: contenido, cabecera, pie, etcétera.

Para escribir una regla de estilo que se aplique a un determinado identificador hay que escribir el símbolo de la almohadilla (**#**) seguido del nombre del identificador. El ejemplo siguiente muestra algunas formas de establecer el tamaño de la fuente en 14 píxeles al elemento p identificado como "textocabecera" del ejemplo anterior:

**p#textocabecera {font-size: 14px;}**

**#textocabecera {font-size: 14px;}**

Con la primera regla indicamos que se aplique el estilo a un párrafo cuyo identificador sea "textocabacera" pero, como el id es único en cada documento, **realmente basta con la segunda forma para decir lo mismo**, porque no va a haber otro elemento **<p>** o diferente de **<p>**que tenga ese mismo identificador.

Si tenemos varios elementos que necesitan un tratamiento similar, emplearemos el atributo **class**.

**Clases** (**class**).

Se emplea el atributo **class** para identificar distintos elementos como parte de un grupo conceptual. Así, los elementos de una clase pueden modificarse con una única regla de estilo.

En el siguiente ejemplo se muestra como dos elementos distintos se clasifican de la misma forma mediante la asignación del valor "especial" al atributo **class**.

**<h1 class="especial">¡Atención!</h1>**

**<p class="especial">Hoy tenemos grandes rebajas.</p>**

También se puede hacer que un elemento pertenezca a más de una clase separando **sus nombres de clase con espacios.** En el siguiente ejemplo el párrafo pertenece a dos clases: "textocabecera" y "especial".

**<p class="textocabecera especial">Hoy tenemos grandes rebajas.</p>**

Para escribir una regla de estilo que se aplique a todos los elementos de una determinada clase hay que escribir un punto seguido del nombre de la clase. Por ejemplo: **.especial {color: green;}**

El siguiente ejemplo muestra la forma de lograr que todos los elementos de la clase "especial" tengan un color verde a excepción de las cabeceras de primer nivel que tienen que ser rojas.

**.especial {color: green;}**

**h1.especial{color: red;}**

Los nombres de clases y de identificadores no pueden contener espacios en blanco.

[Recomendaciones generales sobre CSS.](http://www.librosweb.es/css/capitulo13/recomendaciones_generales_sobre_css.html)

## 2.4.- Pseudoselectores.