

Guía (muy) básica de hojas de estilo (CSS) y de marcado semántico de páginas web (v1.)

Lluís Codina
UPF. Abril 2007

1. Hojas de estilo: una base para el marcado semántico

Una hoja de estilo es un conjunto de especificaciones que declara cómo deben mostrarse los componentes de una página web, es decir, cómo debe interpretar el navegador, a efectos de presentación visual, los distintos elementos presentes en una página web. El formato de cada elemento de una página web se especifica mediante declaraciones de estilo denominadas reglas.

2. Reglas

Una regla CSS se compone de tres partes:

1. Selector
2. Propiedad
3. Valor

Una regla simple organiza los tres componentes anteriores de acuerdo a la siguiente estructura:

```
selector {propiedad: valor;}
```

Por ejemplo, la regla siguiente evita que un elemento marcado con h1 aparezca en negrita.:

```
h1 { font-weight: normal; }
```

Una regla puede contener más de una propiedad con sus respectivos valores, separadas entre ellas por punto y coma (se considera una buena práctica añadir también punto y coma al final). Ejemplo:

```
h1 { font-weight: normal; color: red; }
```

Una regla puede dar formato a más de un elemento a la vez, separando estos mediante comas. Ejemplo:

```
h1, h2, h3, h4 { font-weight: normal; color: red; }
```

3. Declaración

Las reglas de las hojas de estilo se pueden declarar en tres sitios distintos:

1. En la página web
2. En un archivo externo
3. En un elemento

En los casos 1 y 2, las reglas se declaran con la misma sintaxis (que es la que ya hemos visto). En el caso número 3, hay una sintaxis específica ya que se declara dentro de un elemento (X)HTML y debe respetar sus convenciones. Además, cada forma de declaración necesita el procedimiento de vinculación específico que se muestra a continuación.

3.1. En la página web

La hoja de estilo se declara en la sección [head](#), dentro del elemento [style](#). Por ejemplo:

```
<head>
<title>Aquí va un título</title>
<style type="text/css">
h1, h2, h3 { font-weight: normal; color: blue; }
</style>
</head>
```

3.2. En un archivo externo

Las declaraciones se escriben en un archivo de texto con extensión **.css** sin ningún tipo de preámbulo.

En la sección [head](#) de la página se debe indicar el enlace con el archivo que contiene las declaraciones de la hoja de estilo mediante el elemento [link](#) y los atributos [rel](#), [href](#) y [type](#):

```
<head>
<title>Aquí va un título</title>
<link rel="stylesheet" href="estilo.css" type="text/css" />
</head>
```

Como se puede ver, el atributo `rel` (*relation*) advierte que se trata de una hoja de estilo, `href` aporta el nombre (y el camino si fuera necesario) del archivo y `type` indica que se trata de un archivo de texto que contiene una hoja de estilo que sigue la norma CSS.

3.3. En un elemento

Los estilos se pueden aplicar a nivel de un elemento individual. En el siguiente ejemplo, se aplica un estilo específico a un elemento `acronym`. En condiciones normales, un texto marcado con `acronym` aparecería en letra normal (redonda). En el siguiente ejemplo, se ha definido un estilo a nivel de una ocurrencia este elemento para que la palabra marcada con `acronym` en este caso aparezca en cursiva:

```
<p><acronym style="font-style: italic">W3C</acronym> son las siglas
del World Wide Web Consortium</p>
```

Como se puede ver hay un cambio significativo en la sintaxis. Se utilizan las mismas denominaciones para las propiedades y sus valores, pero, por compatibilidad con el lenguaje XHTML, todo el conjunto propiedad/valor de CSS se delimita mediante comillas y se presenta a su vez como un valor de del atributo `style` de (X)HTML.

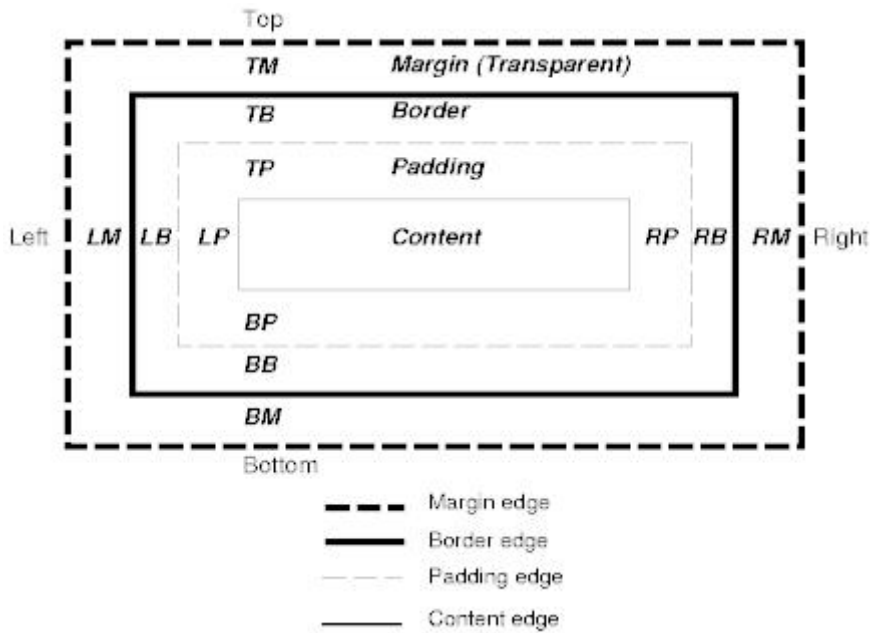
4. Cajas

Desde el punto de vista de CSS, todo lo que hay en una página web es una caja. Lo que hace un selector es identificar la caja a la que se aplica el formato.

Las cajas tienen:

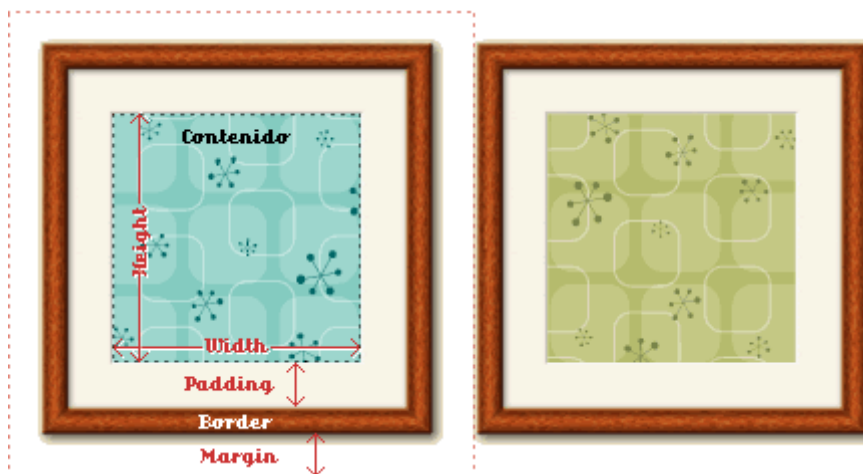
- **Contenido** (*content*). El contenido de la caja: texto, imagen, etc.
- **Relleno** (*padding*). La distancia entre el contenido y el borde.
- **Borde** (*border*). El borde de la caja.
- **Margen** (*margin*). La distancia entre el borde y el elemento contenedor (p.e. `body`) u otro elemento adyacente.

El siguiente diagrama muestra modelo oficial de cajas de CSS según el W3C:



Los componentes de la caja se pueden dividir en cuatro partes: *top*, *bottom*, *right* y *left*. Por tanto, en la ilustración anterior, las siglas LM, RM, etc., significan *LeftMargin*, *Right Margin*, etc.

La figura siguiente, debida a Kemie Guaida, ilustra estas ideas comparando el modelo de cajas de CSS con un cuadro colgado en la pared y de acuerdo con el cual *margin* sería la relación entre el cuadro y la pared, *border* el grosor del marco, etc.:



Las hojas de estilo pueden aplicar formato (p.e. color, anchura, tipo de línea, etc.) a cualquiera de los componentes de esta caja, así como permite posicionar estas cajas en el conjunto de la página.

De este modo, si queremos que un elemento muestre el borde en forma de puntos de color azul, podemos generar esta declaración:

```
p { border-style: dotted; border-color: blue}
```

A partir de lo anterior, este código en la página:

```
<p>Ejemplos de hojas de Estilo</p>
```

Se mostrará así:



Si queremos ampliar el relleno entre el contenido y el borde así como tener un tamaño de punto más pequeño podemos modificar la regla anterior acudiendo a las propiedades de borde (`border`) y de margen (`padding`):

```
p { border-style: dotted; border-width: 2px; border-color: blue;
padding: 15px; }
```

Ahora, el navegador lo mostrará así:



5. Herencia

Las cajas están contenidas dentro de otras cajas y heredan los estilos de la caja contenedora si no hay una regla de nivel específico. Por ejemplo, si declaramos un tipo de letra para el elemento `body`, todos los elementos que están dentro, como `h1`, `h2`, `p`, etc. heredarán este tipo de letra.

En el siguiente ejemplo declaramos un color de letra para `body` y otro distinto para `h2`. De este modo, todos los elementos, excepto `h2`, tendrán el color declarado para `body`, mientras que `h2` tendrá su propio color.

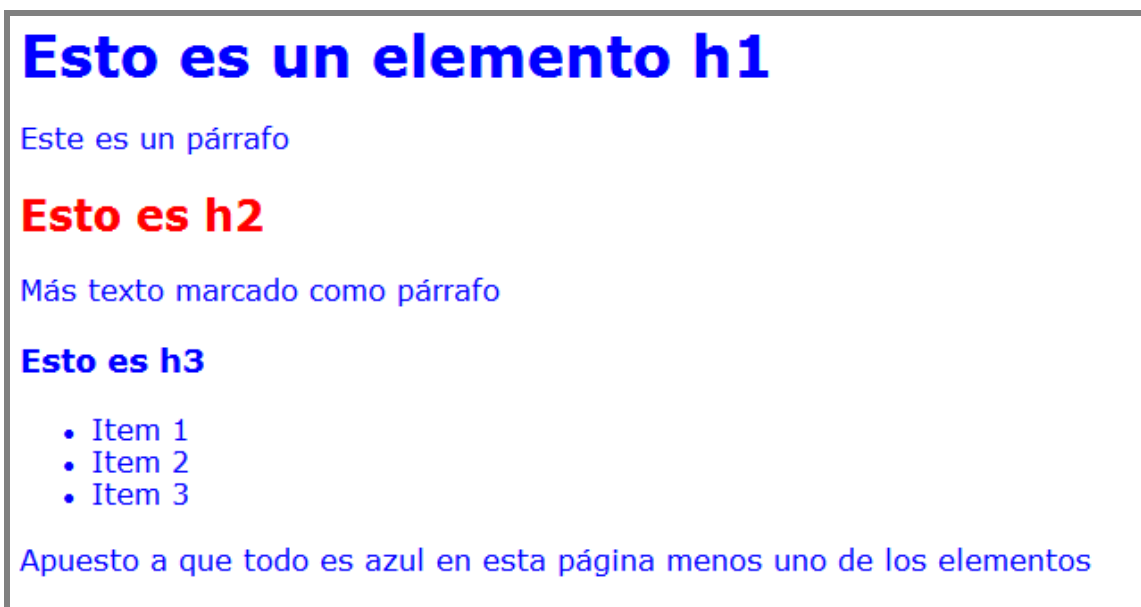
Veamos. Primero la declaración de reglas CSS:

```
body { color: blue }
h2 { color: red }
```

Ahora el código html de la página:

```
<h1>Esto es un elemento h1</h1>
<p>Este es un párrafo</p>
<h2>Esto es h2</h2>
<p>Más texto marcado como párrafo</p>
<h3>Esto es h3</h3>
<ul>
<li>Item 1</li>
<li>Item 2</li>
<li>Item 3</li>
</ul>
<p>Apuesto a que todo es azul en esta página menos uno de los
elementos</p>
```

Ahora, el resultado en el navegador:



6. Tipos de selectores

Existen dos grandes categorías de selectores:

- *Elementos*: son los nombres correspondientes a elementos del lenguaje (X)HTML como `body`, `h1`, `p`, `table`, etc.
- *Nombres propios*: son nombres que puede crear el autor de la hoja de estilo. De esta clase de selectores, existen, a su vez dos tipos: de **clase** (*class*) y de **identidad** (*id*).

Por tanto, en realidad tenemos en total **tres** tipos de selectores que examinaremos a continuación.

6.1. Selectores de elemento

Se declaran utilizando como selector un elemento (X)HTML, p.e.:

```
h1 { color: blue }
```

De este modo, el aspecto de todos y cada uno de los elementos que forman el lenguaje (X)HTML puede ser modificado mediante un regla. Cabe recordar que los navegadores disponen de una hoja de estilo interna que es la que otorga formato a los elementos (X)HTML cuando no hay ninguna hoja de estilo específica. Por ejemplo, aún sin hoja de estilo la mayor parte de los navegadores aplican negrita a los elementos marcados como **strong**, así como un tipo de letra superior + negrita a los elementos marcados con **h1**, etc.

6.2. Selectores de clase

Estos selectores se declaran mediante una palabra propia que asigna el autor de la hoja de estilo. Esta palabra va precedida por un punto. Por ejemplo: supongamos que se necesita una clase de selectores para formatear títulos de películas (pongamos que se trata de una web sobre cinema). El autor de la hoja de estilo puede crear el selector con el nombre **tituloFilm** para formatear de la misma forma los títulos de los films. Para ello, escribe esta regla:

```
.tituloFilm { font-size: 1.5em; font-family: Times New Roman; color: blue; }
```

Se aplica mediante el atributo **class** seguido por el nombre del selector (sin el punto). Por ejemplo, en el siguiente código fuente solo a uno de los dos elementos de párrafo (**p**) se le ha añadido esta clase de selector:

```
<p class="tituloFilm">2001: Una odisea del espacio</p>  
<p>La obra de Kubrick marcó un punto de inflexión en el género de la ciencia-ficción</p>
```

La declaración solamente afectará a uno de ellos y el resultado es que el primer párrafo tendrá un aspecto muy distinto de los demás:

2001: Una odisea del espacio

La obra de Kubrick marcó un punto de inflexión en el género de la ciencia-ficción

6.3. Selectores de identidad

Los selectores de identidad se nombran mediante una palabra propia precedida por el símbolo # (almohadilla). Por ejemplo:

```
#navegacion { background-color: LightGrey; }
```

Los selectores de identidad solo se pueden aplicar a un elemento en cada página. Dicho al revés, y tomando el ejemplo anterior: en cada página solamente puede haber un elemento #navegacion (en cambio, podemos tener múltiples elementos de clase en una misma página).

En este caso, hemos supuesto que queremos dar un estilo propio a la zona de navegación de cada página. Se supone que deseamos destacar la barra de navegación mediante un fondo gris y queremos que la fuente del texto en esa zona sea un poco más pequeña. La declaración sería la siguiente:

```
#navegacion { background-color: LightGrey; font-size: 0.8em }
```

El código fuente en la página sería el siguiente:

```
<div id="navegacion">
<ul>
<li><a href="item1.htm">Item 1</a></li>
<li><a href="item2.htm">Item 2</a></li>
<li><a href="item3.htm">Item 3</a></li>
</ul>
</div>
```

Se supone que Item 1, Item 2, etc., son opciones de menú. El resultado sería este:

- [Item 1](#)
- [Item 2](#)
- [Item 3](#)

7. Uso de div y span

El lenguaje (X)HTML dispone de dos elementos con gran potencialidad cuando se utilizan junto con las hojas de estilo, y cuya característica

principal, a diferencia de elementos como `body`, `p`, `h1`, etc., es que no poseen ningún significado intrínseco. Son los siguientes:

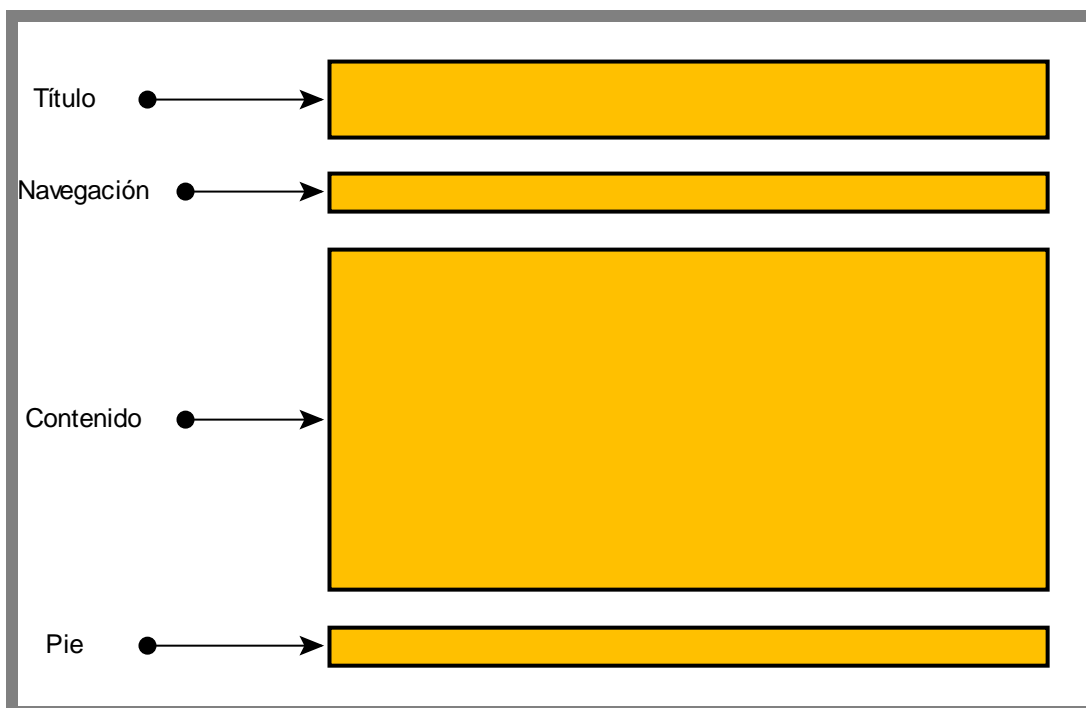
- `div`
- `span`

Mientras el elemento `div` es de bloque, el elemento `span` es de línea. Recordemos que los elementos de bloque tienen un salto de línea integrado. El ejemplo más conocido es `p`. Un elemento de línea, por el contrario no genera ningún espacio a su alrededor, sino que se mantienen en la misma línea que lo contiene. Un buen ejemplo puede ser `strong`.

Tanto `div` como `span` pueden utilizarse para mejorar la apariencia de una página, pero además contribuyen a añadir valor semántico a la misma.

7.1. `div`

En concreto, `div` ayuda a crear la estructura básica de la página. Por ejemplo, podemos imaginar un sitio web cuyas páginas se estructuran en tres grandes secciones, como muestra la figura siguiente:



Con una estructura como la anterior, el código fuente de cada página puede contener cuatro elementos `div`, cada uno de ellos identificado con un atributo `id` diferente, de este modo:

```
<body>
<div id="tituloPag">
</div>
<div id="navegacion">
</div>
<div id="contenido">
</div>
<div id="piePag">
</div>
</body>
```

La estructuración anterior permitirá definir estilos para las cuatro secciones de la página sin perjuicio de aplicar estilos más específicos mediante selectores de elemento, de clase, etc. Esta estructuración facilitará el mantenimiento de los estilos del sitio y, en caso necesario, el posicionamiento de los mismos en la página.

7.2. span

El elemento `span` suele utilizarse también combinado con `id` o con `class` para especificar con tanto nivel de detalle el aspecto de cualquier elemento de la página. Por ejemplo, podemos desear marcar y tratar de forma distinta el apellido de una lista de nombres, para lo cual podemos usar `span` de este modo. Definimos un selector de clase `apellido` para el cual declaramos un estilo:

```
.apellido { font-style: italic; font-weight: bold }
```

Aplicamos entonces `span` y `class` a los componentes (`li`) de la siguiente lista:

```
<h1>Realizadores de los años 90</h1>
<ul>
  <li>Martin <span class="apellido">Escorsese</span></li>
  <li>James <span class="apellido">Cameron</span></li>
  <li>Pedro <span class="apellido">Almodóvar</span></li>
</ul>
```

El resultado en el navegador será el siguiente:

Realizadores de los años 90

- Martin *Escorsese*
- James *Cameron*
- Pedro *Almodovar*

8. Posicionamiento

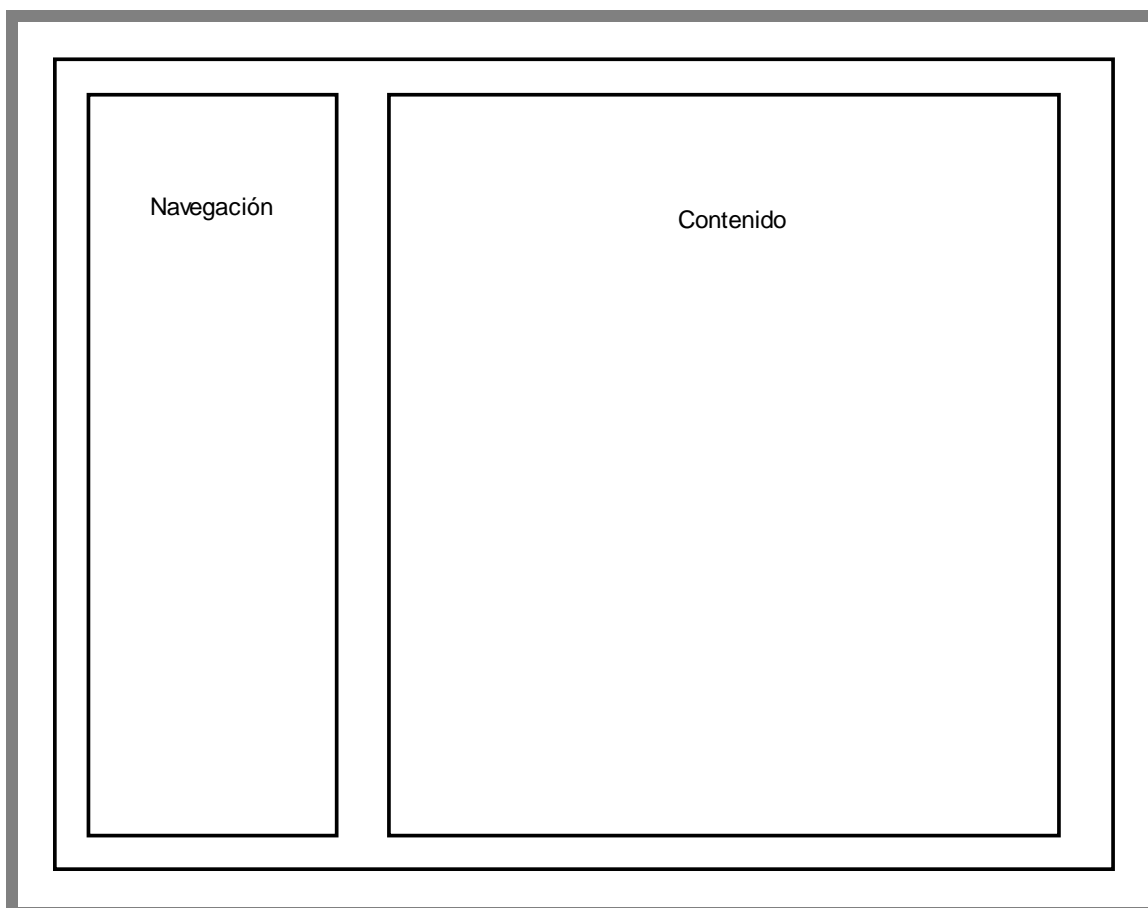
Las hojas de estilo proporcionan propiedades que permiten posicionar de forma fija, absoluta o relativa (*fixed, absolute, relative*) las cajas en una página web. Estas propiedades, junto con los selectores de clase y de identidad que permiten identificar y clasificar las cajas respectivamente, proporcionan al diseñador un dominio total de la estructura de la página sin necesidad de recurrir, por ejemplo, al uso de tablas.

A título ilustrativo, mostramos un ejemplo típico de código de estilo con indicaciones de posicionamiento:

```
#navegacion { position: absolute; top: 0; left: 0; width: 30%; }
```

```
#contenido { width: 70%; margin-left: 30%; }
```

El código anterior generaría dos secciones como las que muestra la figura siguiente:



Lo cierto es que el posicionamiento mediante CSS merece un capítulo aparte y por ello escapa a los objetivos de esta presentación. Pero no hemos querido dejar de señalar su importancia con el fin de animar a todo aquel que, precisamente, quiera avanzar en estos temas.

9. Conclusiones y buenas prácticas

9.1. Conclusiones

El uso (adecuado) de hojas de estilo es una de las bases de la codificación semántica, la cual descansa sobre el principio esencial de separar el contenido de la presentación.

De este modo, se promueve la utilización de los elementos de (X)HTML con valor semántico, y no para dar formato. Éste último es responsabilidad de las reglas que se declaran en las hojas de estilo. Por ejemplo, si no nos gusta que el título principal de una página aparezca en negrita, la buena práctica consistirá en modificar la apariencia de `h1`, y no en marcar el título con un elemento que no le corresponda (o al revés, si nos interesa dotar a una palabra o una frase de un tamaño mayor, la clave estará en definir su presentación con una regla de estilo y no en asignarle, por ejemplo, un marca de cabecera que no le corresponda).

9.2. Buenas prácticas

Para finalizar, mostraremos una lista de algunas buenas prácticas:

1. Declarar la hoja de estilo en un archivo separado (mejor que en la hoja de estilo, salvo necesidad específica).
2. Utilizar medidas relativas (tanto por ciento en lugar de píxeles, o unidades `em` en lugar de puntos, etc.).
3. Usar elementos semánticos (`h1`, `h2`, `blockquote`, etc.) para organizar los contenidos de la página, no para conseguir una determinada apariencia de la misma.
4. Modificar la apariencia de la página exclusivamente mediante estilos (y no con elementos o atributos desaconsejados como `font`, `center`, etc. o usando elementos fuera de lugar).
5. Limitar al máximo declarar estilos dentro de elementos individuales (es decir, evitar el uso del atributo `style` dentro de elementos).

6. Denominar los selectores de clase y de identidad con nombres de función, p.e.: `tituloFilm`, `menuGlobal`, `menuLocal`, etc., (y no con el nombre de la apariencia resultante, p.e., `negrita`, `letraMenor`, `letraGrande`, etc.). Por tanto: usar nombres *semánticos* y no de *apariciencia*.
7. Organizar los elementos principales de la página, p.e. cabecera, navegación, contenido, etc., con secciones marcadas como elementos `div`.
8. Utilizar elementos `div` (así como atributos y selectores `id`) con propiedades de posicionamiento para establecer la estructura de la página (*layout*) si fuera necesario en lugar de tablas o *frames*.