Tema 12. Nociones básicas de CSS

Indice

1. Declaración de estilo CSS.

La declaración de un estilo se realiza mediante el binomio propiedad: valor;

Ejemplo

```
background-color: red;
```

Lo que puede interpretarse como: "asignar al color de fondo el valor rojo".

Detallemos esta declaración:

- La propiedad identifica al elemento que se definirá con el estilo adoptado. Estas propiedades están enumeradas en las especificaciones CSS. Existen numerosas propiedades relacionadas, por ejemplo, con el tipo de letra (font), con el texto (text), con el fondo (background), con el borde (border), etc.
- La propiedad se separa de su valor por el signo de dos puntos.
- Están permitidos los espacios. De este modo algunos programadores tienen el hábito de situar un espacio entre los dos puntos y el valor de la propiedad, para mejorar la legibilidad del código.
- El valor identifica la naturaleza del efecto de estilo deseado. El valor se expresa mediante una palabra clave, un porcentaje, o un tamaño en función de la propiedad a la que se asigna.
- Una declaración de estilo termina siempre con un punto y coma.
- Es posible definir varias declaraciones de estilo en el mismo selector. Por ejemplo: propriedad1:valor; propriedad2:valor; propriedad3:valor;

2. Selectores básicos.

2.1. Selectores de etiqueta.

En el capítulo anterior hemos definido las hojas de estilo como elementos agregados al código Html5.

Las propiedades de estilo se "enganchan" con los elementos y etiquetas Html. A esto se le llama selector.

La sintaxis de un selector de etiqueta consiste en el nombre de la etiqueta seguido de la declaración de estilo entre llaves.

```
selector { declaración de estilo }
```

Ejemplo

```
div { background-color: red;}
```

Observe que se indica únicamente el nombre de la etiqueta, sin los signos menor que (<) y mayor que (>). Es decir, solamente el texto de la etiqueta.

Esto puede leerse como: "*aplicar a todas las etiquetas de división el efecto de estilo descrito entre llaves".*

También es posible aplicar el mismo efecto de estilo a distintos selectores de etiqueta. En este caso, cada una de las etiquetas implicadas se escribe separada con una coma.

Por ejemplo:

```
h1,h2,h3 { declaración de estilo }
```

Lo que significa que las etiquetas <h1>, <h2> y <h3> tendrán el mismo efecto descrito entre llaves.

2.2. Selectores de clase.

Hasta el momento hemos visto cómo se aplican las hojas de estilo a un selector, siendo este selector una etiqueta Html.

Esta noción de selector está demasiado limitada. Sería bueno poder aplicar varios estilos a una misma etiqueta. Esto lo aportan las clases. Con ellas, el diseñador podrá, de algún modo, definir sus propios selectores.

La definición de una clase se realiza de la forma siguiente:

```
.nombre_de_la_clase { declaración(es) de estilo}
```

Es decir un punto, seguido del nombre que queremos atribuir a la clase, seguido de la declaración de estilo escrita entre llaves.

Ejemplo

```
.rojo { background-color: red;}
```

Esta definición de clase puede utilizarse para cualquier etiqueta del documento o de la aplicación. De ahí el término "*clase universa*l".

No obstante, la clase así definida no tiene ningún efecto si no se invoca en el documento.

La llamada a la clase se realiza del siguiente modo:

```
<etiqueta class="nombre_de_la_clase">
```

Ejemplo

```
<div class="rojo"> ... </div>
```

Comentarios

- El punto del selector de clase no se utiliza a la hora de llamar a la clase class (error frecuente).
- Una misma clase se puede invocar varias veces en el documento Html. Por ejemplo:

```
<div class="rojo">Item 1<div>

<div>Item 2<div>

<div class="rojo">Item 3<div>
```

Aquí solamente las divisiones con la clase rojo tendrán un color de fondo rojo.

- Un nombre de clase puede estar formado por letras, cifras, el guión y el carácter de subrayado. El primer carácter no puede ser un número, un guión o un carácter de subrayado. Hay que evitar los espacios así como las palabras reservadas de JavaScript.

Existe otra forma de definir una clase, aunque es menos práctica (y menos utilizada) que la anterior:

```
nombre_etiqueta.nombre_de_la_clase {declaración(es) de estilo}
```

Es decir un nombre de etiqueta, seguido de un punto, seguido del nombre que queremos asignar a la clase, seguido de la declaración de estilo entre llaves.

Ejemplo

```
blockquote.rojo { background-color: red;}
```

Con esta definición de clase, no podremos aplicarla más que a la etiqueta seleccionada, en este caso blockquote.

2.3. Selectores de identificador.

El selector id, también llamado identificador id, permite aplicar una hoja de estilo (como el selector class) aunque sólo se podrá invocar una única vez en el documento Html.

El selector id permite, por tanto, identificar un elemento único en la página. Esta distinción cobra especial relieve cuando se procesa mediante JavaScript o Dhtml. En efecto, id no puede emplearse más de una vez en un documento, por un único elemento, que podrá tratarse como un objeto único que se manipulará mediante JavaScript.

La definición de un selector id es:

```
#nombre_del_identificador {declaración(es) de estilo}
```

Es decir una almohadilla (#), seguida del nombre que vamos a proporcionar al identificador id, seguido de la declaración de estilo entre llaves.

Ejemplo

```
#rojo { background-color: red;}
```

Esta selección sólo se podrá utilizar una única vez en el documento.

El selector así definido no tiene ningún efecto si no se invoca en el documento.

La llamada se hace mediante:

```
<etiqueta id="nombre_del_identificador">
```

Ejemplo

```
<div id="rojo"> ... </p>
```

Comentarios

- Un identificador id no puede figurar más de una vez en el documento Html. En caso contrario, ¡el resto del código será incorrecto!

  ```
  <p id="rojo"> ... </p>

  ...

  <p id="rojo"> ... </p>
  ```

- Un documento puede contener varios identificadores id con nombres diferentes aunque sólo pueden utilizarse una única vez.
- Es posible mezclar declaraciones de estilo class e id.
- El nombre de un identificador puede contener letras, números, el guión y el carácter de subrayado. El primer carácter no puede ser un número, un guión o un carácter de subrayado. Es conveniente evitar los espacios así como las palabras reservadas de JavaScript.

Existe otra forma de definir una clase aunque es menos práctica (y por tanto menos utilizada) que la anterior:

```
nombre_etiqueta#nombre_del_identificador {declaración(s)
de estilo}
```

Es decir un nombre de etiqueta, seguido de una almohadilla (#), seguido del nombre que queremos asignar al identificador, seguido de la declaración de estilo entre llaves.

Ejemplo

```
blockquote#rojo { background-color: red;}
```

Con este selector, sólo podremos aplicar este estilo a la etiqueta <blockquote> designada mediante el identificador rojo.

Observación
Señalemos que las hojas de estilo CSS3 han agregado numerosos selectores, que veremos con detalle más adelante en el capítulo Hojas de estilo CSS3 - Los selectores CSS3.

3. Comentarios.

Todo lenguaje de definición o de programación permite insertar comentarios para facilitar así la comprensión del código y su mantenimiento posterior. En una hoja de estilo CSS, un comentario comienza con los caracteres de apertura barra oblicua y asterisco (es decir /*) y termina con los caracteres asterisco y barra oblicua (es decir */). Es posible situar comentarios en cualquier lugar de la hoja de estilo, salvo en el interior de una cadena de caracteres.

Ejemplo

```
/* Esto es un comentario */
```

4. Unidades de medida.

Las hojas de estilo CSS permiten utilizar numerosas unidades de medida, bien en pulgadas (inches), en centímetros, en milímetros, en puntos, en picas, en píxeles o en porcentaje.

También se distingue entre valores relativos, que pueden variar según el ordenador utilizado, y valores absolutos, que permanecen constantes sea cual sea el dispositivo o el software utilizado.

Los valores absolutos son:

Unidad	Nombre	Descripción	Valor	Ejemplo
pt	punto	72 pt = 1 inch	entero	48pt
pc	pica	1 pc = 12 pt	real	4.5pc
mm	milímetro	1 mm = .24 pc	entero	60mm
cm	centímetro	1 cm = 10 mm	entero	6cm
in	pulgada (inch)	1 in = 2.54 cm	real	0.1in

Los valores relativos son:

Unidad	Descripción	Valor	Ejemplo
em	Unidad relativa que se basa en el tamaño del tipo de letra por defecto de la página.	real	1.8em
ex	Unidad relativa a la altura de la letra minúscula en el elemento seleccionado.	real	1.3ex
px	Un píxel es la parte más pequeña de una imagen. Depende de la resolución de pantalla.	entero	220px
%	Porcentaje.	entero	80%

Comentarios

- Las unidades de medida siempre se abrevian mediante dos caracteres.
- No puede dejarse un espacio entre el valor y la unidad. Si se deja algún espacio entre el valor y la unidad, la hoja de estilo no se acepta y, por lo tanto, no se muestra.
- Ciertas propiedades aceptan valores negativos.
- Por lo general, se recomienda utilizar la unidad em para describir el tamaño del tipo de letra y así lograr una estabilidad mayor entre los distintos sistemas operativos y navegadores.

5.Notación de los colores.

Las hojas de estilo CSS proporcionan múltiples notaciones para declarar un color. Éstas son:

- La notación hexadecimal clásica, es decir **#ffcc00**. Esta notación es muy conocida para aquellos que están habituados a los colores en Html. Define el color, o más bien los tres componentes del color: rojo (r de red), verde (g de green) y azul (b de blue), mediante una notación hexadecimal de tipo #rrggbb.
- La notación hexadecimal abreviada es #fd3. Esta notación particular permite ganar algunos caracteres. Cada cifra está implícitamente duplicada. Por ejemplo, #fd3 corresponde con la notación clásica #ffdd33. De este modo, esta notación abreviada nunca podrá definir, por ejemplo, el color #cfe4f5.
- La notación decimal, por ejemplo, color: rgb(0, 0, 255). El código RGB del color no se codifica según su valor hexadecimal sino por un número comprendido entre 0 y 255. Es el equivalente decimal a la notación hexadecimal.
- La notación en porcentaje, por ejemplo, color: rgb(25%, 50%, 0%). El valor 0% significa la ausencia de dicho componente y 100% su valor máximo.
- Palabras clave, por ejemplo, color: red. Estos colores, 17 en total, se designan mediante un nombre en inglés y constituyen colores básicos. Entre ellos encontramos: green (verde), yellow (amarillo), blue (azul medio), orange (naranja), white (blanco), red (rojo), black (negro), maroon (marrón oscuro), lime (verde limón), aqua (turquesa), teal (cian oscuro), navy (azul marino), olive (aceituna), fuchsia (fucsia), purple (púrpura), silver (gris claro) y gray o grey (gris oscuro).

A estas notaciones, la especificación CSS3 añade:

- La notación RGBa, que obedece las mismas reglas de funcionamiento que la notación clásica RGB, sabiendo que la parte inicial de la declaración se corresponde con el valor: rgb(0,0,0). Se convierte por tanto en rgba(0,0,0,0). Aquí el último valor indica el grado de opacidad o de transparencia, definido entre 0 y 1.
- La notación HSL *(Hue Saturation Luminance*), en castellano Tonalidad, Saturación, Luminosidad. La notación HSL consiste en tres valores. El primero se expresa en grados, de 0° a 359° (aunque el símbolo de grado ° no aparece en la notación). Se corresponde con un color de la rueda cromática: rojo (0°), amarillo (60°), verde

(120°), cian (180°), azul (240°) y magenta (300°). El segundo y tercer valor se expresan en porcentaje y describen respectivamente la saturación y la luminosidad. Por ejemplo, color: hsl(0, 100%, 50%) para el rojo.

- La notación HSLa, que agrega un valor comprendido entre 0 y 1 para la transparencia o la opacidad. Por ejemplo, color: hsl(0, 100%, 50%, 0.5) para obtener un color rojo semitransparente.
- Para obtener una ayuda más completa referente a la codificación de colores, los siguientes sitios Web le serán de gran utilidad:
  - http://html-color-codes.info/codigos-de-colores-hexadecimales/
  - http://es.wikipedia.org/wiki/Colores_HTML
  - http://www.lawebera.es/recursos/herramientas/colores.php
  - http://www.usuarios.sion.com/pauluk/coloreshtml.htm
  - http://www.htmlquick.com/es/reference/color-codes.html

6. Convención de escritura.

Las hojas de estilo no son sensibles a la distinción entre mayúsculas y minúsculas (*case insensitive*). Poco importa entonces que las escribamos en mayúsculas o en minúsculas. No obstante, los elementos que no están bajo el control de las hojas de estilo, como por ejemplo los nombres de los tipos de letra o las URLs, puede que sí sean sensibles a las mayúsculas y minúsculas (*case sensitive*).

El uso quiere que la codificación de las hojas de estilo se realice, por lo general, en minúsculas.

7. CSS integradas en un elemento Html5.

En Html es posible agregar directamente en el código una declaración de estilo en una etiqueta determinada.

Este estilo en línea se presenta del siguiente modo:

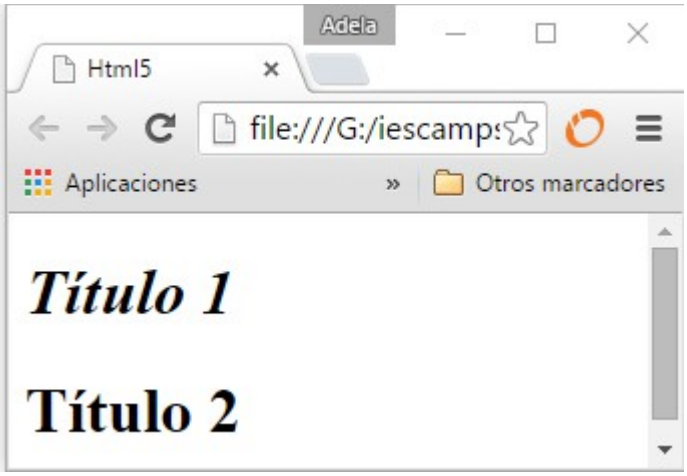
```
<h1 style="color: red;">Título de nivel 1</h1>
```

Es conveniente evitar esta forma de proceder en virtud del principio de separación entre contenido y presentación.

No obstante, puede resultar útil para dar prioridad a una propiedad de estilo respecto a la que se haya especificado en la declaración interna o externa.

Ejemplo

```
<!DOCTYPE html>
<html lang="es">
<head>
<title>Html5</title>
<meta charset="UTF-8">
</head>
<body>
<h1 style="font-style: italic;">Título 1</h1>
<h1>Título 2</h1>
</body>
</html>
```



Únicamente la primera etiqueta <h1> con la declaración de estilo se visualiza en cursiva.

8. CSS internas al documento Html5.

El uso principal de las hojas de estilo CSS consiste en determinar un estilo que se aplique a la totalidad de una página Html5.

A este efecto, el código de las hojas de estilo se agrupa en el encabezado del documento, entre las etiquetas `<head> y </head>`. Hablamos de estilo interno.

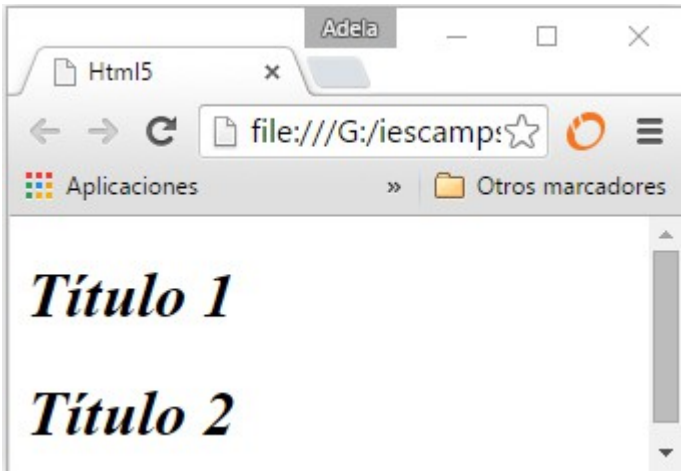
La sintaxis se presenta de la siguiente manera:

```
<head>

<style type="text/css">

h1 { color: red;}

</style>

</head>
```

Esta declaración de estilo tendrá como efecto poner en rojo el texto de todas las etiquetas <h1> del documento Html.

Ejemplo

```
<!DOCTYPE html>
<html lang="es">
<head>
<title>Html5</title>
<meta charset="UTF-8">
<style type="text/css">
h1 { font-style: italic;}
</style>
</head>
<body>
<h1>Título 1</h1>
<h1>Título 2</h1>
</body>
</html>
```



Destacamos cómo todos los títulos de nivel 1 de esta página aparecen en cursiva.

9. CSS externas al documento Html5.

Del mismo modo, es posible agrupar las declaraciones de estilo en un archivo externo (diferente) al archivo Html.

Este modo de proceder respeta mejor el principio de separación entre contenido y presentación.

Una de las ventajas de este tipo de hojas de estilo externas al documento es que es posible crear una hoja de estilo que se aplica no sólo a una única página Html sino a un conjunto de páginas en un sitio o en una aplicación.

Esta técnica emplea dos archivos:

Un archivo que contiene la declaración de estilo. Este archivo tiene la extensión **.css.**

Un documento Html que contiene un enlace hacia el archivo CSS así creado.

La hoja de estilo externa

```
h1 {text-decoration: overline;}
```

Destacamos:

- Que este archivo de texto puede crearse, por ejemplo, con el Bloc de Notas de Windows.

- Que únicamente contiene declaraciones de estilo.
- Que, por consiguiente, no puede contener etiquetas Html y en particular las etiquetas `<style type="text/css"> ... </style>`, que se usan para definir las hojas de estilo internas.
- Es preciso guardar este archivo con un nombre cualquiera seguido de la extensión .css. En nuestro caso ejemplo.css.
- Por simplicidad, este archivo se situará en la misma carpeta que el documento Html (direccionamiento relativo o local).

El documento Html5

Se agregará en el encabezado, entre las etiquetas `<head> y </head>`, un enlace hacia la hoja de estilo en cuestión.

```
<link rel="stylesheet" type="text/css" href="ejemplo.css">
```

Veamos esta línea de código:

- **link** indica al navegador que lo que sigue es un enlace.
- **rel="stylesheet"** precisa que este enlace es relativo a una hoja de estilo.
- **href="ejemplo.css**" es la escritura clásica de un enlace en Html.
- El enlace puede ser absoluto (comenzando por http://...) o relativo.
- Nada impide incluir varias etiquetas <link> hacia hojas de estilo externas diferentes.
- Nada impide utilizar, además, una hoja de estilo interna.

Ejemplo

El código del archivo ejemplo.css es simplemente:

```
h1 { text-decoration: overline;}
```

El código del archivo Html5:

```
<!DOCTYPE html>
```

```
<html lang="es">
<head>
<title>Html5</title>
<meta charset="UTF-8">
<link rel="stylesheet" type="text/css" href="ejemplo.css">
</head>
<body>
<h1>Título de nivel 1</h1>
</body>
</html>
```

Todos los títulos h1 de todas las páginas que hagan referencia a esta hoja de estilo aparecerán subrayados por encima.

10.CSS importada (@import).

Otra forma de usar las hojas de estilo externas consiste en usar el comando `@import`.

```
<style type="text/css">
@import url(ejemplo.css);
</style>
```

`@import` es una propiedad de estilo de CSS 2 (mientras que la etiqueta <link> lo es de Html). La ventaja consiste en que podrá usarse no sólo para invocar una hoja de estilo externa en un documento Html sino también para importar otra hoja de estilo en la hoja de estilo externa.

Comentarios

- Las etiquetas <style> ... </style> son necesarias, pues @import es una propiedad de estilo (CSS 2).
- No existe espacio entre la arroba y el import.
- La dirección de la hoja de estilo puede ser global (http:// …) o local.
- ¡No olvide, especialmente, el punto y coma final! Recodemos que @import es una propiedad de estilo CSS.
- La etiqueta <link> ya no es necesaria.

El código completo quedaría:

```
<!DOCTYPE html>
<html lang="es">
<head>
<title>Html5</title>
<meta charset="UTF-8">
<style type="text/css">
@import url(ejemplo.css);
</style>
</head>
<body>
<h1>Título de nivel 1</h1>
</body>
</html>
```

Igual que con los estilos externos, todos los títulos <h1> de todas las páginas del sitio web aparecerían subrayadas por encima.

La captura de pantalla es idéntica a la anterior.

11. Noción de cascada.

En los apartados anteriores hemos constatado cómo es posible tener varias definiciones de estilo, bien en línea, interna(s) o externa(s). En caso de que exista una competencia entre varios elementos de estilo, interviene la noción de "cascada" (*la palabra Cascading en Cascading Style Sheets*) u orden de prioridad.

El navegador toma primero en consideración las especificaciones de las hojas de estilo externas (con la extensión css), a continuación las hojas de estilo internas (aquellas situadas en el interior de las etiquetas <head>) y, a continuación, las hojas de estilo en línea (aquellas ligadas a un elemento Html5).

De este modo, en caso de conflicto entre una especificación de estilo definida al mismo tiempo en una hoja de estilo externa y en una hoja de estilo interna, la especificación que prevalece en el navegador es la correspondiente a la hoja de estilo interna. Del mismo modo, en caso de conflicto entre una hoja de estilo interna y una declaración en línea, es la última la que prevalece.

El orden creciente de prioridad (de menor a mayor) es:

1. Propiedades por defecto del navegador.
2. Hojas de estilo externas.
3. Hojas de estilo internas.
4. Hojas de estilo en línea.

Los estilos declarados en la hoja de estilo en línea tienen, por tanto, la prioridad más alta.

Observación
La regla de prioridad, para visualizar el documento en el navegador, consiste en utilizar la hoja de estilo más próxima al elemento.

Ejemplo

Definamos una hoja de estilo externa con los títulos de nivel 1 subrayados y una hoja de estilo interna con los mismos títulos de nivel 1 subrayados por encima.

El archivo de estilo externo (estilo1.css) es:

```
h1 { text-decoration: underline;}
```

El documento Html5:

```
<!DOCTYPE html>
<html lang="es">
<head>
<title>Html5</title>
<meta charset="UTF-8">
<link rel="stylesheet" type="text/css" href="estilo1.css">
<style type="text/css">
h1 { text-decoration: overline;}
</style>
</head>
<body>
<h1>Título de nivel 1</h1>
</body>
</html>
```

El título de nivel 1 se visualiza subrayado por encima (`text-decoration: overline`), es decir, con el efecto de estilo interno, el más próximo a la etapa de visualización.

No obstante, es posible modificar estas reglas de prioridad utilizando el valor `!important` y redefinir la prelación de una declaración, sin importar las declaraciones que puedan seguir.

Ejemplo

Se dará prioridad al estilo externo, es decir al estilo de subrayado, mediante el valor `! important.`

El archivo de estilo externo (estilo2.css) es:

```
h1 { text-decoration: underline !important;}
```

El documento Html5:

```
<!DOCTYPE html>
<html lang="es">
<head>
<title>Html5</title>
<meta charset="UTF-8">
<link rel="stylesheet" type="text/css" href="estilo2.css">
<style type="text/css">
h1 { text-decoration: overline;}
</style>
</head>
<body>
<h1>Título de nivel 1</h1>
</body>
</html>
```

12. Noción de herencia.

El principio de herencia puede explicarse recordando a Html y sus etiquetas anidadas.

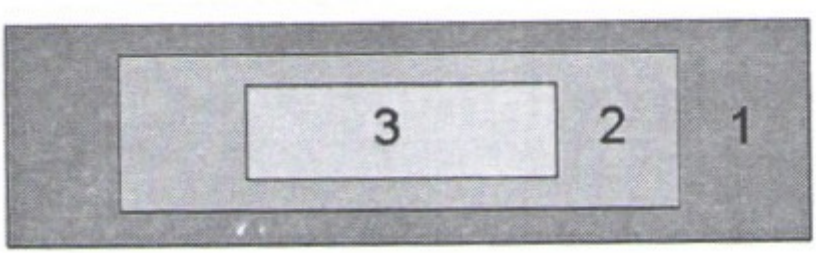
Tomemos el fragmento de código:

```
<b>Ediciones Eni <i>Colección <code>Recursos
informáticos<code></i></b>
```

Recursos informáticos se mostrará no solamente con un tipo de letra de paso fijo (las etiquetas <code> ... </code>) sino también en cursiva (incluida por las etiquetas ...) y además en negrita (etiquetas ...).

Se podría decir que Recursos informáticos hereda el formato en cursiva de la etiqueta y el formato en negrita de la etiqueta .

Del mismo modo ocurre con los estilos.



El estilo 2 se caracteriza por su propio estilo más el del estilo 1, que ha heredado.

El estilo 3 se caracteriza por su propio estilo más el de los estilos 1 y 2, que ha heredado.

Siguiendo la misma lógica podemos utilizar los términos "padre" e "hijo".

El estilo 1 sería el padre del estilo 2. Del mismo modo, el estilo 2 y el estilo 1 serían padres del estilo 3. O bien, el estilo 2 es hijo del estilo 1 y el estilo 3 es hijo de los estilos 2 y 1.

Las nociones de cascada y de herencia son muy sutiles y delicadas de comprender, y se precisarán a lo largo del estudio de las hojas de estilo.