**2.2. Comentarios**

Cuando se crea una hoja de estilo, es conveniente añadir comentarios, especialmente si es extensa y con numerosas líneas de código. De este modo es más fácil recordar, cuando se vuelva a editar el código, qué es lo que hace cada parte.

Para añadir comentarios a las hojas estilo, tanto internas como externas, se usan el carácter (**/\***) como inicio del comentario, y finalizaremos añadiendo (**\*/**). Todo lo que se encuentre entre ambas marcas no será evaluado por el navegador.

<style type="text/css">

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

div { color: red; } **\*/**

h1 { color: red; }

</style>

**2.3. Uso de mayúsculas y minúsculas**

Al escribir el código de las hojas de estilos no hay problema en que los nombres de las propiedades o de los valores aparezcan en mayúsculas o en minúsculas.

Los nombres de selectores referentes a elementos HTML (body, div, ol, ul, etc.) tampoco tienen distinción entre un tipo de carácter u otro, pero los **nombres** **de clases** y los **identificadores** **deben aparecer exactamente igual** a como se hayan definido en el código HTML.

**2.4. La importancia de la cascada y la precedencia**

[2.4. La importancia de la cascada y la precedencia](#_Toc185832920)

[2.4.1. La regla @import](#_Toc185832921)

[2.4.2. La regla !important](#_Toc185832922)

[2.4.3. El orden de la cascada](#_Toc185832923)

[2.4.3.1. Orden de cascada por especificidad](#_Toc185832924)

[2.4.3.2. Orden de cascada por ubicación](#_Toc185832925)

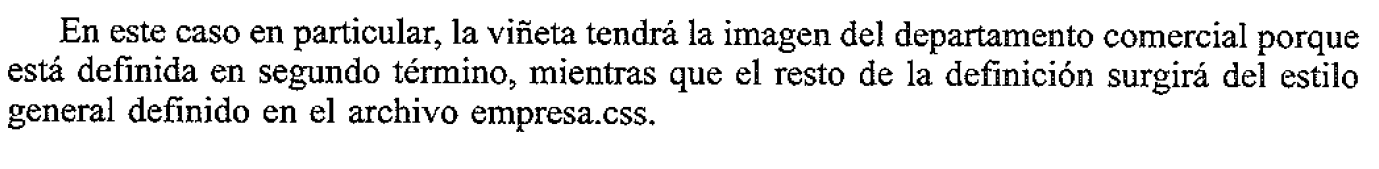
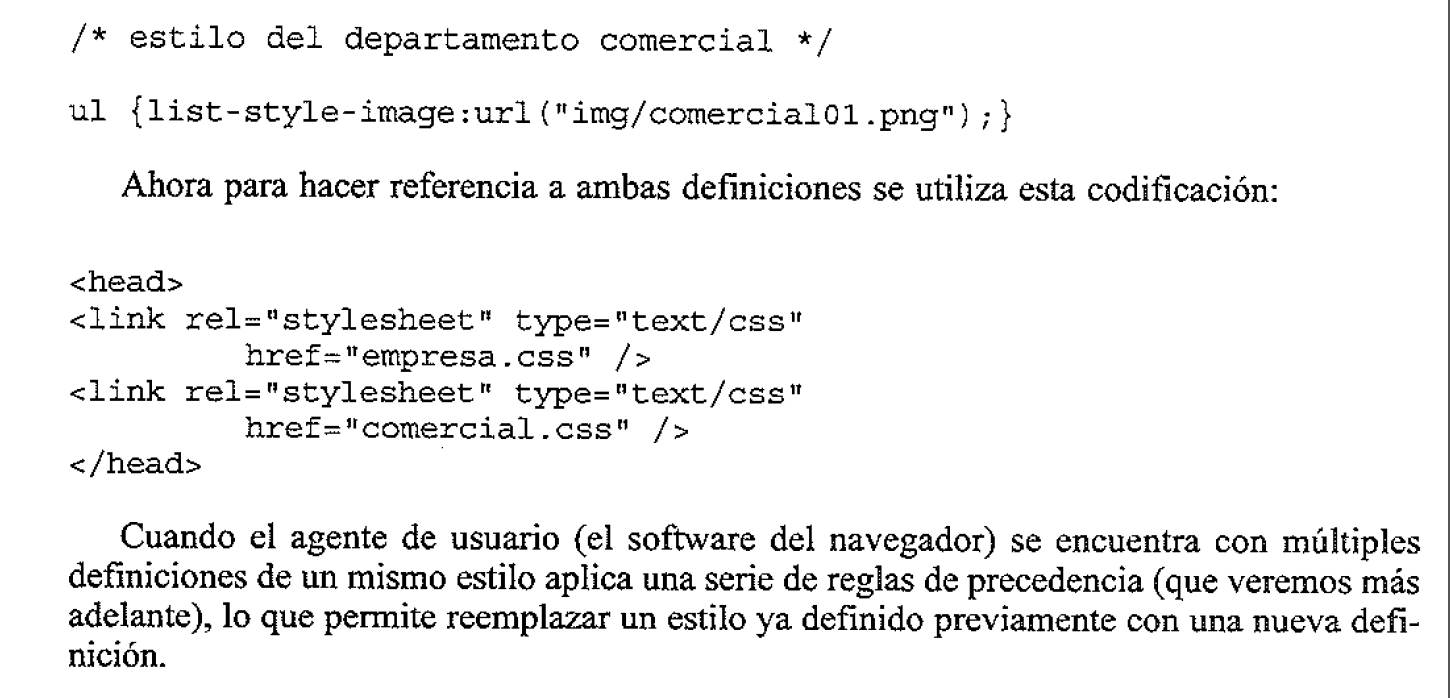
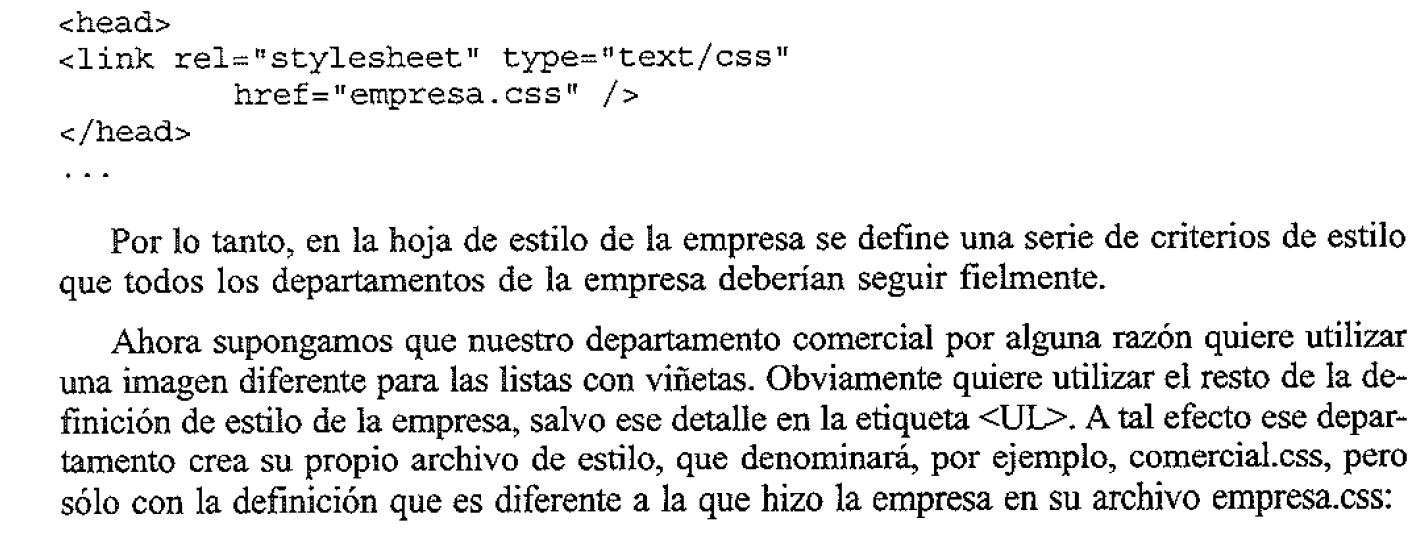
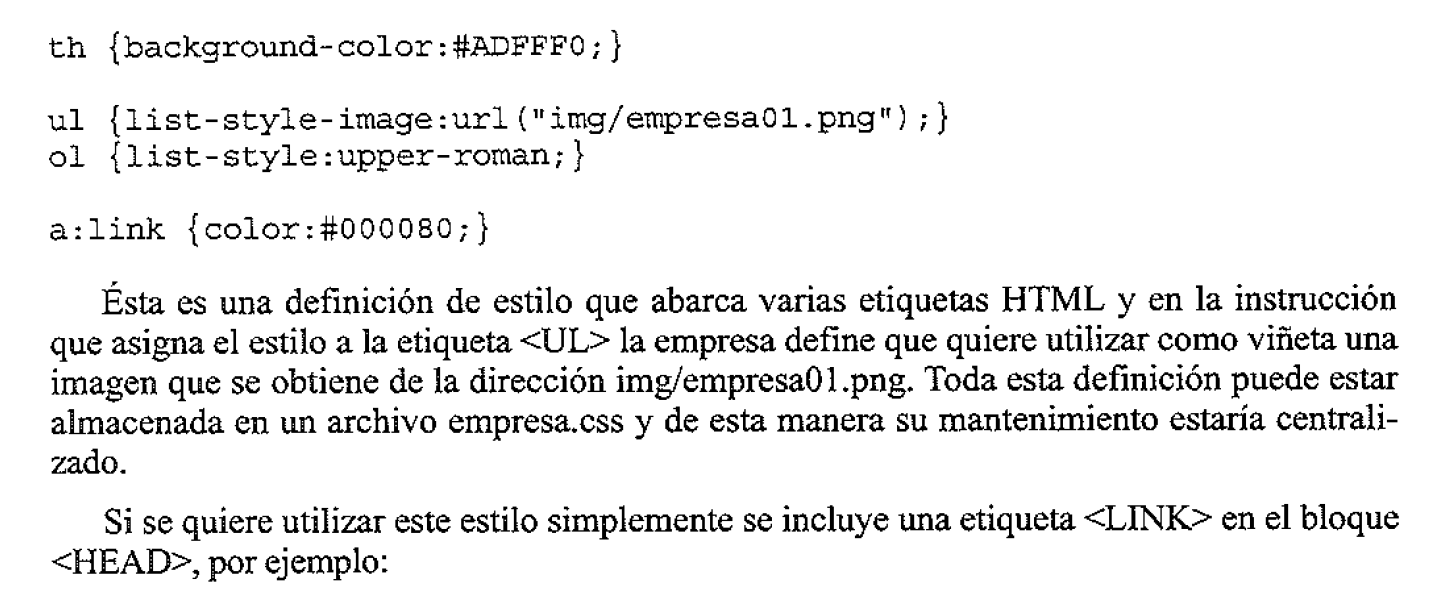
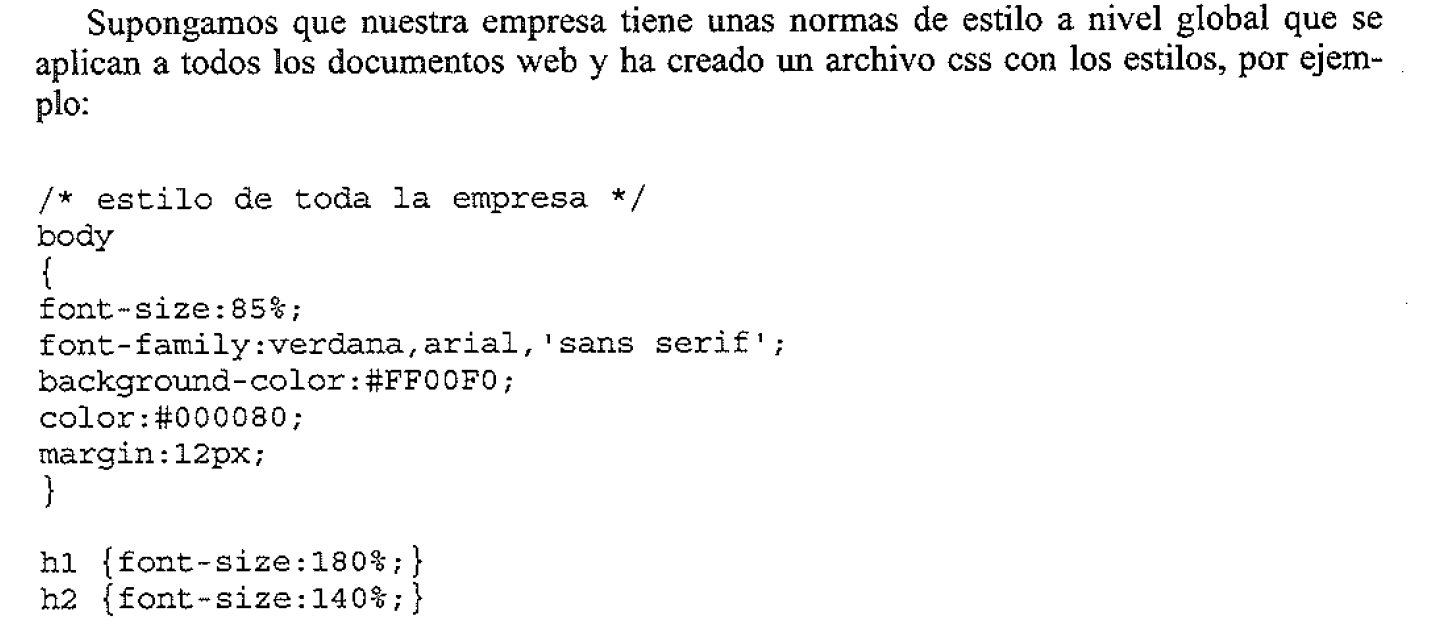
[2.4.4. La herencia](#_Toc185832926).

Una hoja de estilo asociada con una o más páginas Web resulta de mucha utilidad, pero también resulta insuficiente en ciertos casos, como un sitio muy grande o manejado por varias personas, en los cuales una sola hoja de estilo no alcanzaría a cubrir todas las necesidades de diseño.

Esta limitación está cubierta mediante la posibilidad de vincular hojas de estilo y de crear una jerarquía de hojas de estilo relacionadas. Este proceso se llama **CASCADA** porque los estilos definidos en cada hoja se van sumando para finalmente ser volcados a la página.

Pensemos, por ejemplo, en el caso del sitio de una organización con varios departamentos y otros tantos diseñadores encargados del mantenimiento de las páginas. El modo de asegurar la uniformidad gráfica del sitio completo es utilizar una hoja de estilo común donde estén definidos los aspectos básicos del diseño (fuentes, colores, etc.). Luego cada departamento puede tener, además, su propia hoja de estilo donde se definirán los aspectos específicos de ese departamento. Estas hojas de estilo particulares no necesitan reproducir aquella hoja común, sino que bastará con importarla haciendo que los estilos definidos en una y otras actúen conjuntamente. Cualquier cambio en la hoja de estilo común (una modificación en los colores institucionales, por ejemplo) se vería reflejado inmediatamente en todas las páginas vinculadas.

EJEMPLO:



**2.4.1. La regla @import. NO ES MUY RECOMENDABLE**

La regla **@import** permite importar hojas de estilo desde otras hojas de estilo. Se puede usar de las siguientes formas:

**@import** "hoja\_de\_estilo.css";

**@import url**("hoja\_de\_estilo.css");

La regla no tiene un valor predeterminado.

Se necesita punto y coma (;) en la sintaxis; si se omite, la hoja de estilos no se importará correctamente y se generará un mensaje de error. El parámetro **url()** es opcional porque siempre hay una dirección URL después de **@import**.

La regla **@import**, al igual que el elemento **link**, vincula una hoja de estilos externa a un documento. Esto ayuda al autor de páginas web a crear una apariencia coherente entre las distintas páginas HTML. Mientras que el elemento **link** especifica el nombre de la hoja de estilos que importará mediante la propiedad **href**, la regla **@import** indica la definición de la hoja de estilos dentro de un elemento **link** o **style**.

La regla **@import** se debe colocar en la parte superior de la hoja de estilos, antes de cualquier declaración. Aunque las instrucciones de la regla **@import** puedan aparecer en cualquier lugar dentro de la definición de la hoja de estilos, las reglas incluidas en la hoja de estilos de la regla **@import** se aplicarán al documento antes que cualquier otra regla definida para la hoja de estilos que la contiene. El orden de las reglas afecta a la representación.

Las reglas de la hoja de estilos que contengan una regla **@import** reemplazarán a las reglas de la hoja de estilos importada.

En los siguientes ejemplos se usa la regla **@import** para importar una hoja de estilos. Para que el ejemplo funcione, se debe reemplazar **URL** en el código de ejemplo con la dirección de una hoja de estilos:

[Copiar](javascript:CodeSnippet_CopyCode('CodeSnippetContainerCode_46289db4-7c58-430b-8267-04c5d2c0eee0');" \o "Copiar al portapapeles.)

<style type="text/css">

@import url("URL");

p {color:blue}

</style>

En el siguiente ejemplo, sin **url()**, el efecto es el mismo que el del ejemplo anterior:

[Copiar](javascript:CodeSnippet_CopyCode('CodeSnippetContainerCode_a44c70a9-f34c-493a-bb0b-5beb871c2e18');)

<style type="text/css">

@import "URL";

p {color:blue}

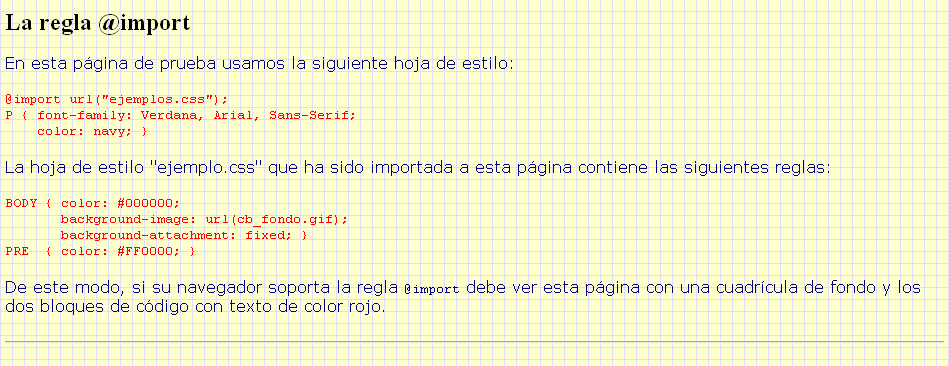
</style>

Esta declaración debe ir antes que cualquier otra regla en la hoja de estilo.

Ejemplo:ejem.html

|  |
| --- |
| <html>  <head>  <title>Untitled</title>  <style type="text/css">  **@import url**("ejemplo.css");  **p** { font-family: Verdana, Arial, Sans-Serif; color: navy; }  </style>  </head>  <body bgcolor="#FFFFFF">  <h2>La regla @import</h2>  <p class="normal">En esta p&aacute;gina de prueba usamos la siguiente hoja de estilo:</p>  <pre>@import url(&quot;ejemplos.css&quot;);  P { font-family: Verdana, Arial, Sans-Serif; color: navy; }</pre>  <p class="normal">La hoja de estilo &quot;ejemplo.css&quot; que ha sido importada a esta p&aacute;gina contiene las siguientes reglas:</p>  <pre>BODY { color: #000000;  background-image: url(cb\_fondo.gif);  background-attachment: fixed; }  PRE { color: #FF0000; }</pre>  <p>De este modo, si su navegador soporta la regla <code>@import</code> debe ver esta p&aacute;gina con una cuadr&iacute;cula de fondo y los dos bloques de c&oacute;digo con texto de color rojo.</p>  <hr>  </body>  </html> |
| ejemplo.css  BODY { color: #000000;  background-image: url(cb\_fondo.gif);  background-attachment: fixed; }  PRE { color: #FF0000; } |

De este modo, si su navegador soporta la regla @import debe ver esta página con una cuadrícula de fondo y los dos bloques de código con texto de color rojo.



Este mecanismo, que permite sumar los estilos de una CSS a otra, puede provocar que existan varias reglas con distintos valores aplicados a un mismo elemento. En esos casos, hay ciertas normas que determinan cuál de esos valores será el que se aplique finalmente al elemento. Es decir, se establece un orden de preferencias dentro de la cascada.

**2.4.2. La regla !important**

La regla **!important** está pensada para permitir una mejor accesibilidad a los documentos, otorgándoles a los usuarios con requerimiento especiales (fuentes grandes, combinaciones de colores, etc.) el control sobre la presentación.

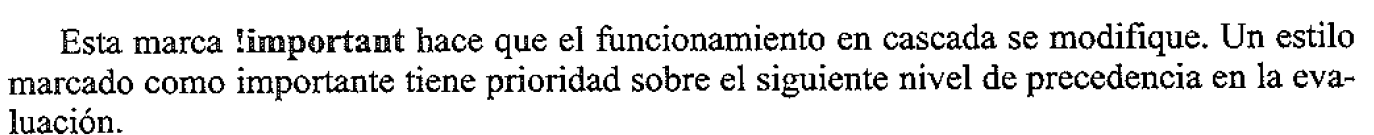
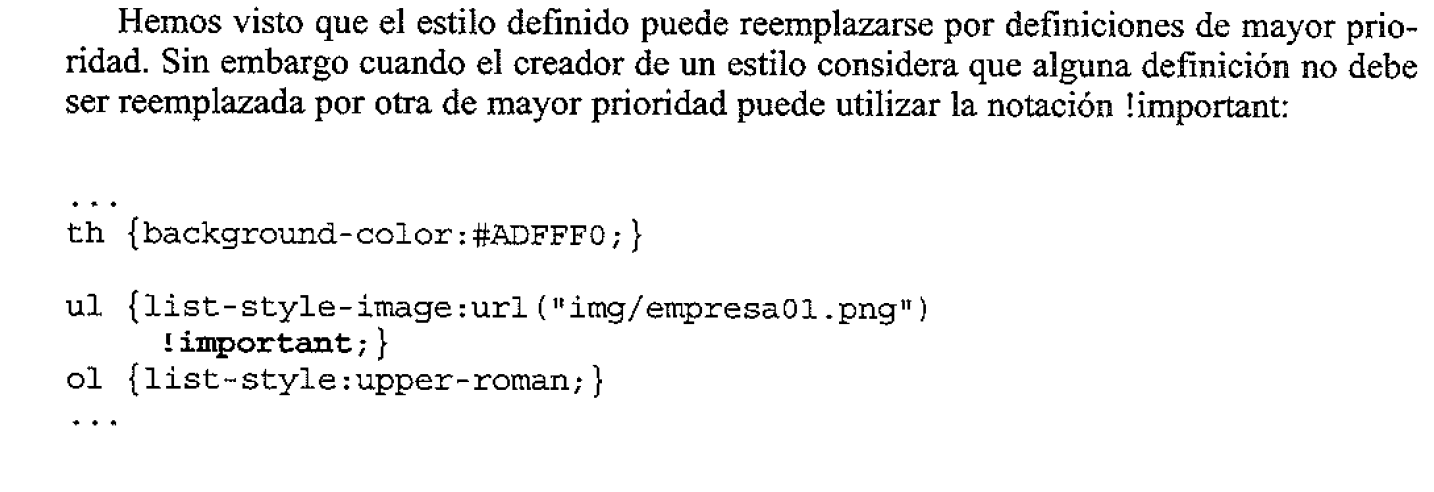
Ejemplo:

**p** { font-size: 14px **!important** }

Estas reglas tienen más fuerza que las declaraciones normales. Las hojas de estilo del autor y del usuario pueden contener la declaración **!important**, en cuyo caso la regla del usuario tiene mayor fuerza que la del autor.

Es importante destacar que, aunque las hojas de estilo del autor tienen preponderancia sobre las del usuario, con la regla **!important** esto se revierte y se le da mayor poder a las preferencias del usuario.

EJEMPLO:



**2.4.3. El orden de la cascada**

El concepto de cascada es la capacidad de definir los estilos a partir de diferentes orígenes, que se unifican para dar como resultado un estilo determinado.  
  
El origen de la definición de los estilos puede ser cualquiera de los siguientes, ordenados por prioridad de elección:

* Estilos definidos por el autor.
* Estilos de usuario.
* Estilos del agente de usuario. El navegador.
* **El autor (programador o diseñador)**

Los estilos definidos por el autor son los estilos que el autor incluye en el propio documento o a través de las hojas de estilo. Es la presentación elegida por el autor.  
Como hemos visto hasta ahora, puede hacerlo a través de una hoja externa o incrustada, incluso definiendo el estilo en línea.

* **El usuario**

Los estilos de usuario son los especificados por el usuario del documento a partir de un conjunto de hojas de estilo locales que se seleccionan en las opciones del navegador.  
Un usuario puede especificar su propia hoja de estilo. Esta posibilidad puede resultar de gran ayuda para aquellas personas con discapacidades visuales o, simplemente, para quienes deseen adaptar las páginas a sus preferencias. Cada navegador tiene su forma particular de proveer al usuario la posibilidad de especificar una hoja de estilo propia.

* **El navegador. Los estilos del agente usuario.**

Los estilos del agente usuario son los estilos aplicados de manera predeterminada cuando no hay otros estilos especificados para un elemento del documento. Suelen ser definiciones básicas, por ejemplo texto en color negro sobre fondo blanco, etc.  
El navegador también aplica una hoja de estilo predeterminada que presenta los elementos de la página de modo que satisfagan las expectativas generales de presentación del documento.

La prioridad de aplicación de los estilos sigue el orden especificado anteriormente, salvo que se utilice la palabra clave "!important", por ejemplo: div {color: red !important;}

Las hojas de estilo con estos tres orígenes actúan conjuntamente y el orden de la cascada determina, en el caso en que existan reglas incompatibles, cuál de ellas tiene preponderancia.

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

1. La primera disposición se hace **por el origen**

Las reglas de la hoja de estilo del autor tienen más prioridad que las del usuario y éstas, a su vez, prevalecen sobre las del navegador.

1. La segunda disposición es **por especificidad**

Los selectores más específicos tienen mayor prioridad que los selectores generales.

1. Finalmente, por el **orden especificado (ubicación)**

Si dos reglas tienen la misma fuerza, origen y especificidad, la última en ser especificada es la que prevalece. Las reglas en las hojas de estilo importadas se considera que están antes que cualquier regla en la propia hoja de estilo.

**2.4.3.1. Orden de cascada por especificidad**

Cuando se aplica más de una regla a un mismo elemento, existe un conflicto de especificidad.

Por ejemplo, si una regla indica que todos los elementos <H1> deben ser azules y otra indica que todos los elementos <H1> de la clase “inicial” deben ser rojos, ¿qué regla de estilo prevalecerá?. En estos casos, la ley de la especificidad indica que **cuanto más concreto es el selector, más valor tiene la regla**. En este caso la segunda regla anulará a la primera y será la que resulte aplicada finalmente.

En este sentido, un selector con el atributo “id” prevalecerá sobre un selector con el atributo “class” y éste, sobre cualquier selector simple o sin atributos.

Finalmente los selectores simples prevalecerán sobre cualquier regla heredada, consideradas como las más generales de todas.

Si tenemos dos definiciones como las siguientes:  
 div {color: blue;}  
 div p {color: green;}  
  
para una etiqueta <p>, la segunda definición es más específica que la primera porque exige que, además, se encuentre dentro de una etiqueta <div>. (?).  
  
W3C suministra una metodología para calcular el valor o nivel de especifidad de una declaración:

1. Contar la cantidad de identificadores de atributos en el selector y asignar ese valor a las centenas del resultado.
2. Contar la cantidad de otros atributos y pseudo clases en el selector y asignar ese valor a las decenas del resultado.
3. Contar la cantidad de nombres de elementos y pseudo elementos en el selector y asignar ese valor a las unidades del resultado.

**2.4.3.2. Orden de cascada por ubicación**

Las reglas que aparecen en último lugar tienen más peso.

Las reglas que se aplican localmente a un elemento HTML se consideran que aparecen después de las definidas en una hoja de estilo interna, y por lo tanto, tienen más relevancia que éstas.

Como resumen en cuanto a la precedencia en la aplicación de las reglas de estilo, debemos decir que, cuando dos reglas afectan al mismo elemento:

**Prevalece la más específica y a idéntica especificidad, la última será la aplicada**

Ejemplo resumen de resolución de conflictos en la aplicación de reglas:

|  |  |
| --- | --- |
| <html>  <head>  <title>Resolucion de conflictos en reglas de estilos</title>  <style type="text/css">  **p** { color: #0000FF; }  **p.grupo** { color: #00FF00; }  **p#tercero** { color: #FF0000; }  </style>  </head>  <body>  <p>Este párrafo no pertenece a ninguna clase ni tiene identificador. Quedará escrito en azul</p>  <p class="grupo">Este párrafo pertenece a la clase "grupo" pero no tiene identificador. Quedará escrito en verde</p>  <p class="grupo" id="tercero">Este párrafo pertenece a la clase "grupo" y tiene el identificador "tercero".Quedará escrito en rojo</p>    <p style="color:#999999">Este párrafo se escribirá en gris, siguiendo el estilo definido localmente.</p>  <p >Como podemos comprobar los conflictos se han resuelto por el principio de especificidad.</p>  </body>  </html> |  |

Prioridad en etiquetas link

Cuando se utiliza más de una hoja independiente, con código que se aplica a un mismo elemento, el archivo especificado en último lugar tiene prioridad sobre el anterior.  
  
Además, si en un fichero externo se define una propiedad individual de un elemento que no está redefinida en ninguno de los demás ficheros externos, en el documento HTML se aplicará ese valor siguiendo el mecanismo de herencia.  
  
La determinación de la prioridad de aplicación es la siguiente:

1. Se ordenan las definiciones por peso (importancia) y origen.
2. Los estilos de autor reemplazan a (tienen prioridad sobre) los estilos de usuario.
3. Los estilos de usuario reemplazan los estilos del agente de usuario.
4. Las definiciones "!important! reemplazan a las definiciones normales.
5. Ordenar las definiciones por la especificidad del selector: los selectores más específicos reemplazan a los más generales.
6. Se ordenan las definiciones especificadas.
7. Si dos definiciones tienen el mismo peso, origen y especificidad, se elige la última que haya aparecido. Las definiciones de hoja de estilo importadas se consideran antes que las definiciones de la propia hoja de estilo

* **LA HOJA DE ESTILOS DEL NAVEGADOR ES LA PRIMERA QUE SE APLICA Y SE UTILIZA PARA ESTABLECER EL ESTILO INICIAL POR DEFECTO DE TODOS LOS ELEMENTOS. PODEMOS COMPROBAR SUS VALORES MEDIANTE MAS HERRAMIENTAS.**
* **LA HOJA DE ESTILOS DEL USUARIO SE CONFIGURA MEDIANTE ALGUNA OPCIÓN AVANZADA DEL NAVEGADOR DESDE CONFIGURACIÓN. EJEMPLO**

**2.4.4. La herencia**



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

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

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

Es necesario hacer algunas precisiones:

* No todas las propiedades se pueden heredar y para cada propiedad se define si ésta se hereda o no.

Las propiedades que **NO SE HEREDAN** son:

* + - Fondo (background)
    - Borde (border)
    - Márgenes (margin)
    - Relleno (padding)
    - Dimensiones (width, height)
    - Posición (top, left)

Las propiedades que **SI SE HEREDAN** son:

* + - Color del texto (color)
    - Formato del texto (font, text-align, text-decoration)

Ejemplo:

Si se define el estilo sobre <H1> en color azul con un borde rojo, cualquier elemento dentro de <H1> será también azul dado que la propiedad color se hereda; sin embargo, no tendrán el borde rojo porque la propiedad “border” no se transmite. Para conseguirlo, se deberá forzar la herencia de la propiedad (como veremos a continuación).

Ejemplo: Cuando se establece el valor de una propiedad CSS en un elemento, sus elementos descendientes heredan de forma automática el valor de esa propiedad.

<html>

<head>

<title>Ejemplo de herencia de estilos</title>

<style type="text/css">

body { color: blue; }

</style>

</head>

<body>

<h1>Titular de la página</h1>

<p>Un párrafo de texto no muy largo.</p>

</body>

</html>

En el ejemplo anterior, el selector body solamente establece el color de la letra para el elemento <body>. No obstante, la propiedad color es una de las que se heredan de forma automática, por lo que todos los elementos descendientes de <body> muestran ese mismo color de letra. Por tanto, establecer el color de la letra en el elemento <body> de la página implica cambiar el color de letra de todos los elementos de la página.

Aunque la herencia de estilos se aplica automáticamente, se puede anular su efecto estableciendo de forma explícita otro valor para la propiedad que se hereda

Ejemplo

<html >

<head>

<meta charset=”utf-8” />

<title>Ejemplo de herencia de estilos</title>

<style type="text/css">

body { font-family: Arial; color: black; }

h1 { font-family: Verdana; }

p { color: red; }

</style>

</head>

<body>

<h1>Titular de la página</h1>

<p>Un párrafo de texto no muy largo.</p>

</body>

</html>

En el ejemplo anterior, se establece en primer lugar el color y tipo de letra del elemento <body>, por lo que todos los elementos de la página se mostrarían con ese mismo color y tipo de letra. No obstante, las otras reglas CSS modifican alguno de los estilos heredados.

De esta forma, los elementos <h1> de la página se muestran con el tipo de letra Verdana establecido por el selector h1 y se muestran de color negro que es el valor heredado del elemento <body>. Igualmente, los elementos <p> de la página se muestran del color rojo establecido por el selector p y con un tipo de letra Arial heredado del elemento <body>.

* El valor **inherit** puede aplicarse a cualquier propiedad de los elementos. Este valor puede usarse para reforzar explícitamente la herencia de una propiedad o para lograr que un elemento herede de su padre una propiedad que de otro modo no sería heredada.

Ejemplo:

{border:inherit}

* Cuando se asigna una propiedad a un elemento, el valor especificado reemplaza al valor heredado.
* Los elementos heredan los valores computados del padre, no los valores especificados
  1. **Valor especificado**

Todas las propiedades tienen siempre un valor asignado. Puede ser el especificado en la CSS o, en su defecto, el valor heredado. En ausencia de los dos anteriores, se usa el valor inicial (el valor predeterminado para cada propiedad).

* 1. **Valor computado**

Cuando se especifica un valor relativo es necesario hace un cálculo para obtener su valor absoluto.

Por ejemplo, para resolver P {font-size: 125%} se debe tomar el valor del elemento padre de <P> y aplicarle un porcentaje de 125% para obtener el tamaño definitivo del texto. Supongamos que <BODY> es el padre de <P> y tiene un tamaño de fuente de 10px, entonces <P> tendrá un tamaño de fuente de 12,5px (10x125%=12,5). Los elementos que sean hijos de <P> no van a heredar el valor especificado (125%) sino el valor computado (12,5px).

* 1. **Valor real**

Puede suceder que el valor resultante no pueda ser utilizado por el navegador debido a limitaciones técnicas y deba ser aproximado a otro valor. En el ejemplo anterior, en la pantalla no sería posible mostrar una fuente de 12,5px, por lo que el navegador la ajustará a 12px ó 13px. Este valor no resulta de mucho interés para el usuario sino para el programador.

**EJEMPLO:**

Como hemos visto hay un conjunto de marcas HTML que forman en sí un árbol que en su raíz, podemos identificar la marca body del cual se desprenden otras marcas contenidas en esta sección, como podrían ser las marcas h1, h2, h3, h4, h5, h6, p, div luego estas en su interior contienen otras marcas HTML como podrían ser em, b, i, pre etc.

Con un ejemplo veremos el resultado de la herencia de propiedades entre las marchas HTML: pagina\_1.html

<html>

<head>

<title>Problema</title>

<style type="text/css">

body {

color:#0000ff;

font-family:verdana;

}

</style>

</head>

<body>

<h1>Este es un título de nivel 1 y con la marca 'em' la palabra:

<em>Hola</em></h1>

<p>Todo este párrafo debe ser de color azul ya que lo hereda de la

marca body.</p>

<a href="pagina2.html">Siguiente ejemplo</a>

</body>

</html>

En este ejemplo hemos definido la siguiente regla para la marca body:

body {

color:#0000ff;

font-family:verdana;

}

Inicializamos la propiedad color con el valor de azul y la fuente de tipo verdana. Con esto todas las marcas contenidas en el body que no redefinan estas dos propiedades recibirán los valores aquí definidos. En este ejemplo la cabecera de primer nivel es decir h1, el párrafo y el hipervínculo tienen como color el azul y la fuente es de tipo verdana.   
Ahora bien en muchas situaciones podemos redefinir propiedades para marcas contenidas, veamos como podemos hacer que el texto contenido en las marcas **em y p** aparezcan de color distinto: pagina2.html

<html>

<head>

<title>Problema</title>

<style type="text/css">

body {

color:#0000ff;

font-family:verdana;

}

em {

color:#008800;

}

p {

color:#999999;

}

</style>

</head>

<body>

</body>

<h1>Este es un título de nivel 1 y con la marca 'em' la palabra:

<em>Hola</em></h1>

<p>Todo este párrafo debe ser de color gris ya que lo redefine la

marca p y no lo hereda de la marca body.</p>

</html>

Ahora hemos definido tres reglas, la primera igual que el problema anterior, define la propiedad color en azul y la fuente de tipo verdana para la marca body:

body {

color:#0000ff;

font-family:verdana;

}

La segunda regla define la propiedad color en verde para la marca em, con esto no heredará el color azul de la marca body (que es la marca padre):

em {

color:#008800;

}

Algo similar hacemos con la marca p para indicar que sea de color gris:

p {

color:#999999;

}

Pero podemos ver que todas las marcas heredan la fuente verdana ya que ninguna marca la sobreescribe.

EJERCICIO

Copiad el siguiente documento HTML en un fichero nuevo del editor de textos y guardadlo como **inherit.html**.

<html >

<head>

<title>Herencia</title>

</head>

<body>

<h1>Título</h1>

<p>Un párrafo de texto.</p>

</body>

</html>

Cread un nuevo fichero vacío con el editor de textos, copiad dentro la regla CSS que se muestra a continuación y guardad el fichero como estilo.css en la misma ubicación que el fichero HTML.

html { font: 75% Verdana,sans-serif;}

Enlazad la hoja de estilos en el documento HTML insertando la línea siguiente antes del tag </head>.

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

Guardad el fichero HTML modificado y recargad el documento en el navegador. La fuente pasará de ser la predeterminada por el navegador (normalmente Times o Times New Roman) a ser Verdana. Si no tenéis Verdana instalada en el ordenador, el texto se mostrará con la fuente Sans Serif especificada por defecto en la configuración del navegador. Además, el texto se verá un 25% más pequeño que en la versión sin estilo.

La regla CSS que hemos especificado se aplica únicamente al elemento html. No hemos especificado ninguna regla para los títulos o los párrafos, pero ahora todo el texto se muestra en Verdana al 75% del tamaño por defecto. ¿Por qué? Por la herencia.

La propiedad font es una propiedad abreviada que establece toda una serie de propiedades relacionadas con las fuentes. Sólo hemos especificado dos, el tamaño de la fuente y la familia de fuentes, pero esta regla equivale a lo siguiente:

html {

font-style: normal;

font-variant: normal;

font-weight: normal;

font-size: 75%;

line-height: normal;

font-family: Verdana,sans-serif;

}

Todas estas propiedades se heredan, de manera que el elemento body las heredará del elemento html y después las transmitirá a sus hijos: el título y el párrafo.

Pero, hay algo que no acaba de quedar claro respecto a la herencia del tamaño de la fuente. El tamaño de la fuente del elemento html se establece en 75%, pero ¿75% de qué? ¿Y el tamaño de la fuente de body, no debería ser el 75% del tamaño de la fuente de su padre, y los tamaños de las fuentes del título y del párrafo, deberían ser el 75% del tamaño del elemento body?

El valor que se hereda no es el valor especificado, es decir, el valor que escribimos en la hoja de estilo, sino algo que se llama *el valor computado*. El valor computado es, en el caso del tamaño de la fuente, un valor absoluto medido en píxeles. Para el elemento html, que no tiene un elemento padre del cual heredar, un porcentaje del valor de tamaño de fuente se asocia al tamaño de fuente predeterminada del navegador. La mayoría de los navegadores actuales tienen un tamaño de fuente predeterminada de 16 píxeles. El 75% de 16 son 12, de manera que el valor computado del tamaño de la fuente del elemento html será probablemente 12 píxeles. Éste es el valor que hereda body y que se transmite al título y al párrafo.

(El tamaño de la fuente del título es mayor porque el navegador aplica algunas normas de estilo integradas propias. Podéis ver el tema de la cascada a continuación.)

Añadid dos declaraciones más a la regla de la hoja de estilo de CSS:

html {

font: 75% Verdana,sans-serif; background-color: blue; color: white;

}

Guardad el fichero CSS y recargad el documento en el navegador.

Ahora el fondo es de color azul fuerte y todo el texto es blanco. La regla se aplica al elemento html: el documento entero, cuyo fondo será azul. El elemento body hereda el color blanco de primer plano y se transmite a todos los hijos de body: en este caso, el título y el párrafo. Éstos no heredan el fondo, pero el fondo se establecerá en el valor por defecto de transparent, de manera que el resultado visual final será texto blanco sobre fondo azul.

Añadid otra regla nueva a la hoja de estilo y guardad y recargad el documento.

h1 { font-size: 300%;}

Esta regla establece el tamaño de la fuente del título. El porcentaje se aplica al tamaño de fuente heredada (el 75% de la predeterminada por el navegador, que suponemos que es 12 píxeles), de manera que el tamaño del título será el 300% de 12 píxeles, es decir: 36 píxeles.

**. Forzar la herencia**

### Mediante la palabra clave inherit (heredar) puede forzarse la herencia incluso para propiedades que no se heredan normalmente. Sin embargo, se debe utilizar con mucho cuidado porque Microsoft Internet Explorer (hasta la versión 7 incluida) no es compatible con esta palabra clave.

La regla siguiente hace que todos los párrafos hereden todas las propiedades de fondo de sus padres:

p { background: inherit;}

Con las propiedades abreviadas se puede utilizar inherit en vez de los valores normales. Se debe utilizar la versión abreviada o bien para todo o bien para nada en absoluto. En la versión no abreviada no se pueden especificar algunos valores y utilizar inherit para otros porque los valores pueden darse en cualquier orden y no hay manera de especificar qué valores queremos heredar.

Forzar la herencia no es algo que haya que hacer a menudo. Puede ser útil para "deshacer" una declaración de una regla conflictiva o para no tener que introducir los datos de determinados valores directamente en el código fuente. Un ejemplo de esto sería el típico menú de navegación:

<ul id="nav">

<li><a href="/">Inicio</a></li>

<li><a href="/news/">Noticias</a></li>

<li><a href="/products/">Productos</a></li>

<li><a href="/services/">Servicios</a></li>

<li><a href="/about/">Sobre nosotros</a></li>

</ul>

Para mostrar esta lista de enlaces como menú horizontal, podéis utilizar el CSS siguiente:

#nav { background: blue; color: white; margin: 0; padding: 0;}

#nav li { display: inline; margin: 0; padding: 0 0.5em; border-right: 1px solid;}

#nav li a { color: inherit; text-decoration: none;}

En este caso, el color de fondo de toda la lista se establece en azul en la regla de #nav. Así, también se establece el color de primer plano como blanco y todos los elementos de la lista y todos los enlaces heredan el mismo. La regla de los elementos de la lista establece un límite a la derecha, pero no especifica el color del margen, lo que significa que utilizará el color de primer plano heredado (el blanco). Para los enlaces, hemos utilizado color:inherit para forzar la herencia y anular el color de los enlaces predeterminado del navegador.

Lógicamente, también podría haber especificado el blanco como color del margen y del texto de los enlaces, pero lo mejor del hecho de dejar que lo haga la herencia es que, si más adelante decidimos cambiar los colores, sólo deberemos hacer un cambio en este punto.