结构和层叠

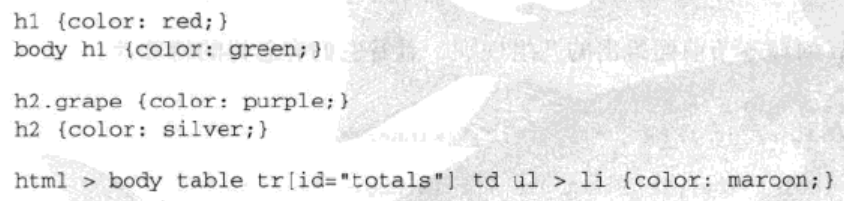
1. 继承(Inheritance)

从一个元素向其后代元素传递属性值。

确定应当向一个元素应用哪些值时，用户代理不仅要考虑继承，还有考虑声明的特殊性，另外需要考虑声明本身的来源。这个过程就叫做层叠(cascade)

**特殊性-继承-层叠**

1. 特殊性



对于每个选择器的特殊性(specificity)。对于每个规则，用户代理会计算选择器的特殊性，并将这个特殊性附加到规则中的各个声明。如果一个元素有两个或多个冲突的属性声明，那么有最高特殊性的声明就会胜出。

* 对于选择器中给定的各个ID属性值，加0,1,0,0.
* 对于选择器中给定的各个类属性值、属性选择或伪类，加0，0，1，0。
* 对于选择器中给定的各个元素和伪元素，加0，0，0，1。
* **结合符【空格 > +】**和通配选择器对特殊性没有任何贡献（即没有特殊性）。也就是其特殊性为0，0，0，0

例子:



特殊性值：

0，0，0，1

0，0，0，2

0，0，1，0

0，0，1，0

0，0，2，2，

0，1，0，0

0，1，1，1

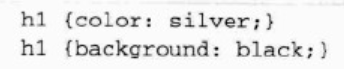
1. 声明和特殊性

一旦确定了一个选择器的特殊性，这个值将授予其所有相关声明。

**由于特殊性的缘故，用户代理必须相应地处理这个规则，将其‘解组’为单独的规则。**



会被解组为：



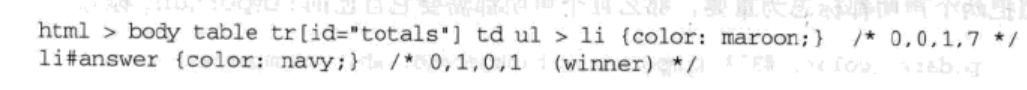
1. 通配选择器特殊性

**通配选择器的特殊性值为：0，0，0，0**



**结合符根本根本没有特殊性，甚至连0特殊性都没有。**

1. ID 和属性选择的特殊性



1. 内联样式特殊性

第一个0是为内联样式声明保留的，它比所有其他声明的特殊性都高。

**每一个内敛的声明的特殊性都是1，0，0，0.**

1. 重要性

**允许在这些声明的结束分号之前插入!important来标志。**

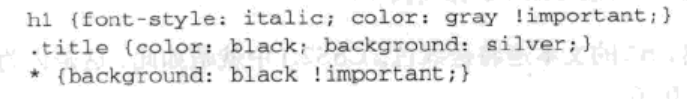


**必须将!important标志放在声明的最后。**

备注：

标志为!important的声明并没有特殊的特殊性值，不过要与非重要声明分开考虑。

实际上，所有!important声明会分组在一起，重要声明的特殊性冲突会在重要声明内部解决，而不会与非重要声明相混。类似的，我们认为非重要的声明也会归为一组，使用特殊来在内部解决。如果一个重要声明和一个非重要声明冲突，胜出的总是重要声明。



<h1 class="title" style="background: blue;">NightWing</h1>

!important 非重要

h1{font-style:italic}

.title{color:black;background:silver;}

h1{color:gray }  
\*{background:back}

引用的样式：

h1{color:gray; font-style:italic; background:back;}

即使使用内联元素，!important也会其作用。

1. 继承

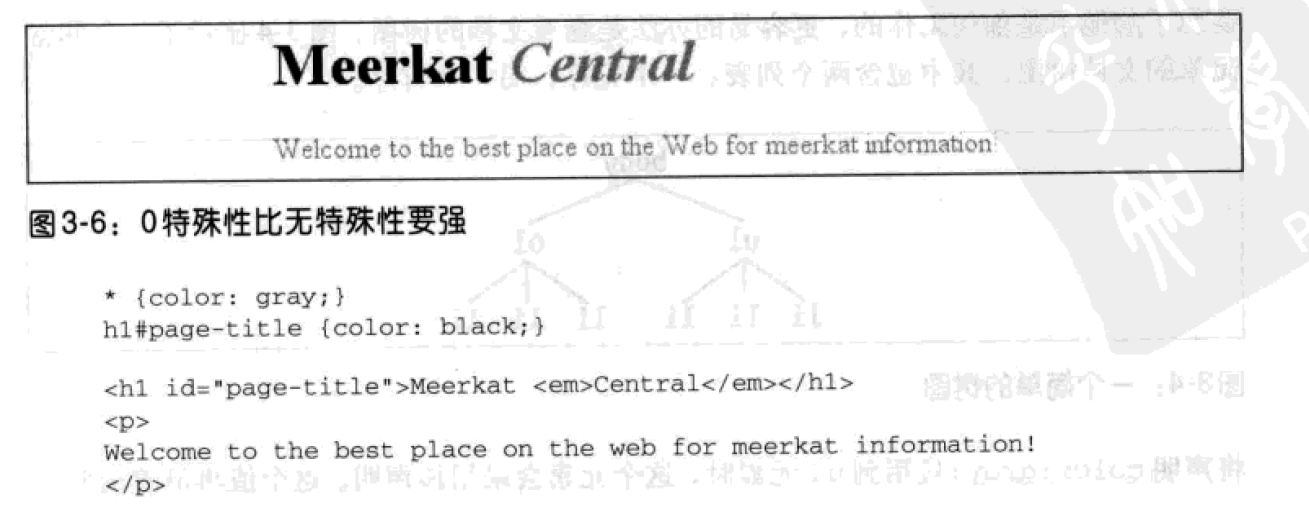
基于继承机制，样式不仅应用到指定的元素，还会应用到它的后代元素。

首先，有些属性不能继承，这往往归因于一个简单的常识。

例如，属性border(用于设置元素的边框)就不会被继承。

一般地，大多数框模型属性（包括外边距，内边距，背景和边框）都不能继承。

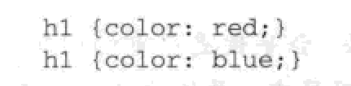
继承的值根本没有特殊性，甚至连0特殊性都没有。



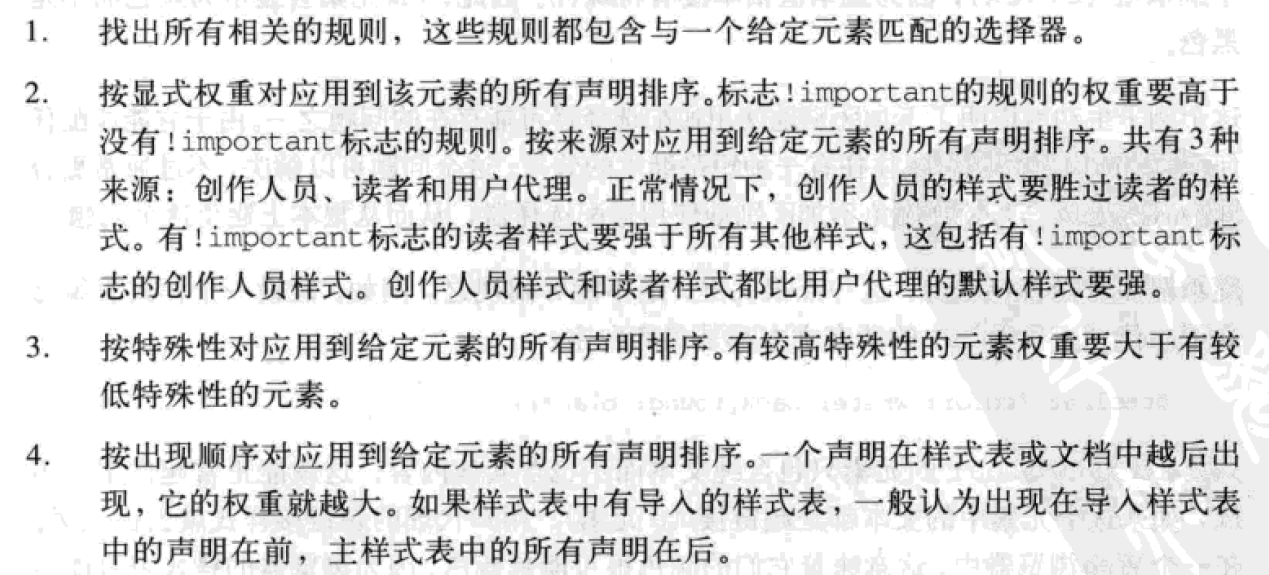
因为通配选择器适用于所有元素，而且有0特殊性，其颜色声明指定的值gray要优先于继承值（black）,因为继承值根本没有特殊值。

1. 层叠

如果特殊值相等的两个规则同时应用到同一个元素怎么办？

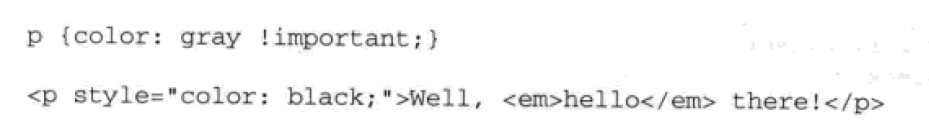


根据第二条规则，如果两个样式规则应用到同一个元素，而且



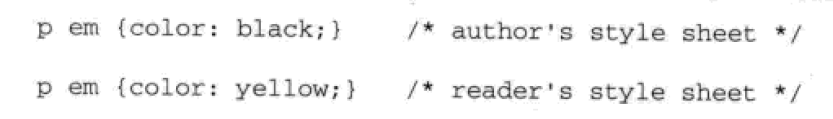
1. 按权重和来源排序

根据第二条规则，如果两个样式规则应用到同一个元素，而且其中一个规则有！important标识，这个重要规则将胜出：

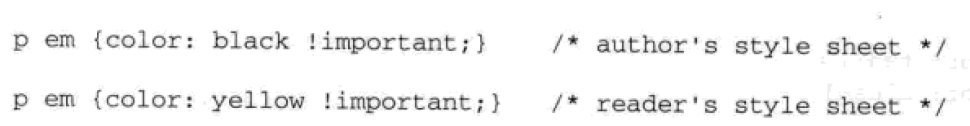


因为有!important标识的规则则胜出，所以段落为灰色，em元素继承这个灰色。

如果一个元素与创作人员样式表中的正常权重样式匹配，另外还与读者样式表中的正常权重样式匹配，则会使用创作人员的样式。



这个情况下，段落中强调的文本为黑色，因为有正常权重的创作人员样式要优先于正常权重的读者样式。不过，如果两个规则都标有!important，这种情况下就不同了：

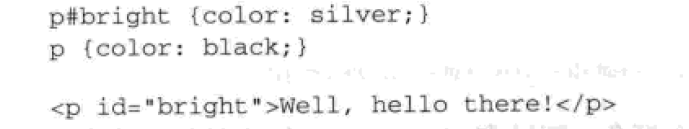


现在段落中的强调文本将是黄色而不是黑色。

在声明权重方面要考虑5级，权重由大到小的顺序依次为：

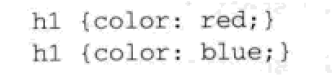
1. 读者的重要申明
2. 创作人员的重要声明
3. 创作人员的正常申明
4. 读者的正常申明
5. 用户代理声明（如web浏览器）
6. 按特殊性排序

根据规则3，如果向一个元素应用多个彼此冲突的声明，而且它们的权重相同，则按特殊性排序，最特殊的声明最优先

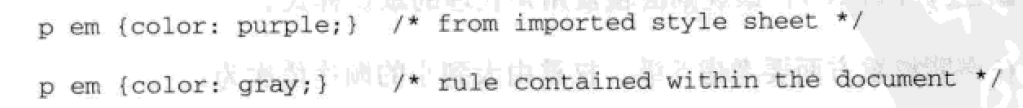


1. 按顺序排序

根据第4规则，如果两个规则的权重，来源和特殊性完全相同，那么在样式表中后出现的一个会胜出，



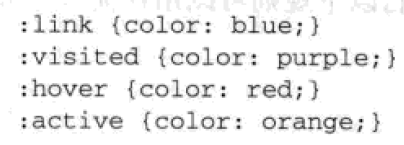
因为指定blue的规则在样式表中较后出现，所以文档中所有h1元素的color值将是blue.



文档中的样式表，会胜出导入的规则。

根据层叠的第4条规则，认为元素style属性中指定的样式位于文档样式表的最后，即放在所有其他规则的后面。

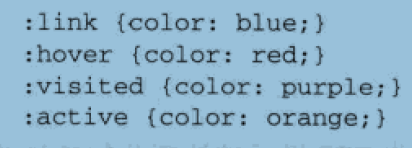
正由于这个按顺序排序，所以才有通常推荐的链接样式顺序，一般建议按link-visited-hover-active(LVHA)的顺序声明链接样式



这些选择器的特殊性都是一样的：0，0，1，0，

因为它们都有相同的权重，来源和特殊性，因此与元素匹配的最后一个选择器才会胜出。按照**LVHA**顺序，:active会胜出，这可能正是创作人员所期望的。

（link:未访问）（vistited:已访问）（hover:悬浮）（active：正访问）（这样都有效）



只有未访问的有悬浮样式，已访问的链接没有这样样式

有时候所有规则视图试图设置同一个属性，就会出现这种情况，如果各个规则未不同的属性设置，那么顺序就没关系了。



