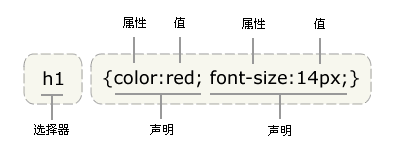
1. CSS基础
   1. 基础语法

CSS规则由两个主要部分构成：选择器，一条或多条声明，如下图所示



*☛ 声明需要使用花括号包围*

代表颜色的值的几种写法：英文单词（red）、十六进制颜色值（#ff0000）、CSS缩写（#f00）、RGB（255,0,0）（100%，0%，0%）

值为若干单词：需要使用引号，如“sans serif”

多重声明时，需要使用分号隔开

CSS对大小写不敏感，除非一个情况：与HTML一起时，class和id名称大小写敏感

* 1. 高级语法
     1. 选择器的分组

相同样式的多个元素作为一个分组，将多个元素用逗号隔开

将选择器放在规则左边，右边为样式，这样定义一个规则。逗号告诉浏览器，规则包含几个不同的选择器

例如：

h2, p {color:gray;}

在这里，有两个选择器h2，p，它们有相同的样式{color：gray}

通配符选择器

通配选择器（universal selector），显示为一个星号（\*）。该选择器可以与任何元素匹配，就像是一个通配符。

\* {color:red;}

* + 1. 声明分组

同一个元素声明多个样式，将多个样式用分号隔开，如下

未分组：

h1 {font: 28px Verdana;}

h1 {color: blue;}

h1 {background: red;}

使用声明分组后

h1 {font: 28px Verdana; color: white; background: black;}

☛注意：对声明分组，一定要在各个声明的最后使用分号，这很重要。浏览器会忽略样式表中的空白符。

* + 1. 层叠 Casecading

层叠性值的是样式的优先级，

1. 层叠次序

浏览器显示元素时的顺序

元素内嵌样式＞文档内嵌样式＞外部样式表＞用户样式＞浏览器样式

找出一个给定的元素的所有声明后，先按显示权重和来源进行排序，如果相同，咋比较特殊性，若再相同，则比较顺序。

1. 调整层叠次序 ！important

使用！important语句可以把样式属性标记为终于，这样不管该样式属性定义在什么地方，浏览器都会予以优先考虑

p{

color:#fff !important;

background:#ccc;

}

1. 同级样式冲突specificity

如果有两条定义于同一层次的样式都应用于同一个元素，且都包含有浏览器要求的CSS属性值，此时就会根据**专一程度**和**定义次序**来解决冲突

1. 专一程度

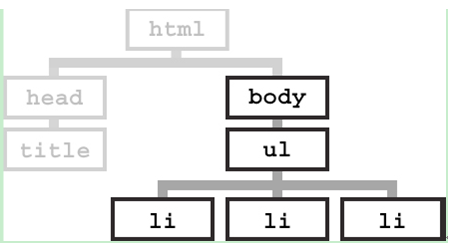
选择器的特殊性

1. 对于元素内嵌样式，特殊性首位加1，即1-0-0-0
2. 选择器中的id的数目，加0-1-0-0，有多少个id，则第2位+
3. 选择器中的类属性值、属性选择及伪类的数目，加0-0-1-0，第3位+
4. 选择器中的元素和伪元素的数目，加0-0-0-1，第4位+
5. 通配符对特殊性没有任何贡献，即为0-0-0-0
6. 结合法没有特殊性，连0特殊性也没有
7. 继承的css完全没有特殊性，连0也没有
8. 定义次序

如果同一个样式属性出现在具体程度相同的几条样式中，那么浏览器会根据其位置的先后顺序选择所用的值 : **后来居上原则**

* + 1. 继承及其问题

首先理清文档树（HTML DOM），如图，文档树与家族树类似，也有祖先、后代、父亲、孩子和兄弟。



CSS的继承就是通过对父亲的CSS样式设置，其孩子可以继承父亲的CSS样式设置

1. 并不是所有的CSS类都会被子类继承

与元素外观（文字颜色、字体等）相关的样式可以被继承；与元素在页面上的布局相关的样式不会被继承。CSS样式表属性可以被继承的有如下：

|  |  |
| --- | --- |
| 文本相关 | font-family，font-size，font-style，font-variant，font-weight，font，  letter-spacing，  line-height，  text-align，text-indent，texttransform，  word-spacing |
| 列表属性 | list-style-image，list-style-position，list-style-type，list-style |
| 其他 | azimuth  border-collapse，border-spacing  caption-side，color，cursor  direction，elevation，empty-cells  orphans，pitch-range，pitch，quotes，richness  speak-header，speaknumeral，speak-punctuation，speak，speechrate，stress  visibility，voice-family，volume，  whitespace，widows |

在样式中使用inherit可以实现强制继承，明确指示浏览器在该属性上使用父元素的样式值

p {

color:#fff;

border:medium solid #000;

}

span{

border:inherit; /\*border:medium solid #000;\*/

}

Font-size：子类继承的不是实际值，而是计算后的值

1. 如果子类未指定值，则直接继承父类计算后的值
2. 如果子类有指定值，则子类计算后的值=父类计算后的值\*指定值

举例:

body { font-size: 85%; }

h1 { font-size: 200%; }

h2 { font-size: 150%; }

浏览棋默认字体大小16px,而body定义了字体大小(16px X 85% = 13.6px). 如果子元素没有指定字体大小13.6px这个值将被子元素继承



1. 并不是所有的浏览器都支持继承

当浏览器不支持继承特性时，可以分级分组来编写组选择器，这样做有点冗余，但是避开了浏览器的不支持

如

body {

font-family: Verdana, sans-serif;

}

p, td, ul, ol, li, dl, dt, dd {

font-family: Verdana, sans-serif;

}

1. 特性值

样式表中的**特性值**描述了不同规则的相对权重，它的基本规则是：

◆统计选择符中的ID属性个数。

◆统计选择符中的CLASS属性个数。

◆统计选择符中的HTML标记名格式。

最后，按正确的顺序写出三个数字，不要加空格或逗号，得到一个三位数。(注意，你需要将数字转换成一个以三个数字结尾的更大的数)。相应于选择符的最终数字列表可以很容易确定较高数字特性凌驾于较低数字的。

以下是一个按特性分类的选择符的列表：

*H1{color:blue;}　    特性值为：1*

*PEM{color:purple;} 特性值为：2*

*.apple{red;}　  特性值为：10*

*P.bright{color:yellow;}    特性值为：11*

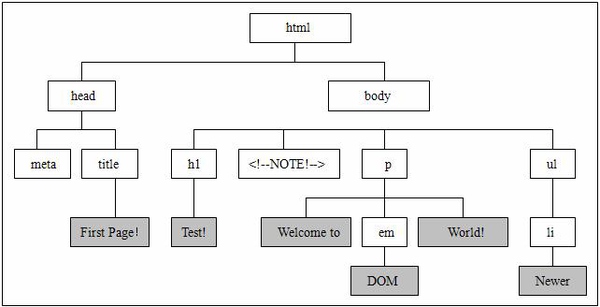
*P.brightEM.dark{color:brown;}　  特性值为：22*

*#id316{color:yellow}　   特性值为：100*

从上表我们可以看出#id316具有更高的特殊性，因而它有更高的权重。当有多个规则都能应用于同一个元素时，权重越高的样式将被优先采用。而继承属性的特性值为0；也就是说，**任何一条与css继承值冲突的属性值都会覆盖继承的属性值**！！！

* 1. 派生选择器

派生选择器又称为上下文选择器，即依赖于上下文来应用或避免应用某项规则，关于上下文的理解，可以参考下图



对其中的元素em，如果使用如声明：

em {color:red;}

则会将整个HTML的em元素都设置成红色，但是如果使用派生选择器来设置：

p em {color:green;}

则会仅会将p元素包含的em元素设置成绿色。因为em元素只有在上下文中存在p时才会有效。一句话：前面先写祖先元素，后面写子孙元素来定义声明

* 1. id选择器

id选择器为标有特定id的HTML元素指定特定样式，id选择器以井号“#”来定义

语法

#id名{标签属性：值；标签属性：值……标签属性：值}

引用方法：

<标签 CLASS=“ID名”>

#red {color:red;}

#green {color:green;}

<p id="red">这个段落是红色。</p>

<p id="green">这个段落是绿色。</p>

*☛ id属性只能在每个HTML文档中出现一次，其原因在于：id属性为一个元素分配一个唯一的名字，每个籽岷只能在被赋予的页面使用一次，相反地，class属性可以被一遍一遍地使用在页面中。*

*☛ id的规则：*

*id 值必须以字母或者下划线开始；不能以数字开始。虽然 W3C 验证不会捕获这个错误，但是 XML 解析器会的。同时，如果你将 id 与 JavaScript 在表单中配合使用，那么 id 名称和值必须是合法的 JavaScript 变量。空格和连字号，特别是连字号，是不被允许的。不仅如此，将下划线用于 class 或者 id 名都不是个好主意，这是由于在 CSS2.0（以及某些浏览器）中的限制*

1. id选择器和派生选择器

id选择器常用于建立派生选择器，如下的样式置灰应用于出现在id是sidebar的元素的段落。

#sidebar p {

font-style: italic;

text-align: right;

margin-top: 0.5em;

}

即使同一id在每个文档页面中仅能出现一次，但是这个id的选择器作为派生选择器也可以被多次使用

#sidebar p {

/styel

}

#sidebar h2 {

/styel

}

1. 单独选择器

#sidebar {

border: 1px dotted #000;

padding: 10px;

}

* 1. 类选择器

创建类可以创建同一个HTML标识的多种风格，在CSS中，类选择器，以一个点号（.）显示，如下

语法：

标签.类名{属性：值；属性：值；……；属性：值}

.类名{标签属性：值；标签属性：值；……标签属性：值}

引用方法：

<标签 CLASS=“类名”>

.center {text-align: center}

和id一样，类也可以用作派生选择器。如下类名为 fancy 的更大的元素内部的表格单元都会以灰色背景显示橙色文字。

.fancy td {

color: #f60;

background: #666;

}

类也可以基于它们的类而被选择，如下类名为 fancy 的表格单元将是带有灰色背景的橙色。

td.fancy {

color: #f60;

backgr

}

* 1. 属性选择器

[title]

{

color:red;

}

属性和值选择器

[title=W3School]

{

border:5px solid blue;

}

属性和值选择器-多个

[title~=hello] { color:red; }

* 1. CSS创建

样式表放置在不同的地方，产生的作用范围也不同

* + 1. 样式的来源
* 元素内嵌样式(将样式直接应用于元素，style属性),也叫内联样式
* 文档内嵌样式(创建可用于多个元素的样式，style元素),也叫内部样式
* 外部样式表样式(创建可用于多个HTML文档的样式，link元素),<link>应用的css后缀的文件
* 浏览器样式
* 用户自定义样式

根据以上可以划分为3类

一类:创作人员(即内联样式/内部样式/外部样式)

二类:读者(即用户自定义)

三类:用户代理(即浏览器)

* + 1. 文档内嵌样式

即内部样式，直接将CSS代码放置在html代码的<head>区中，这些定义的样式就应用到页面中了，样式表用<style>标记插入，如下例：

<head>

<style type="text/css">

<!—

p { font-family: "宋体"; font-size: 12px; line- height: 10pt; }

td { font-family: "宋体"; font-size: 12px; color: #000000}

a:link { color: #0000FF; text-decoration: none}

a:visited { color: #666666; text-decoration: none}

a:active {color:green;text-decoration: none;}

a:hover { color: #ff0000;background- color:#FF9900;text-decoration: underline }

-->

</style>

</head>

值得注意的是：有些低版本的浏览器不能识别style标记，会忽略style标记的内容，并把style标记的内容以文本直接显示到页面上，为了避免这种情况，使用了HTML注释的方式< !—注释 -- >隐藏内容，而不让它显示。

* + 1. 元素内嵌样式举例

<body>

<p style=”font-size:16px;background-color:#ccc;”>use Global Attribute ‘style’ to set style</p>

</body>

1. 放置与<span>……</span>标签之间

<span style=“background：yellow”>World Enterprises</span>

1. 现存的HTML标签内部

< h1 style=“color：blue；font-size：30pt”>Join the club ！</h1>

<ul style=“font-family：verdana”>

<li>Events

<li>Meetings

</ul>

* + 1. 外部样式表样式

外部样式表分为外部导入（import）和外部链接（link）

1. 外部导入

导入外部样式表是指在内部样式表的<style>里导入一个外部样式表（\*.css），导入时用@import，看下面这个实例：

<head>

<style type=”text/css”>

<!—

@import “mystyle.css”

其他样式表的声明

-->

</style>

</head>

此种做法的好处是：可以将所有的样式表放在一起，导入的样式表成为总样式表的一部分

坏处是：局限性较大，只有IE支持，且也会有很多问题

1. 外部链接

先编写一个\*.css文件，然后使用link元素，将这个css文件链接到所需的页面，如果文档想要链接多个样式表,则为每个样式表使用一个link元素即可.

可以把所有的页面都链接到同一css文件中，一旦这个css文件某一样式做了修改，则网站的所有页面的该元素样式都会随之改变。

如果不同的样式表使用了相同的选择器，则样式表的导入顺序很重要，此时使用的原则是**“后来居上”，**即使用后导入的样式表

/\*styl.css 文件\*/

p{

font-size: 16px;

background-color:#ccc;

}

<!—HTML文档-->

<head>

<link href=’style.css’ rel=’stylesheet’></link>

</head>

<body>

<p >use ‘link’ to set style</p>

</body>

1. @import

使用@import，可以将一个样式表导入到另外一个样式表中，每一个样式表使用一条@import语句

@import语句必须位于样式表顶端

此方法不建议使用，原因在于：

①**link阻断@import：**链接通过@import混合使用将使得IE并行下载中断

举例：使用link导入a.css和proxy.css，又通过proxy.css来导入b.css

HTML代码

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

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

proxy.css的代码

@import url("b.css");

此种用法，在IE中在下载a.css完成之前，不用开始下载b.css

②**link嵌套@import**：从一个链接的样式表中使用@import将使得所有浏览器的并行下载中断

举例：如下代码，浏览器必定先下载a.css并分析，然后通过@import再去下载b.css

HTML代码

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

a.css中

@import url("b.css");

③**link混合@import**：@import在IE中嵌入其他样式表，链接被阻挡

举例：如下代码，在IE中，会导致样式表逐个加载

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

<**style**>

@import url("b.css");

</**style**>

④**多个@import：**@import让IE中的资源下载次序被打乱

举例：以下代码，在IE中，会下下载最后的js脚本，在下载@import，那么如果js脚本有引入@import的样式，就会出现脚本先于样式被加载。

<**style**>

@import url("a.css");

@import url("b.css");

@import url("c.css");

@import url("d.css");

@import url("e.css");

@import url("f.css");

</**style**>

<**script** type="text/javascript" src="one.js"></**script**>

⑤必须将@import放在代码的顶端，否则它将不起作用

/\*combined.css文件 \*/

@import ‘style.css’

p{

color:#fff;

}

1. @charset

@charset语句是唯一的可以出现先@import语句之前的。@charset用于声明样式使用的编码。

如果样式表未声明字符编码，则浏览器将使用HTML文档声明的编码

如果HTML也未声明字符编码，那么浏览器将使用默认的UTF-8。

* + 1. 浏览器样式

是元素尚未设置样式时浏览器给它设置的默认样式。这些样式因浏览器而略有差异

* + 1. 用户自定义

大多数浏览器都允许用户定义自己的样式表（custom.css）

1. CSS样式
   1. 背景
      1. 背景色background-color

background-color 不能继承，其默认值是 transparent。也就是说，如果一个元素没有指定背景色，那么背景就是透明的。可以为所有元素设置背景色

p {background-color: gray; padding: 20px;}

* + 1. 背景图background-image

background-image 属性的默认值是 none，表示背景上没有放置任何图像。如果需要设置一个背景图像，必须为这个属性设置一个 URL 值：

body {background-image: url(/i/eg\_bg\_04.gif);}

* + 1. 背景重复background-repeat

属性值 repeat 导致图像在水平垂直方向上都平铺， repeat-x 和 repeat-y 分别导致图像只在水平或垂直方向上重复，no-repeat 则不允许图像在任何方向上平铺。

默认地，背景图像将从一个元素的左上角开始。如下：

body

{

background-image: url(/i/eg\_bg\_03.gif);

background-repeat: repeat-y;

}

* + 1. 背景定位background-position

 可以利用 background-position 属性改变图像在背景中的位置。如下在 body 元素中将一个背景图像居中放置：

body

{

background-image:url('/i/eg\_bg\_03.gif');

background-repeat:no-repeat;

background-position:center;

}

为 background-position 属性提供值有很多方法。不同类型的值对于背景图像的放置稍有差异。

1. 关键字：top、bottom、left、right 和 center。一般这些关键字会成对出现

位置关键字可以按任何顺序出现，若只出现一个关键字，则认为另一个关键字是 center。

|  |  |
| --- | --- |
| 单一关键字 | 等价的关键字 |
| center | center center |
| top | top center 或 center top |
| bottom | bottom center 或 center bottom |
| right | right center 或 center right |
| left | left center 或 center left |

1. 长度值， 100px 或 5cm，
2. 百分数值。
   * 1. 背景关联background-attachment

较长的文档向下滚动时，背景图像也会随之滚动。当文档滚动到超过图像的位置时，图像就会消失。

在此可通过 background-attachment 属性防止这种滚动。通过这个属性，可以声明图像相对于可视区是固定的（fixed），不会受到滚动的影响。其属性的默认值是 scroll。

body

{

background-image:url(/i/eg\_bg\_02.gif);

background-repeat:no-repeat;

background-attachment:fixed

}

* 1. 文本
     1. 文本对齐（text-align）

1. text-align:center 与 <CENTER>

text-align:center仅影响内容居中

<CENTER>：内容和元素都会居中

Justify：两端对齐

* + 1. 文本缩进（text-indent）

p {text-indent: 5em;}

缩进可以使用负值设置“悬挂缩进”，但为了避免文本首行超出边界，建议针对负缩进再设置一些边框、边距等

可以使用长度值、百分比等

缩进是可以继承的

* + 1. 行高（line-height）
    2. 字间距（letter-spacing）

word-spacing 字间距

默认值normal（等同于0）

接受一个正长度值（间隔增加）或一个负长度值（间隔减小）

p.spread {word-spacing: 30px;}

p.tight {word-spacing: -0.5em;}

<p class="spread">

This is a paragraph. The spaces between words will be increased.

</p>

<p class="tight">

This is a paragraph. The spaces between words will be decreased.

</p>

letter-spacing：字母间距

与word-spacing的在于，字母间距修改的是字符或字母之间的间隔

* + 1. 文本装饰（text-decoration）

属性值包括：none、underline（下划线）、overline（上划线）、line-throngh（贯穿线）、blink（文本闪烁）

* + 1. 垂直对齐（vartical-align）
    2. 文本变换（text-transform）

属性值包括：none、uppercase、lowercase、capitalize（单词首字母大写）

* + 1. 处理空白符white-space

white-space 属性会影响到用户代理对源文档中的空格、换行和 tab 字符的处理。

|  |  |  |  |
| --- | --- | --- | --- |
| 值 | 空白符 | 换行符 | 自动换行 |
| pre-line | 合并 | 忽略 | 允许 |
| Normal | 合并 | 忽略 | 允许 |
| Nowrap | 合并 | 忽略 | 不允许 |
| Pre | 保留 | 保留 | 不允许 |
| Pre-wrap | 保留 | 保留 | 允许 |

* 1. 字体

 font-family 属性 定义文本的字体系列，为避免系统未安装某一特定指定字体，建议设置是使用指定字体+通用字体结合设置。如下

h1 {font-family: Georgia, serif;}

font-style 属性最常用于规定斜体文本：normal、italic（斜体）、oblique（倾斜）

font-variant 属性可以设定小型大写字母

font-weight 属性设置文本的粗细。

font-size 属性设置文本的大小。（使用 em代替px 来设置字体大小，1em等于当前字体尺寸）

* 1. 链接

a:link - 普通的、未被访问的链接

a:visited - 用户已访问的链接

a:hover - 鼠标指针位于链接的上方

a:active - 链接被点击的时刻

* 1. 列表

 ul {list-style-type : square} //把无序列表中的列表项标志设置为方块。

ul li {list-style-image : url(xxx.gif)} //使用图像作为标志。

li {list-style : url(example.gif) square inside} //简写属性。用于把所有用于列表的属性设置于一个声明中。

* 1. 表格

//border属性设置表格边框

table, th, td {border: 1px solid blue;}

// border-collapse 属性设置是否将表格边框折叠为单一边框

table {border-collapse:collapse;}

table,th, td {border: 1px solid black;}

//通过 width 和 height 属性定义表格的宽度和高度。

table {width:100%;}

th {height:50px;}

//text-align 和 vertical-align 属性设置表格中文本的对齐方式。

td {text-align:right;}

// vertical-align 属性设置垂直对齐方式，比如顶部对齐、底部对齐或居中对齐

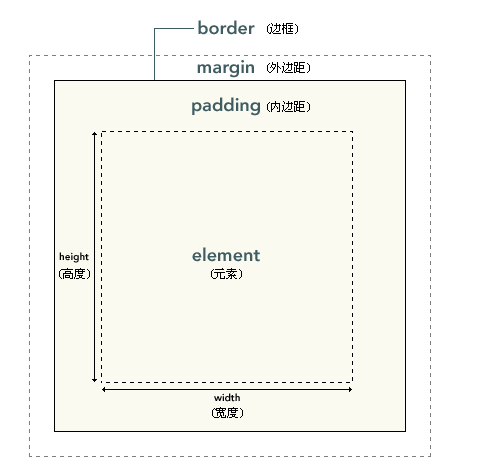
td {height:50px;vertical-align:bottom;}

//  td 和 th 元素设置 padding 属性可控制表格内边距

td {padding:15px;}

* 1. 轮廓（outline）

1. CSS框模型
   1. 框模型概述



元素框中包含实际内容

内边距包含的部分呈现了元素的背景，其边缘即为边框

边框以外的外边距默认是透明的，外边距可以是负值

//通用选择器设置

\* {

margin: 0;

padding: 0;

}

//

#box {

width: 70px;

margin: 10px;

padding: 5px;

}

* 1. 内边距

Padding可以设置各边等宽，也可以顺序设置

//设置各边内边距等宽

h1 {padding: 10px;}

//按顺序设置上下左右

h1 {padding: 10px 0.25em 2ex 20%;}

还可以通过以下四个属性单独设置上下左右的内边距

padding-top

padding-right

padding-bottom

padding-left

还可以通过百分比设置，其相对于父元素的值。

* 1. 边框

边框绘制在元素背景之上的，每个边框有三个方面可设置（样式、颜色、背景）

边框的样式设置方式类似于内边距，可以使用一个值设置四个边样式为同一类型，也可按顺序分别设置上下左右四个边框的样式，还可以通过单独的属性来设置上下左右的边框样式（border-top-style）

a:link img {border-style: outset;}

p.aside {border-style: solid dotted dashed double;}

//以下两个设置等价，但是单边框属性必须放在简写属性后，不然简写属性会覆盖单边框属性

p {border-style: solid solid solid none;}

p {border-style: solid; border-left-style: none;}

border-width，边框属性可以数值或者（thin、medium、thick）指定

border-color：transparent属性值可以定义一个透明边框

* 1. 外边距
  2. 外边距合并

1. CSS定位
   1. 定位概述
   2. 相对定位
   3. 绝对定位
   4. 浮动
2. CSS选择器
   1. 元素选择器
   2. 选择器分组
   3. 类选择器详解
   4. id选择器详解
   5. 属性选择器详解
   6. 后代选择器
   7. 子元素选择器
   8. 相邻兄弟选择器
   9. 伪类
   10. 伪元素

（未完成的后续补充）