# css的基本用法

1. 语法

css的基本用法是使用style属性，语法为：

<元素 style=”属性1:属性值; 属性2:属性值;”/>，如：

<span style="font-size: *50px*;font-weight:*spaborder*:;font-style: *italic*;color: *#009DFE*">字体一</span>

1. css的分类

css可分为内部css和外部css，内部css直接嵌在html代码内，外部css是单独的文件，只要在html中引入该.css文件即可，内部css又分为由单个标签使用的和作用于整个文件的样式

1. <link>标签

link标签的使用格式为：

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

rel属性只有stylesheet一个属性值被所有浏览器支持，它指定了当前文档与被链接文档之间的关系，href规定被链接文档的位置。

# css选择器

css提供了四种选择器，分别为： id选择器，类选择器，标签选择器，通配符选择器

## 1.id选择器

语法格式：

#id选择器名{

属性名:属性值;

}

eg：

#rightFrameStyle{

background-color: #F0F0F0;

}

注意：当两个选择器有相同属性时，可以统一声明，如：

.titleFontStyle{

background-color: gray;

font-style:italic;

font-weight: 900;

}

.fontStyle{

font-style:italic;

font-weight: 900;

}

可以改为：

.titleFontStyle{

background-color: gray;

}

**.fontStyle,.titleFontStyle**{

font-style:italic;

font-weight: 900;

}

此方法对类选择器同样适用

## 2.类选择器

语法格式：

.类选择器名{

属性名:属性值;

}

eg：

.classStyle{

font-size: 50px;

border: 5px;

background-color: red;

font-weight: 2px;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<html>

<head>

<title>css选择器</title>

<meta charset="utf-8">

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

</head>

<body>

<span><font class="classStyle">新闻一</font></span>

<span><font class="classStyle">新闻二</font></span>

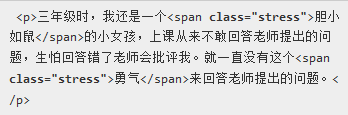
<span><font>新闻三</font></span>

</body>

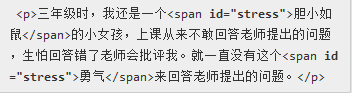
</html>

类选择器和id选择器有两大不同：第一，一个类选择器可以在同一个html中使用多次，而id选择器只能被选择一次，因为不同标签的id只能有一个；第二，一个class属性可以引用两个或多个类选择器，而一个id属性只能引用一个id选择器，如：

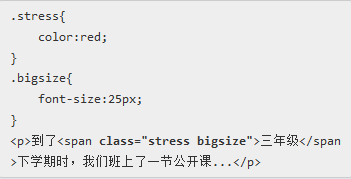
**下面代码是正确的：**

[](https://gss0.baidu.com/-fo3dSag_xI4khGko9WTAnF6hhy/zhidao/pic/item/8601a18b87d6277fe0d5954b2e381f30e824fcfa.jpg)

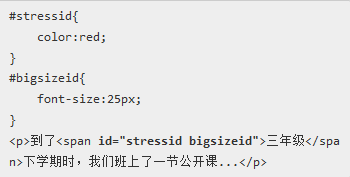
而下面代码是错误的：

[](https://gss0.baidu.com/-vo3dSag_xI4khGko9WTAnF6hhy/zhidao/pic/item/7c1ed21b0ef41bd590914f8757da81cb38db3d86.jpg)

**下面的代码是正确的**

[](https://gss0.baidu.com/9vo3dSag_xI4khGko9WTAnF6hhy/zhidao/pic/item/42166d224f4a20a4e7bc9ab496529822720ed027.jpg)

下面的代码是不正确的

[](https://gss0.baidu.com/-vo3dSag_xI4khGko9WTAnF6hhy/zhidao/pic/item/77c6a7efce1b9d1608093c12f5deb48f8d5464f3.jpg)

注意：当class=”stress bigsize”中两种类选择器发生冲突时，不以二者在html文件中的顺序为准，而是谁在.css文件中放在后面，谁生效，如：

html文件

<span><font class="classStyle">新闻一</font></span>

<span><font class="classStyle2 classStyle ">新闻二</font></span>

css文件

.classStyle{

color: red;

}

.classStyle2{

color: green;

}



## 3.标签选择器

语法格式：

html选择器名{

属性名:属性值;

}

eg：

span{

color: green;

}

注意：在优先级上，若发生冲突，选择器优先级为：**id选择器>类选择器>标签选择器**

## 4.通配符选择器

如果希望所有的元素使用统一的样式，可以使用通配符选择器。

语法格式：

\* {

属性名:属性值;

}

eg：

\*{

margin: 0;

padding: 0;

}

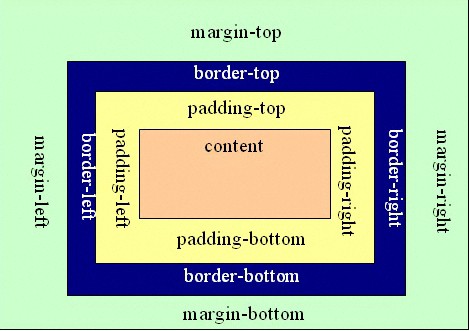
注意：如果margin**(padding)**给了四个值，则分别代表top，right，bottom，left，若给了三个值，则分别代表top，left&right(共用第二个属性)，bottom，如：

\*{

margin: 0px 10px 0px;

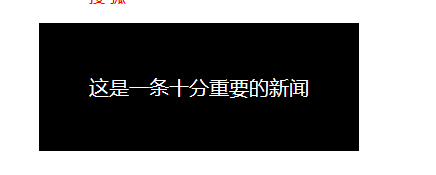
padding: 0;

}



## 5.父子选择器

当需要对某个引用某选择器的标签中的子标签做出样式修改，如：



将十分重要用红色标记，代码如下：

#idStyle{

color: white;

background-color: black;

font-size: 20px;

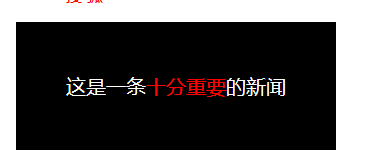
}

#idStyle span{

padding: 0px;

color: red;

}



注意：父子选择器适用于id选择器和类选择器

## 6.伪类

(1)html标签(一般为a标签):link，正常状态

选择所有未被访问的链接。  
(2) html标签(一般为a标签):visited，鼠标点击后再次停留在上面的状态

选择所有已被访问的链接。  
(3) html标签(一般为a标签):hover，鼠标经过的状态

选择鼠标指针位于其上的链接。  
(4) html标签(一般为a标签):active 按下鼠标时的状态

选择活动链接。

(5)html标签:after 这个伪元素允许制作人员在元素内容的最后面插入生成内容，需要和content属性一起使用，设置在对象后发生的内容。默认地，这个伪元素是inline行内元素，不过可以使用属性 display 改变这一点。

(6)last-child和

eg：

a:link{

color: black;

text-decoration: none;/\*文字定义为什么都没有，去掉下划线\*/

}

a:visited{

color: red;

}

a:hover{

text-decoration: underline;/\*文字定义为加上下划线\*/

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<a href="http://www.qq.com.cn/" target="\_blank">腾讯</a><br>

<a href="http://www.sohu.com/" target="\_blank">搜 狐</a>

注意，visited是你本次浏览器访问过的所有网站网站，应清除所有缓存后重新打开浏览器，才会显示应有样式

<style type="text/css">

h1:after {content:url(logo.gif)}

</style>   
<h1>标题内容</h1>

# 块元素和行内元素

## 1.行内元素(内联元素)

只能容纳文本或其他内联元素（有的浏览器什么都能容纳），如：<span>和<a>元素。

它只占内容的宽度，且默认不换行

**注意：行内元素不能设置宽高！**

## 2.块元素

一般从新行开始，可以容纳任何标签，即使内容不能占满一行，块元素也要把整行占满。如<div>和<p>

它不管内容的多少，都要换行，且占满整行

<span class="spanStyle">这是一个span</span>

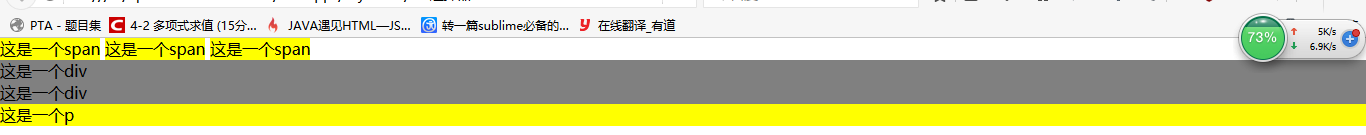
<span class="spanStyle">这是一个span</span>

<span class="spanStyle">这是一个span</span>

<div class="divStyle">这是一个div</div>

<div class="divStyle">这是一个div</div>

<p class="spanStyle">这是一个p</p>



当加上

\*{

margin: 0;

padding: 0;

}

通配符选择器之后，p和div之间不再有换行(空着的一行)存在

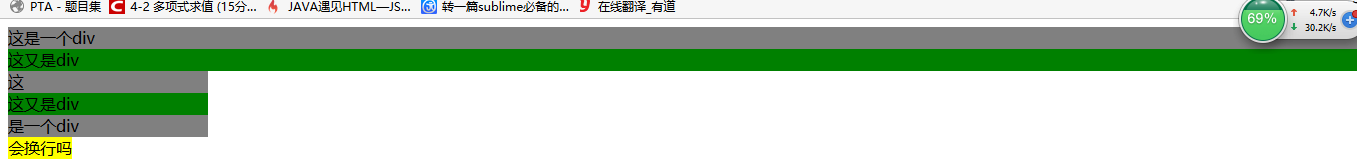
当然，我们也可以为div指定宽度来满足一些特别的要求。

注意：如果在div中嵌套div，会以第一个div的宽度为最长宽度，如：

**<div class="divStyle">**这是一个div<div class="divStyle1">这又是div</div>**</div>**

**<div class="divStyle" id="id">**这 <div class="divStyle1">这又是div</div>是一个div**</div>**

<span class="spanStyle">会换行吗</span>



有些版本的浏览器中，css的一些属性对span等行内元素不生效，应当尽量使用块元素。

## 3.块元素和行内元素的互相转化

在css属性中添加**display:block;**即可使行内元素转化为块元素来显示，如：

.s1{

color:red;

display:block;

}

这样就使得所有引用了s1的元素都按照块元素使用

在css属性中添加**display:inline;**即可使块元素转化为行内元素来显示，如：

.s2{

color:green;

display:inline;

}

这样就使得所有引用了s2的元素都按照行内元素使用，此时块元素既不换行，又不占满整行。

# css的标准流和非标准流

## 1.标准流

在网页布局中，写在前面的元素出现在前面，写在后面的元素显示在后面，这时默认的布局方式，也称为标准流

## 2.非标准流

在实际的网页开发中，我们经常需要用到非标准流来布局，即让某个元素脱离它原先的位置，出现在别的位置上。

# css的盒子模型



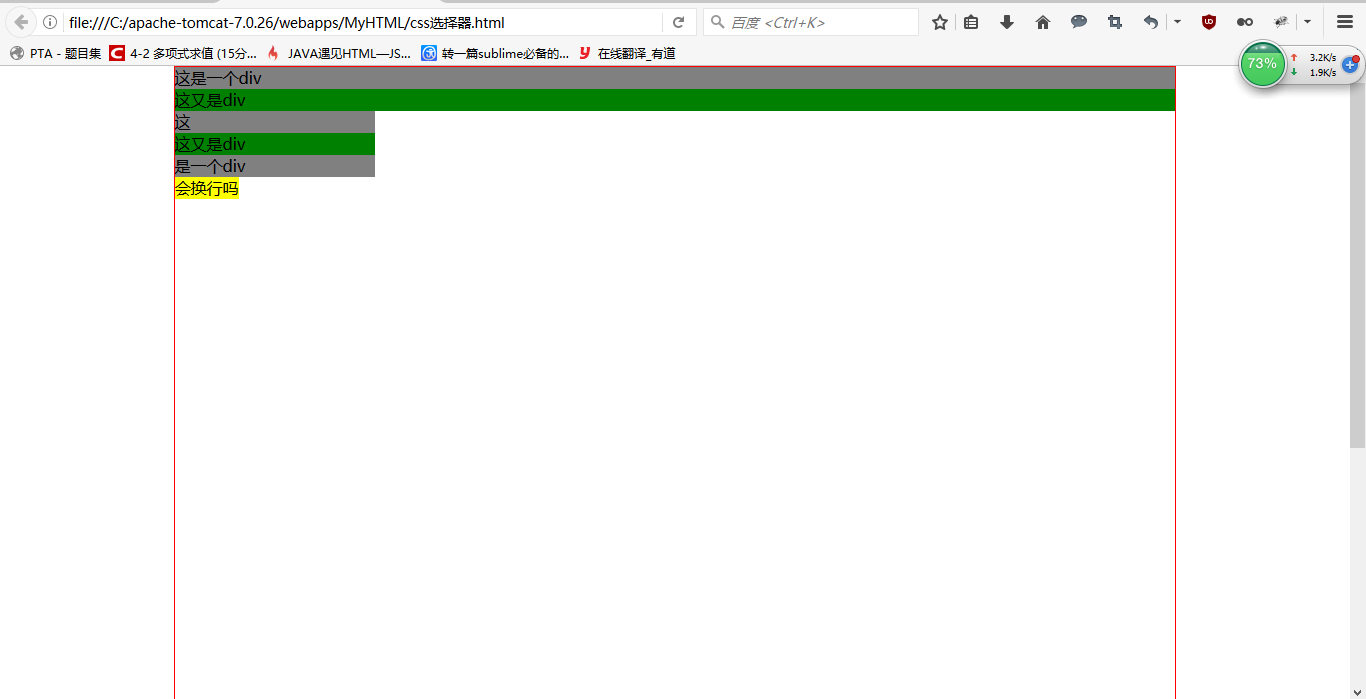
margin：两个元素间的距离，或者是元素距离浏览器边界的距离

border：本元素的边框

padding：本元素内容物(如：文字，图片，其他元素等各种标签)距离本元素边框的距离

content：内容物(本元素包裹着的元素)

**案例：我们经常需要控制body的大小，给body两边留白，并且浏览器缩小时留白也随之均匀缩小，直到留白为0，效果如下：**



**body{**

**border:1px red solid;**

**margin:0 auto;//0表示上下的margin为0，auto表示左右居中**

**width:1000px;**

**height:1000px;**

**}**

注意：如果margin**(padding)**给了四个值，则分别代表top，right，bottom，left，若给了三个值，则分别代表top，left&right(共用第二个属性)，bottom，如：

\*{

margin: 0px 10px 0px;

padding: 0;

}

margin:10px 5px;

上外边距和下外边距是 10px

右外边距和左外边距是 5px

优先满足上和左属性

# 6.定位和浮动

## 1.定位

定位的基本思想很简单，它允许你定义元素框相对于其正常位置应该出现的位置，或者相对于父元素、另一个元素甚至浏览器窗口本身的位置。

CSS 有三种基本的定位机制：普通流、浮动和绝对定位。除非专门指定，否则所有框都在普通流中定位。也就是说，普通流中的元素的位置由元素在 HTML 中的位置决定。

**定位：相对定位**

将position属性设为**relative**，即相对于一个元素的原本的正常位置来对其定位。但其在文本流中的位置还存在

将position属性设为**fixed**，即相对于浏览器窗口进行定位。通常，该元素会随着滚动条滚动而移动

将position属性设为**absolute**，它与relative的区别在于其脱离文本流，在文本流中不占位置，并且它是相对于html进行定位，**绝对定位元素的任何祖先元素没有进行任何的“relative”或者“absolute”设置，那么绝对定位的元素的参考物就是html**

## 2.浮动

当父窗口不够大时，会将右浮动部分挤到下一行，当同时左、右浮动都存在时，应先写右浮动，再写左浮动，否则还是会挤到下一行

# 7.css常用属性

## 1.图片控制

(1) background-position图像位置

该属性控制图像的位置，属性值为：

top left

top center

top right

如果您仅规定了一个关键词，那么第二个值将是"center"。

默认值：0% 0%。

center left

center center

center right

bottom left

bottom center

bottom right

x% y% 第一个值是水平位置，第二个值是垂直位置。

左上角是 0% 0%。右下角是 100% 100%。

如果您仅规定了一个值，另一个值将是 50%。

xpos ypos

第一个值是水平位置，第二个值是垂直位置。

左上角是 0 0。单位是像素 (0px 0px) 或任何其他的 CSS 单位。

如果您仅规定了一个值，另一个值将是50%。

您可以混合使用 % 和 position 值。

(2) background-repeat图片重复

该属性控制图像的重复，属性值为：

repeat 默认。背景图像将在垂直方向和水平方向重复。

repeat-x 背景图像将在水平方向重复。

repeat-y 背景图像将在垂直方向重复。

no-repeat 背景图像将仅显示一次。

inherit 规定应该从父元素继承 background-repeat 属性的设置。

(3) background-attachment滚动效果

background-attachment 属性设置背景图像是否固定或者随着页面的其余部分滚动。

scroll 默认值。背景图像会随着页面其余部分的滚动而移动。

fixed 当页面的其余部分滚动时，背景图像不会移动。

inherit 规定应该从父元素继承 background-attachment 属性的设置。

**注意：在firfox中需把 background-attachment 属性设置为 "fixed"，才能保证background-position在 Firefox 和 Opera 中正常工作。**

(4) box-shadow阴影控制

语法

box-shadow: h-shadow v-shadow blur spread color inset;

|  |  |
| --- | --- |
| h-shadow | 必需。水平阴影的位置。允许负值。 |
| v-shadow | 必需。垂直阴影的位置。允许负值。 |
| blur | 可选。模糊距离。 |
| spread | 可选。阴影的尺寸。 |
| color | 可选。阴影的颜色。请参阅 CSS 颜色值。 |
| inset | 可选。将外部阴影 (outset) 改为内部阴影。 |
| eg:  .Imagebackground{  height: 280px;  margin-top: 60px;  background: url(img/主页填充效果.png);  background-attachment: fixed;  background-position: 0% 70%;  background-repeat: repeat-x;  box-shadow: 0px 10px 5px #888888;  } |  |

(4) 渐变效果

1.线性渐变linear-gradient

linear-gradient(direction, color-stop1, color-stop2, ...);

direction 方向，to bottom(默认)、to top、to right、to left、 to bottom right等

(5) 滤镜效果

## 2.元素旋转

transform：none | 类型

类型：scale：缩放，1为原始大小。scale(x)。正数放大，负数缩小。属性值为一个时，x/y轴同时缩放；属性值为两个值时，分别控制x、y轴的缩放。

rotate：水平旋转，属性值格式为Xdeg。(deg是“度”的意思)rotate(Xdeg)。X为正数时，顺时针旋转；为负数时，逆时针旋转

translate：平移，基于XY轴重新定位元素。translate(Xpx,Ypx)。属性值为一个时，x、y轴参数相同；为两个时，x、y轴分别定位

skew：将元素沿水平方向倾斜变形。skew(Xdeg,Ydeg)。这个比较难表述，想象一下平行四边形吧。属性值为一个时，x、y轴参数相同；为两个时，x、y轴各自倾斜

matrix：矩阵

transition属性可以控制变换的花费的时间

## 3. nth-child选取标签

ie8以前的版本不支持该选择器

:nth-child(*n*) 选择器匹配属于其父元素的第 N 个子元素，不论元素的类型。

n： Odd 匹配下标是奇数的子元素的属性

even 匹配下标是偶数的子元素的属性

an + b 表示周期的长度，n 是计数器（从 0 开始），b 是偏移值

eg：

指定了下标(下标从1开始，n从1开始)是 3 的倍数的所有 p 元素的背景色：

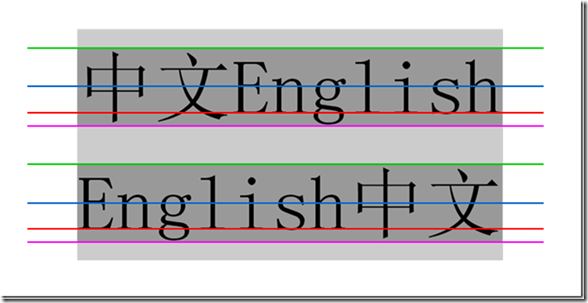
p:nth-child(3n+0)

{

background:#ff0000;

}

## 4. line-height行高



从上到下四条线分别是顶线、中线、基线、底线

**行高**是指被设置行高的元素内部上下文本行的基线间的垂直距离，即图中两条红线间垂直距离

**line Box**

如果一个标签没有定义height属性(包括百分比高度)，那么其最终表现的高度一定是由line-height起作用，比如有一个空的div，<div></div>，如果没有设置至少大于1像素高度height值时，该div的高度就是个0。如果该div里面打入了一个空格或是文字，则此div就会有一个高度。可能有人会跟认为这是文字撑开的，实际上不是，比如：

.test1{font-size:20px; line-height:0; border:1px solid red; background:gray;}

<div class="test1">测试</div>



那么，究竟是什么将div撑开了，答案是line-height。在盒子模型中，存在line boxes，它负责包裹每**一行**文字，而它也只有一个表现出来的特性，那就是高度，一个未设置高度的div，其高度是由一个一个包裹文字的line boxes撑起来的，line boxes将选取它所包裹的文字、图片、inline标签等元素中行高最高的一个作为自己的高度

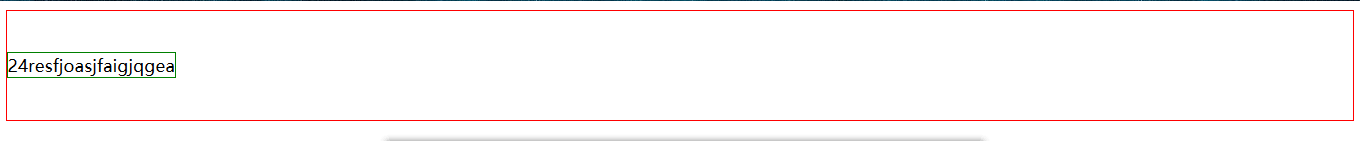
**因此，设置行高，实际上是设置line boxes的高度**

**行高还有个特点是line boxes中的元素相将会处于line-height的中间，因此**将行高设置和父元素一样大，可实现块级元素元素内部的行内元素的垂直居中，如果是块级元素想要用此方法实现垂直居中，则应该对其设置：display：inline-block; eg：

<div style="height: 100px;border: 1px red solid;">

<span style="border: 1px green solid;line-height: 100px;">24resfjoasjfaigjqgea</span>

</div>



## 5. vertical-align 属性

vertical-align属性设置一个元素的垂直对齐。

baseline 默认。元素放置在父元素的基线上。

sub 垂直对齐文本的下标。

super 垂直对齐文本的上标

top 把元素的顶端与行中最高元素的顶端对齐

text-top 把元素的顶端与父元素字体的顶端对齐

middle 把此元素放置在父元素的中部。

bottom 把元素的顶端与行中最低的元素的顶端对齐。

text-bottom 把元素的底端与父元素字体的底端对齐。

length

% 使用 "line-height" 属性的百分比值来排列此元素。允许使用负值。

inherit 规定应该从父元素继承 vertical-align 属性的值。

## 6. css优先级!important

!important属性用来声明css属性的优先级，添加了!important的属性优先级较高

eg：

.testClass{

color:blue !important;

}

<div class="testClass" style="color:red;">

测试Css中的Important

</div>

//显示为蓝色

## 6.其他

**1. overflow 属性**

overflow 属性规定当内容溢出元素框时发生的事情。

visible 默认值。内容不会被修剪，会呈现在元素框之外。

hidden 内容会被修剪，并且其余内容是不可见的。

scroll 内容会被修剪，但是浏览器会显示滚动条以便查看其余的内容。

auto 如果内容被修剪，则浏览器会显示滚动条以便查看其余的内容。

inherit 规定应该从父元素继承 overflow 属性的值。

**2.圆角属性**

border-radius用于向 div 元素添加圆角边框，eg：

div{

border:2px solid;

border-radius:25px;

}

**注意： 1.在设置该属性时应当先设置border，<input>标签也不例外**

**2.在table表单中border-collapse:collapse和border-radius不相容**

**3.visibility隐藏元素**

通过visibility:hidden隐藏的元素即使被隐藏了，仍会留下空白，但是通过display隐藏的元素不会留下空白

# 8.css常见问题

1.块级元素内部添加内联元素留有空隙问题<http://www.zhangxinxu.com/wordpress/?p=4925>

2.使用float当窗浏览器口缩小，浮动的元素会折行，eg：

<div style="width:300px; height:30px; background:#eee; float:left;">Left</div>  
<div style="width:300px; height:30px; background:#ccc; float:right; clear:right;">Right</div>

解决方案：

添加margin属性，eg：

<div style="width:300px; height:30px; background:blue; float:left; margin-right:300px;">Left</div>  
<div style="width:300px; height:30px; background:#ccc; float:right; margin-left:-300px;">Right</div>

或者在外面添加一个固定大小的div

3.为了使背景色在浏览器页面缩放时仍能覆盖全部背景，应使用background-repeat:repeat属性,eg:

.**topBottomBackground** {  
 **background-color**: **#F1EFF1**;  
 **background-repeat**:**repeat** ;  
}

有时，缩小浏览器窗口，底部出现进度条，当滑动至右边，某个元素的右边就出现空白，这时由于页面的该元素宽度小于另一个元素的width，且另一个元素的width大于窗口的width，导致把页面撑开，出现了滚动条，右侧的空白长度=另一个元素的宽度-该元素的宽度，解决方法是给该元素设置**min-width:另一个元素的宽度**，高度也是一样

# 9.css命名规范

对于CSS，为了避免样式冲突，我们总会给其赋予相当特殊的命名，我们都希望写出精简高效的CSS代码，如果CSS重用性越高，想必就越高效，就像人名一样，如果名字越普通，越没有含义，越容易被重用，所以CSS要想重用性高，就需要**命名简单**。但是，简单的命名越容易造成**样式冲突**，例如，当你写网页主栏的头部时，可能写了一个类选择器.head{}，但当你写侧边栏时，你却发现.head{}名称被占用了，为了处理样式冲突和命名简单之间的矛盾，我们需要一套命名规范来帮助我们命名

## 9.1面向属性的命名方法

在给选择器命名时，我们很容易掺杂语义，如：. side-item-header代表侧边栏的头部，但如果主栏的头部也用到了. side-item-header的所有属性，该怎么办？我们能做的估计就是新命名一个class，就像是.body-item-header，明明是同样的CSS属性，结果却不能重用

那么如何解决这个问题呢？答案就是“**面向属性的命名**”。将所有的页面设计，语义通通抛弃，只针对选择器自身属性的一种命名方式，eg:

.**pt9**{  
 **padding-top**: 9**px**;  
}

.**w265**{  
 **width**: 265**px**;  
}

毫无疑问，这种css样式彻底分离的重用率将会达到一种惊人的地步，但它也有副作用，对于一些复杂的效果，一味地分离css样式会使得HTML代码变得很痛苦，HTML代码量可能会非常庞大。所以，我们需要在拆分和融合间找到一个平衡点，并且，面向属性的命名方法还不具有语义化的特点，维护不方便，因此，我们一般将一些常用的属性值分离，eg：

.**block**{  
 **display**: **block**;  
}

.**hidden**{  
 **visibility**: **hidden**;  
}

## 9.2 BEM规范

BEM的意思就是块（block）、元素（element）、修饰符（modifier），块应当是一个独立的实体，而元素应当是块的一部分，修饰符作为一个块或是一个元素的一种属性，代表这个块或这个元素在外观或是行为上的改变。

‘\_\_’符号用于连接block和element

‘--’符号用于连接block和modifier或者element和modifier，eg：

.block{}

.block\_\_element{}

.block--modifier{}

**.person**{}  
.**person\_\_hand**{}  
.**person--female**{}  
.**person--female\_\_hand**{}  
.**person\_\_hand--left**{}

除了BEM外，还有OOCSS，SMACSS，Atomic Design等命名规范

## 9.3使用和命名原则

1.避免使用ID选择器

 2.无标签

 3.避免层层嵌套，即父子选择器

eg：

#test{}

li{}

.test span{}

CSS的渲染方式是“**从右往左**”渲染的, 就拿#test ul{}举例，先查找页面上所有的ul标签，再去寻找其中父元素id为test的ul元素，因此，分层级会使渲染效率降低