

# CSS入门和高级技巧

第8天课堂笔记（本课程共8天）

班级：北京前端训练营1期

讲师：邵山欢

日期：2016年4月2日

邵山欢老师

微博：@邵山欢

QQ : 179427026

E-mail : shaoshanhuan@163.com

爱前端官网 ：[www.iqianduan.cn](http://www.iqianduan.cn)

# 目录

[CSS入门和高级技巧 1](#_Toc9073)

[目录 2](#_Toc20806)

[一、图片透明 3](#_Toc29942)

[二、样式表的组织 6](#_Toc14927)

[2.1 内嵌样式表 6](#_Toc28985)

[2.2 外联样式表 6](#_Toc12038)

[2.3 行内样式 8](#_Toc10159)

[2.4 导入式的样式表 8](#_Toc23156)

[2.5站点的css组织 9](#_Toc4886)

[三、说道说道display:inline-block; 11](#_Toc16591)

[四、iconfont 12](#_Toc4469)

# 一、图片透明

先来复习一下盒子的透明问题：

|  |
| --- |
| 1. opacity:0.6; |

介于0~1之间，为了让IE兼容，我们要使用IE自己的滤镜：

|  |
| --- |
| 1. filter:alpha(opacity=60); |

盒子的透明，会让里面的所有内容都透明了，如果想要让文字清晰，就不能把文字装到盒子里，要用绝对定位定。

图片格式：

● jpg/jpeg：著名图片压缩格式，有压缩算法，能够自己更改质量；即使质量是100，也是压缩的，也是颜色失真的。没有图层，不能动。banner、新闻图、焦点图都要用jpg格式的图片。



●png：不压缩的，无法调整质量。有图层，放到服务器上的时候，要导出一下，平面化整个图像，不要图层，压平。不能动。擅长做logo、几何图形比较规律的图片，尺寸比jpg还要小：



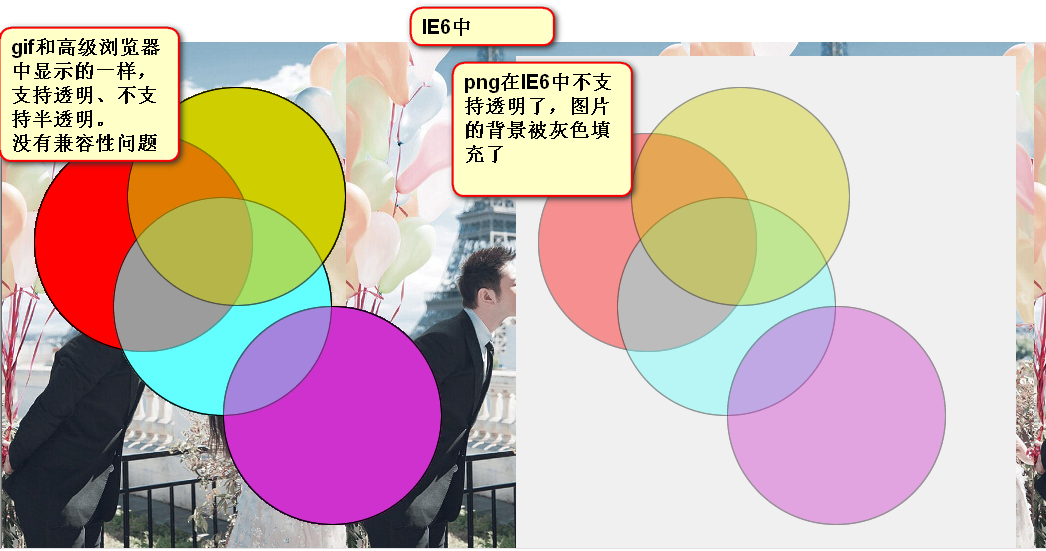
支持透明和半透明。

●gif：严重失真，颜色数量只有256、128…… 2种颜色。能够制作动画。没有图层。

支持透明，但是不支持半透明。



IE6的显示有问题的：



所以现在我们就能琢磨出来一种兼容性的解决方法：

在IE6中就是用gif来替代png

|  |
| --- |
| 1. background: url(images/icon\_v9.png) no-repeat 0 0; 2. \_background-image: url(images/icon\_v9.gif); |

还有一种办法，就是用一个神奇的js，来解决png的IE6透明问题。

老师给你一个png.js这个文件，这个文件业界比较著名，机理就是反编译这张图片，把所有的透明像素点都用IE特有的滤镜来反编译一下：

需要先“引包”：

|  |
| --- |
| 1. **<script type="text/javascript" src="png.js"></script>** 2. <script type="text/javascript"> 3. **DD\_belatedPNG.fix(".haha,.xixi");** 4. </script> |

红色的语句，就是引包，src里面放置的就是js文件的相对路径。

蓝色的语句，是调用，圆括号里面要放置所有需要解决png问题的图片的选择器。需要注意的是，选择器可以是img直接插入的，也可以是背景图片。

可以用HTML hack，加一个壳，让其他浏览器不加载这个js代码：

|  |
| --- |
| 1. <!--[if IE 6]> 2. <script type="text/javascript" src="js/png.js"></script> 3. <script type="text/javascript"> 4. DD\_belatedPNG.fix(".haha,.xixi"); 5. </script> 6. <![endif]--> |

所有有背景定位的图片，这个js包就有bug，就定位不准了。所以精灵图就不要用这个js来解决了。

|  |
| --- |
| 1. display:none; |

加上这个属性的元素，就如同HTML标签没有写一样。将会放弃自己的位置。后面的元素就会上来。

还有一种可以让元素隐藏的东西，叫做

|  |
| --- |
| 1. visibility:hidden; |

不会放弃现在的位置，就是自己隐藏掉了。就如同

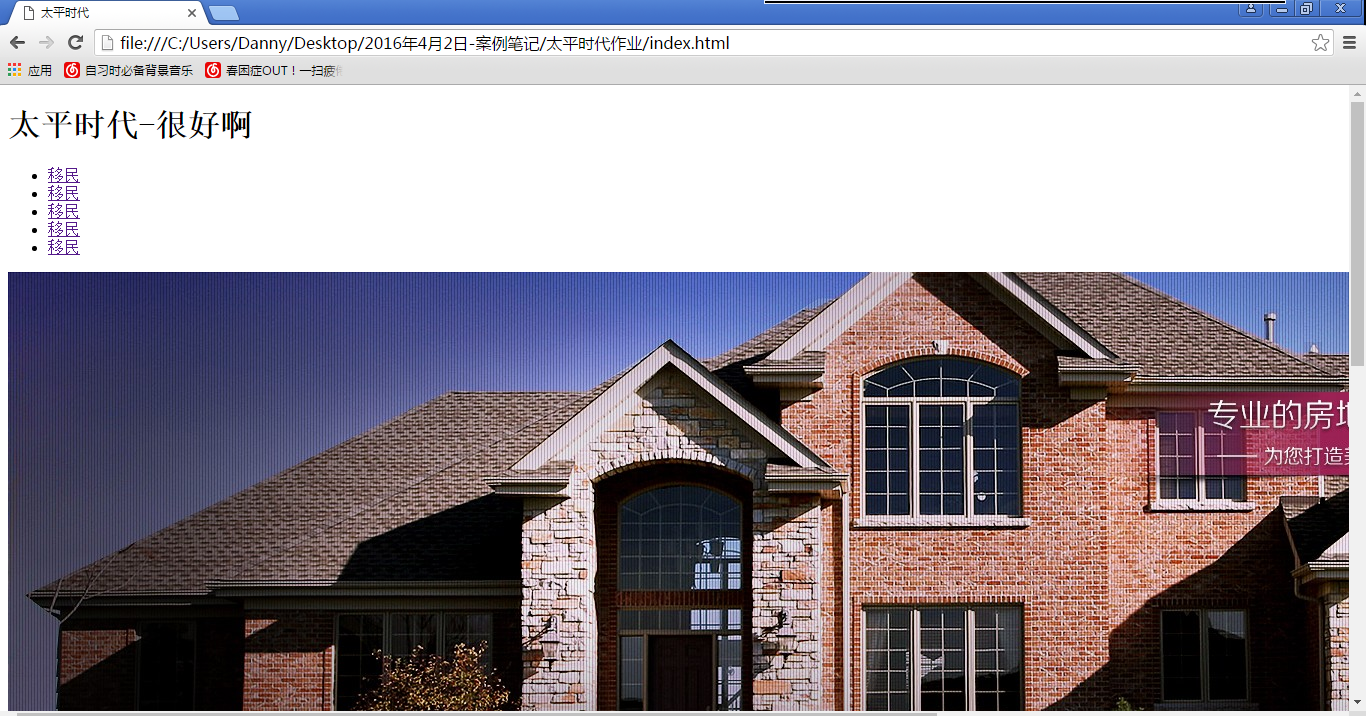
|  |
| --- |
| 1. opacity:0; |

# 二、样式表的组织

## 2.1 内嵌样式表

原来把样式写在<style type=”text/css”> </style>叫做内嵌样式表。内嵌样式表，感觉HTML和CSS没有分开，但**是有好处，可以保证HTML加载的时候，就已经有样式了。页面不会出现CSS加载不正常的情况**。

**如果样式表外联，可能出现极小概率的样式表没有请求成功的情况，刷新一下就好了**：



不过内嵌样式表的问题更大，HTML、CSS混在一个文件了，不好维护；并且css也不能共用。

## 2.2 外联样式表

可以把css单独写成一个.css文件，html页面用link标签来链接这个样式表：

|  |
| --- |
| 1. <link **href**=”css.css” **type**=”text/css” **rel**=”stylesheet” /> |

link就是连接的意思，表示现在这个html页面要链接一个样式表；

href就是超文本地址的意思，和a里面href一样一样的，相对路径、绝对路径都行；

type就是类型，text/css就是纯文本的css类型

rel就是relationship，关系，就是我引用的这个文件和我这个html的关系，是stylesheet，样式表。

“**三小件儿**”：href、type、rel

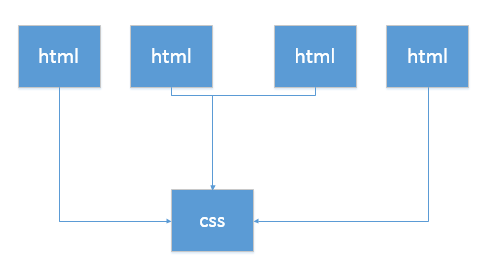
link是一个自封闭标签，至此我们已经见过的自封闭：

|  |
| --- |
| 1. <img src=”1.jpg” /> 2. <input type=”button” /> 3. <meta name=”” content=”” /> 4. <link href=”” type=”” rel=”” /> 5. <br /> |

这样的好处非常明显，就是:

1) HTML和CSS文件彻底分离了！html文件里面只有html结构了，css样式放到了单独的css文件中了；

2) 多个html文件，共享一个css文件了。页面中相同的部分，就可以引用同样的样式表，不用重写了！



要注意两个严重的问题：

1） 外联的css中，如果要用到背景图片，那么路径是从css文件出发，而不是从html文件出发。

background:url(**../**images/1.jpg);

2） 外联样式表不影响权重，就如同我们的css写在html页面内的link那里一样，同样是比选择器权重、就近原则。

## 2.3 行内样式

如果想给一个元素加一个样式，可以直接在这个元素的html标签内加上style属性就行了

|  |
| --- |
| 1. <div **style="color:red**;**border:1px solid red**;**width:800px**;**height:800px**;**font-family:'微软雅黑','宋体'**;**background:url(images/shishi.jpg) repeat-x 10px 10px;"**>文字</div> |

行内样式已经丧失了我们的css的一些优点，就没有选择器的概念了，因为就是给这一个元素加样式（该继承的还是继承）。所以**工作很少用，后台工程师爱用这个东西**。

Chrome浏览器中，用element.style来表示行内样式：



行内样式的权重，是最大的，可以认为是写在第4位的权重，比id厉害：

比如#box div.haha ul li 它的权重(1,1,3)

而<div style=”width:100px;color:red;”></div> 的权重(1,0,0,0)

任何的权重都比不上行内。没有!important大。

|  |
| --- |
| 1. <div id="father"> 2. <div style="color:red" id="box"> 3. 文字 4. </div> 5. </div> |

|  |
| --- |
| 1. #father #box{color:green;} |

文字是红色的，因为行内的权重是最大的。

## 2.4 导入式的样式表

几乎已经绝迹了，没有人在用了。

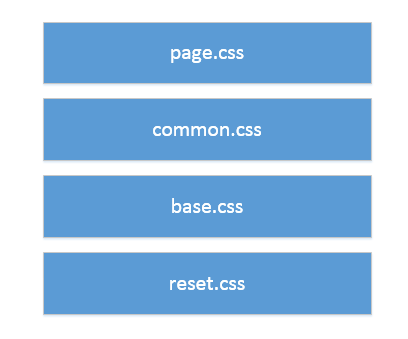
|  |
| --- |
| 1. <style type="text/css"> 2. **@import url(css/css.css);** 3. body{ 4. background-color: pink; 5. } 6. </style> |

用@import url(地址) ， 这是一个固定的语法，没有什么好说的。就可以把样式表导入进来。

比如写在第一行。这种写法，就是一个理论上的意义，没有实战意义，因为感觉是内嵌样式表、外联样式表混合起来了。如果你要外联，你就老实的用<link href=”” />。如果内嵌，就老实内嵌。不要杂糅。

## 2.5站点的css组织

css文件是分层次：



**reset.css：**让所有的元素都有默认的样式，比如ul没有小圆点，比如h1的字号22px……

这是因为不同浏览器对标签有不同的默认样式，比如某个浏览器h1默认字号是22px，而有的是24px。

最最著名的reset.css就是雅虎公司的YUI的reset.css。

网址：<http://yuilibrary.com/yui/docs/cssreset/>

以后页面上不允许出现这个：

|  |
| --- |
| 1. \*{margin:0;padding:0;} |

有超过1秒的渲染时间。所以就要用h1,p,h2罗列出来，清除margin、padding。我们的YUI reset.css就是这么做的。

表格的边框合并：

|  |
| --- |
| 1. border-collapse: collapse; |

|  |  |
| --- | --- |
|  | bordre-collapse:collapse; |

这个样式表不能更改，就是负责统一全浏览器默认样式的，去掉了所有的margin、padding ，去掉了小圆点。

**base.css** 公共类，就是原子类

|  |
| --- |
| 1. .f12{font-size:12px;} 2. .f13{font-size:13px;} 3. .f14{font-size:14px;} 4. .f16{font-size:16px;} 5. .f20{font-size:20px;} 6. .fb{font-weight:bold;} 7. .fn{font-weight:normal;} 8. .t2{text-indent:2em;} 9. .lh150{line-height:150%} 10. .lh180{line-height:180%} 11. …… 12. …… |

建议同学们自己总结这些原子类。

**common.css： 页面和页面之间，公共的样式部分。**

**page.css: 每个页面自己的样式**

# 三、说道说道display:inline-block;

标准流的事儿，标准流把元素分为行内、块级。性质截然不同。

|  |
| --- |
| 1. display:inline-block; |

让元素同时具有行内、块级元素的性质。

设置为行内快的元素，在高级浏览器中的表现和float：left;非常的像，有一个非常大的不同：

就是空白折叠现象。代码竖着写，中间会折叠成一个空格：

|  |
| --- |
| 1. <div class="box1"></div> 2. <div class="box2"></div> 3. <div class="box3"></div> |



span也能设置宽度、高度了；

margin也好用。IE6中不让默认的块级元素转为行内块。

display:inline-block;那么好用，为什么不用，而是用浮动呢？

1） 空白折叠现象，元素贴边爱有一个空格。

2） 只有左，没有右。

3） 盒子加上inline-block之后，就和文本一样一样了，div转为inline-block，就和能设置宽高的span一样了。

4） 时代原因，大家都爱写浮动，inline-block在抢占技术实施标准那几年，没有火起来。

但是inline-block能够解决兼容性问题。

|  |  |
| --- | --- |
|  |  |

浮动的父元素不写宽度，会自动收缩为儿子的宽度；儿子不巧，转块了也没写宽度，所以要撑满父亲。

高级浏览器、IE7、IE8等：会尽全力满足父亲的收缩，缩到不能缩位置；

IE6：会尽全力满足让儿子撑满。

解决办法：

|  |
| --- |
| 1. \_display:inline-block; |

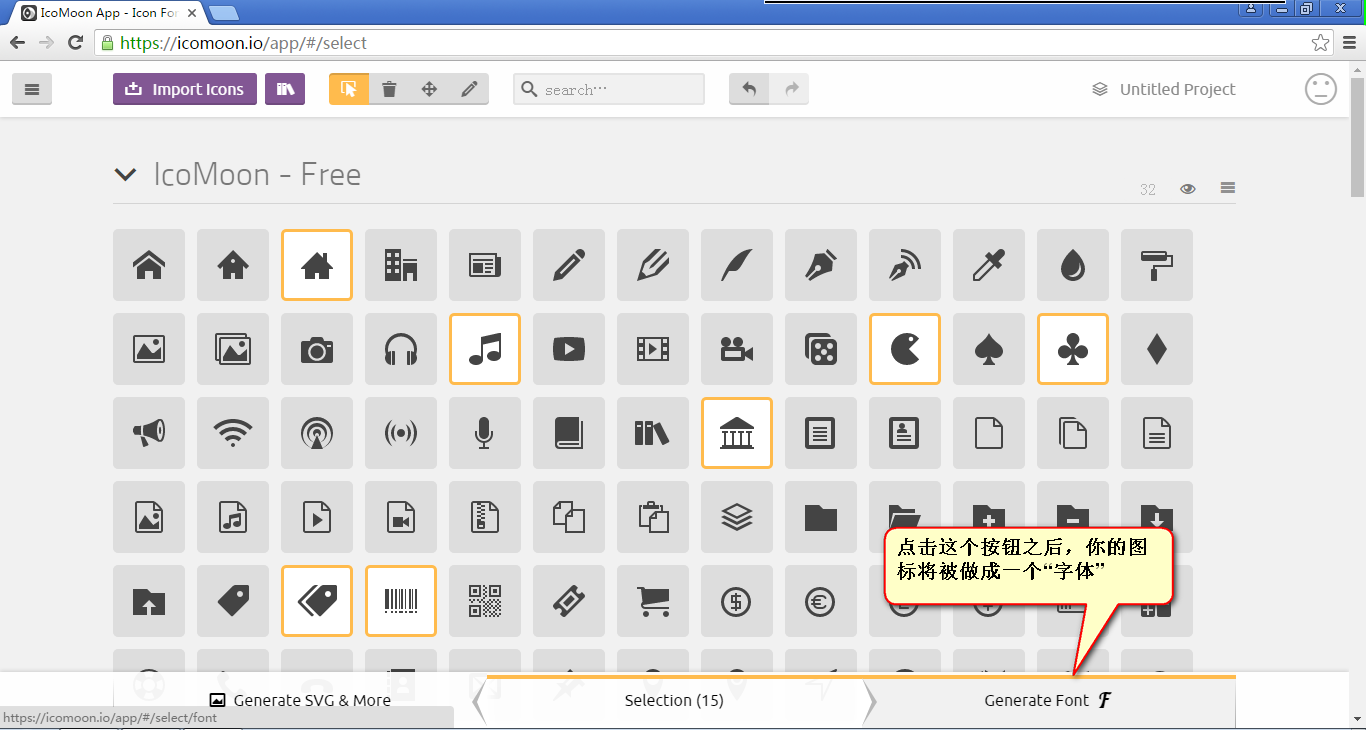
# 四、iconfont

字体图标：图标不是图标了，也不是精灵了，而是字了。

字怎么是图标，这个字在某一个字体下，就是图标。

**icomoon.io 是国外非常著名的一个字体图标引擎，可以免费的、在线把任何矢量图片变成字体。**

https://icomoon.io/app/



字体图标的特点：

1） 不用下载图片了，就不浪费http请求了，但是要下载字体文件，到底省流量没有，见仁见智。

2） 只能是单色，不能是多个颜色。无法制作彩色花图标。

3） 控制这个图标，就用控制字体的方式来控制就行了。

icomoon里面图标不多：

美工来制作ai格式的图标，可以在icomoon里上传，然后改。

字体文件，每个系统有自己的拓展名

