# CSS3

第2天课堂笔记（本课程共3天）

目录

[CSS3 1](#_Toc478723537)

[一、兼容性策略 2](#_Toc478723538)

[二、CSS3概述 3](#_Toc478723539)

[三、CSS3选择器 5](#_Toc478723540)

[3.1 属性选择器 5](#_Toc478723541)

[3.2 儿子序选择器 9](#_Toc478723542)

[四、2D变形 12](#_Toc478723543)

[4.1 旋转变形 13](#_Toc478723544)

[4.2 缩放变形 13](#_Toc478723545)

[4.3 斜切变形 14](#_Toc478723546)

[五、3D变形 15](#_Toc478723547)

[5.1 舞台和演员 15](#_Toc478723548)

[5.2 rotateX 15](#_Toc478723549)

[5.3 rorateY 16](#_Toc478723550)

[5.4 小案例 16](#_Toc478723551)

[5.5 旋转轴 17](#_Toc478723552)

[六、空间移动 18](#_Toc478723553)

# 一、兼容性策略

**平稳退化：**高级浏览器很炫酷，低级浏览器看不见炫酷的效果，页面结构稳定、骨架稳定、该显示的东西都有。CSS的兼容策略，基本都是平稳退化。

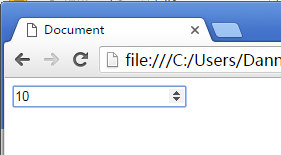
比如你现在用一个border-radius:15px; 那么IE6、7、8是没有圆角的，ok，你心想没有圆角就没有了吧，因为盒子稳定，只不过就是美观性差了点，这就是平稳退化的思路。平稳退化就是放低低端用户，低端用户在审美上上就有一些瑕疵。

**渐进渐强：**给低版本用户普通的功能，现代浏览器的用户能有更方便的功能。和平稳退化一样，都放弃了低端用户，但是也保障了低端用户的实用性。

比如，购买一个东西输入购买的数量

|  |
| --- |
| 1. <input type="number" id="kk"/> |

低版本浏览器会把这个东西当做text框，没有任何附加的功能。而高版本浏览器的number控件多了上下键的功能：



再比如百度的新闻页面：

IE8看到的是换图策略的按钮，这个效果在高级浏览器上应用也会很好看。



但是我们不满足，我们希望高级浏览器会更好看，渐进渐强。



平稳退化和渐进渐强哲学上都是一致的：放弃低端用户，只给低端用户足够的使用性，但是美观、易用程度上有瑕疵。平稳退化先以高版本思考问题，低版本用户就不兼容就算了，因为已经足够能使用这个页面了；渐进渐强是对自己的严格要求，给高级浏览器用户更多的功能。

还有一个更经常使用的策略，就是破罐破摔，就是不让低版本浏览器用户看到这个页面，强行跳转：

|  |
| --- |
| 1. <!--[if lte IE9]> 2. <script type="text/javascript"> 3. window.location = "error.html"; 4. </script> 5. <![endif]--> |

IE6、7、8都不认识新标签，比如header、section。

有神人帮你写了一个js，叫做html5shiv，能够把所有新标签换成div来解析。

# 二、CSS3概述

CSS3是层叠式样式表的第3个版本，也是最新版，是各个浏览器厂商推动的标准，实际上W3C没有严格的CSS3标准。



CSS3没有对CSS进行大的颠覆，以前有人猜想，CSS3会有大的改变，比如CSS3有计算能力，但实际上CSS3并没有增加那么强的计算能力。CSS4中，估计要引用变量、函数的概念了。越来越像LESS、SASS。

但是，CSS3很炫，因为增加了3D、过渡、变形、动画等等新特性。

CSS3和HTML5没有任何关系，也就是说HTML5能用CSS3，HTML4.01也能用。

CSS为什么能够蓬勃发展，就是因为CSS底子好，CSS有一个非常好的特点：遇见不认识的选择器、属性，会静默，不会报错。挑自己认识的来解析。

|  |
| --- |
| <style type="text/css">  ewjlkj894{  adsf:@34adf;  adsfads:23423;  adsfad5:3245;  }  **div{**  wejrlkj:4654;  **width:400px;**  asdtlkj45:rtret;  **height:400px;**  sadgfdsa:w3432;  sadgfdsa:w3432;  sadgfdsa:w3432;  **background-color: orange;**  adsf:234324;  **}**  </style> |

CSS3新增了：

选择器40多种

伪元素

伪类

颜色表示法

边框和阴影

background系列属性改变

过渡

2D变形

3D变形

动画

响应式的新属性，比如多列，文字溢出折叠…

# 三、CSS3选择器

## 3.1 属性选择器

我们再CSS2.1中遇见的选择器：

|  |
| --- |
| p  #box  .red  #box ul li  div.red  div,ul  \* |

CSS3中新增了40种选择器。我们按类别讲，现讲属性选择器。

比如CSS：

|  |
| --- |
| img[src="images/2.png"]{  border: 10px solid #000;  } |

选择src属性为images/2.png的元素

|  |
| --- |
| <img src="images/0.png" alt="">  <img src="images/1.png" alt="">  **<img src="images/2.png" alt="">**  <img src="images/3.png" alt="">  <img src="images/4.png" alt=""> |

IE7开始兼容！

特别的，没有必要必须写标签名：

合法：

|  |
| --- |
| [src="images/2.png"]{  border: 10px solid #000;  } |

引号不能省略：

|  |  |
| --- | --- |
| [src=images/2.png]{  border: 10px solid #000;  } |  |

可以用中文：

|  |
| --- |
| [alt="哈哈"]{  transform:rotate(30deg);  } |

有拓展：

= 完全匹配

^= 开头匹配

$= 结尾匹配

~= 空格匹配

|= 短横匹配

\*= 任意匹配

|  |
| --- |
| div[data-info^="哈"]{  background-color: red;  } |

匹配开头：

|  |
| --- |
| <div data-info="哈"></div>  <div data-info="2哈"> </div>  <div data-info="嘻嘻 哈"></div>  <div data-info="哈 嘻嘻"> </div> |

|  |
| --- |
| div[data-x$="哈"]{  background-color: red;  } |

匹配结尾：

|  |
| --- |
| <div data-x="哈"></div>  <div data-x="2哈"></div>  <div data-x="嘻嘻哈"></div>  <div data-x="嘻嘻 哈 么么哒"></div> |

|  |
| --- |
| div[data-y **~=** "哈哈"]{  background: green;  } |

匹配单蹦儿的，用空格和别人分开的，不和别人粘连的：

|  |
| --- |
| **<div data-y="哈哈"></div>**  <div data-y="2哈哈"></div>  **<div data-y="嘻嘻 哈哈"></div>**  **<div data-y="嘻嘻 哈哈 么么哒"></div>**  <div data-y="嘻嘻 哈哈么么哒"></div>  <div data-y="哈哈-么么哒"></div> |

所以：

|  |
| --- |
| div.box |

等价于

|  |
| --- |
| div[class ~= "box"] |

|  |
| --- |
| div[data-z |= "哈哈"]{  background: green;  } |

必须以哈哈-开头的，当然只有哈哈，也被选择：

|  |
| --- |
| <div data-z="哈哈"></div>  <div data-z="2-哈哈"></div>  <div data-z="嘻嘻-哈哈"></div>  <div data-z="嘻嘻 哈哈 么么哒"></div>  <div data-z="嘻嘻-哈哈么么哒"></div>  <div data-z="哈哈-么么哒"></div>  <div data-z="哈哈-2"></div>  <div data-z="哈哈 么么哒"></div>  <div data-z="-哈哈-么么哒"></div>  <div data-z="嘻嘻 哈哈-么么哒"></div> |

|  |
| --- |
| div[data-z |= "哈哈"] |

99.9999%等价于：

|  |
| --- |
| div[data-z ^= "哈哈-"] |

上面的比下面的多选了一个哈哈本身。

|  |
| --- |
| div[data-z \*= "哈哈"] |

只要有任何地方有哈哈，就匹配：

|  |
| --- |
| <div data-m="哈哈"></div>  <div data-m="哈哈 嘻嘻"></div>  <div data-m="嘻嘻哈哈么么哒"></div>  <div data-m="哈么哈么哒"></div>  <div data-m="么哈么哈哒"></div>  <div data-m="我有哈哈"></div> |

统统兼容到IE7.也就是说你们公司如果不用IE6了，就直接嗷嗷用这些选择器！！！

特别的，如果没有运算符，就表示有这个属性就行。

|  |
| --- |
| a[title]{  color:red;  } |

|  |
| --- |
| <a href="">文字</a>  <a href="" title="哈哈">文字</a>  <a href="">文字</a>  <a href="">文字</a> |

## 3.2 儿子序选择器

选择的是#s1里面的第一个div儿子：

|  |
| --- |
| #s1 div:first-child{  background-color: red;  } |

选择最后一个儿子：

|  |
| --- |
| #s1 div:last-child{  background-color: green;  } |

选择第二个儿子：

|  |
| --- |
| #s2 div:nth-child(2){  background-color: red;  } |

4th anniversary 四周年纪念日

注意，下标从1开始，而不是0。

从第1个开始，每2个选一个：

|  |
| --- |
| #s3 **div:nth-child(2n + 1)**{  background-color: red;  } |

从第2个开始，每3个选一个：

|  |
| --- |
| #s3 div:nth-child(3n + 2){  background-color: red;  } |

公式从第b个开始，每a个选一个：

|  |
| --- |
| #s3 div:nth-child(an + b){  background-color: red;  } |

所有单数：

|  |
| --- |
| div:nth-child(2n+1) |

所有偶数：

|  |
| --- |
| div:nth-child(2n+2) |

用好这些选择器挺难的，有大坑：

比如 header div:nth-child(2) 不吹哨，选择的是每个header里面的第2个div。暗含each。

|  |
| --- |
| <header>  <div></div>  <div></div>  <div></div>  <div></div>  </header>  <header>  <div></div>  **<div></div>**  <div></div>  <div></div>  </header>  <header>  <div></div>  **<div></div>**  <div></div>  <div></div>  </header> |

第二个坑，就是不是标签的顺序，而是儿子的顺序！不是在同种类里数序号，而是在总儿子中数序。

比如，想让下面标红的标签变色：

|  |
| --- |
| <nav>  <p>1</p>  <p>2</p>  <p>3</p>  <p>4</p>  <h3>5</h3>  <h3>6</h3>  <h3>7</h3>  <h3>8</h3>  </nav> |

应该：

|  |
| --- |
| h3**:nth-child(6)**{  } |

小题目，想让下面★的元素变红：

|  |
| --- |
| <main>  <div class="teshu"></div>  <div class="teshu"></div>  <div></div>  <div></div>  <div class="teshu">★</div>  </main> |

答案：

|  |
| --- |
| div:nth-child(**5**){  } |

还有一个系列，叫做nth-of-type()。当然还有first-of-type、last-of-type。

比如

|  |
| --- |
| div .haha:nth-of-type(5){  color:red;  } |

选择的是div里面第5个有haha类的元素。

|  |
| --- |
| <div>  <p class="haha">p</p>  <p>p</p>  <p class="haha">p</p>  <p class="haha">p</p>  <h3>h</h3>  <h3 class="haha">h</h3>  <h3 class="haha">h★</h3>  <h3>h</h3>  <h3>h</h3>  </div> |

注意一个大坑，

h3:nth-of-type(2) 选择的是：只要某个h3恰好为父亲的第2个h3，就被选择。

|  |
| --- |
| <div>  <p>p</p>  <p>p</p>  <p>p</p>  <h3>h</h3>  <p>p</p>  <p>p</p>  **<h3>h</h3>**  <h3>h</h3>  </div>  <div>  <p>p</p>  <p>p</p>  <p>p</p>  <h3>h</h3>  <p>p</p>  <p>p</p>  **<h3>h</h3>**  <h3>h</h3>  </div> |

总结一下，都是选择儿子：

|  |
| --- |
| :first-child IE7  :last-child IE9  :nth-child() IE9  :nth-last-child() IE9  :first-of-type IE9  :last-of-type IE9  :nth-of-type IE9  :nth-last-of-type IE9 |

特别的jQuery支持所有CSS3选择器，并且能做到IE6兼容。

|  |
| --- |
| $("h3:eq(2)").animate({"font-size":100},1000); |

不等价于：

|  |
| --- |
| $("h3:nth-of-type(2)").animate({"font-size":100},1000); |

eq是大序列！！！ nth-of-type是某个人的第2个同类儿子。

# 四、2D变形

CSS3中最大的更新：

1） transition 过渡

2） transform 变形

3） animation 动画

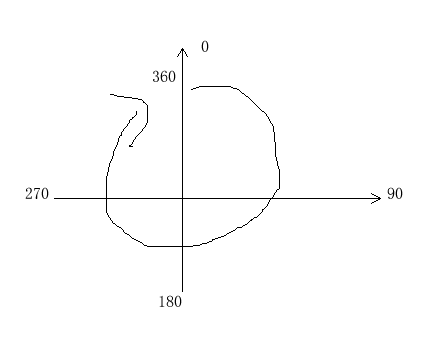
transform属性就是变形，transformer就是变形金刚。IE9开始兼容。

## 4.1 旋转变形

|  |
| --- |
| 1. transform:**rotate(45deg)**; |

rotate就是旋转的意思，deg为单位，角度。

正方向：



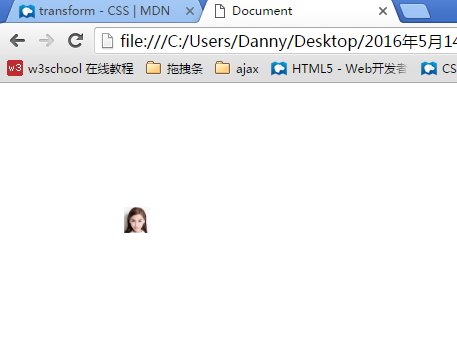
值可以是正数，也可以是负数： transform:rotate(-90deg) 等价于transform:rotate(270deg)

任何的变形，都能够被过渡。

## 4.2 缩放变形

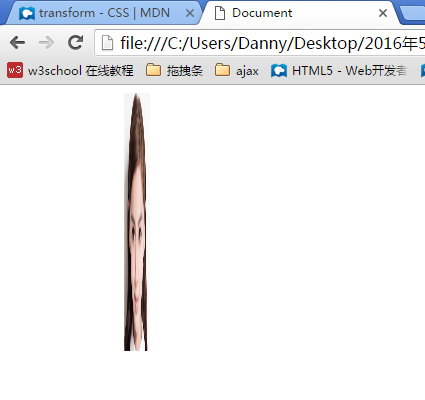
|  |
| --- |
| 1. transform:**scale(0.1)**; |

scale就是缩放的意思，值是正数，放大就是大于1，缩小就是0~1。值没有单位。



可以写两个参数，用逗号隔开，表示横向的缩放倍数，和纵向的缩放倍数。

|  |
| --- |
| 1. transform:scale(0.1,1); |



可以拆开写：

|  |
| --- |
| 1. transform:**scaleX(0.1) scaleY(1)**; |

## 4.3 斜切变形

skew就是斜切的意思，可以有两个值，分别表示横向斜切度数，纵向斜切度数。

|  |
| --- |
| 1. transform:**skew(30deg,10deg)**; |

等价于：

|  |
| --- |
| 1. transform:**skewX(30deg) skewY(10deg)**; |

总结一下2D变形就这三个：

|  |
| --- |
| 1. transform:rotate(30deg); 2. transform:scale(4,1); 3. transform:skew(30deg,40deg); |

如果想要有多个变形，用空格隔开

|  |
| --- |
| 1. transform:skew(50deg,20deg) scale(2,0.2) rotate(40deg); |

# 五、3D变形

## 5.1 舞台和演员

IE10开始兼容

3D变形就要有舞台的概念，舞台负责加perspective属性，表示景深，单位是px，表示你的眼睛离这个舞台的距离，也可以理解为里面变形的图片“陡峭”的程度。演员就是里面的img，img负责transform变形

|  |
| --- |
| 1. <style type="text/css"> 2. div{ **→ 舞台** 3. width: 258px; 4. height: 258px; 5. border: 1px solid #000; 6. /\*你的眼睛到div的距离\*/ 7. **perspective:500px; → 舞台负责加景深** 8. margin: 100px; 9. } 10. img{ **→ 演员** 11. border: 1px solid #000; 12. **transform:rotateX(45deg); → 演员负责变形** 13. } 14. </style> |

|  |
| --- |
| 1. <div><img /></div> |

## 5.2 rotateX



属性也是transform，也就是说是2D变形还是3D变形主要看后面的值。

|  |
| --- |
| 1. transform:**rotateX(45deg)**; |

正方向是后仰。 负数就是磕头。



## 5.3 rorateY



|  |
| --- |
| 1. transform:**rotateY(40deg)**; |



正方向就是从顶部看，逆时针旋转。右肩往前转动。

同样的，如果想有多个旋转，就要用空格隔开：

|  |
| --- |
| 1. transform:**rotateY(90deg) rotateX(90deg)**; |

rotateZ就是rotate就是平面中旋转。

## 5.4 小案例

背面不可见

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

像两张扑克牌背靠背贴在一起，baby从-180 → 0 ， xiaoming从0 → 180

|  |
| --- |
| 1. <div> 2. <img class="baby" src="images/baby.png" alt="" /> 3. <img class="xiaoming" src="images/xiaoming.png" alt="" /> 4. </div> |

## 5.5 旋转轴

旋转轴：

|  |
| --- |
| 1. transform-origin:0% 0px; 2. transform:rotateY(45deg); |

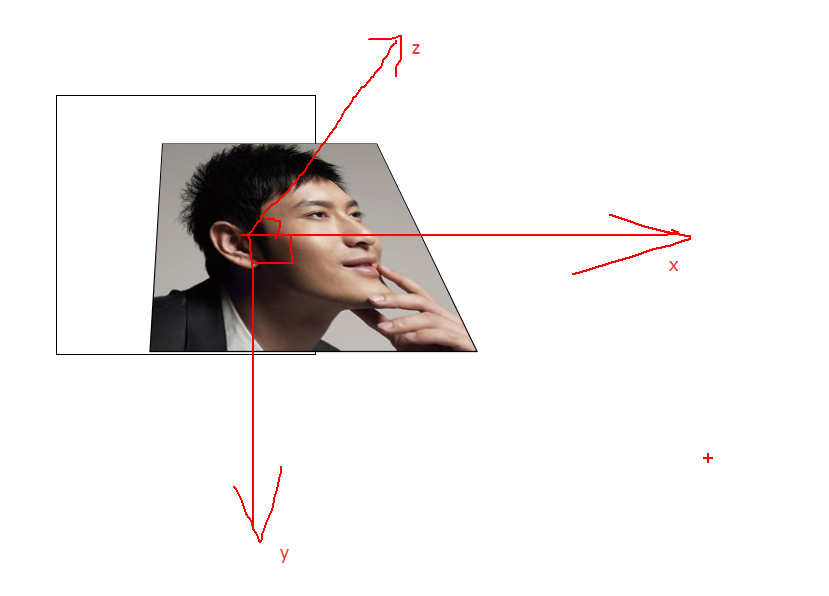


rotateY的时候，是绕着Y轴旋转，所以我们只关心轴的横向位置。即第一个参数起作用。

rotateX的时候，是绕着X轴旋转，所以我们只关心轴的纵向位置。即第二个参数起作用。

百分比表示舞台的百分之多少。

# 六、空间移动



每个空间里的元素，都有x、y、z三个轴。

x轴的正方向是这个元素的左手。

y轴的正方向是这个元素的下边。

z轴的正方向是这个元素的朝向。

我们可以利用

|  |
| --- |
| 1. transform:translateX(300px); 2. transform:translateY(300px); 3. transform:translateZ(300px); |

来进行空间移动。



CSS3中最最经常干的事儿，就是先让一个元素有rotateX或者rotateY或者两个都有。然后进行空间移动。

|  |
| --- |
| 1. transform:rotateX(30deg) rotateY(30deg) translateZ(300px); |

当一个组合体（内部有3D），自己要进行旋转，那么一定要加上属性：

|  |
| --- |
| 1. transform-style:preserve-3d; |

加上之后内部的小元素的3D变化将保留。