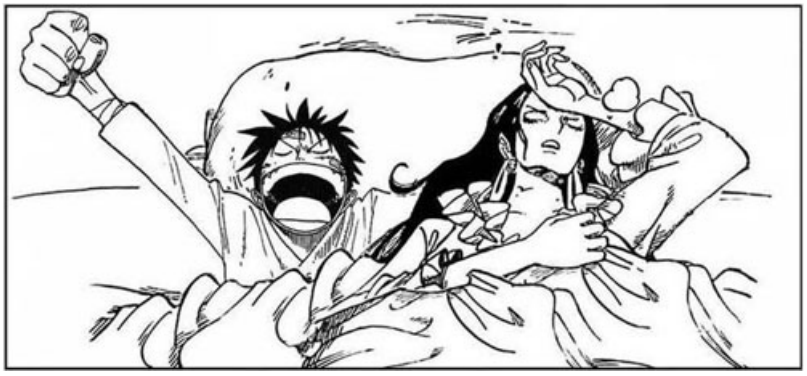
好吧，CSS3 3D transform变换，不过如此！

这篇文章发布于 2012年09月7日，星期五，01:05，归类于 CSS相关。 阅读 680735 次, 今日 59 次 366 条评论

一、写在前面的秋裤

早在去年的去年，我就大肆介绍了2D transform相关内容。看过海贼王的都知道，带D的家伙都不是好惹的，2D我辈尚可以应付，3D的话，呵呵，估计我等早就在千里之外被其霸气震晕了~~

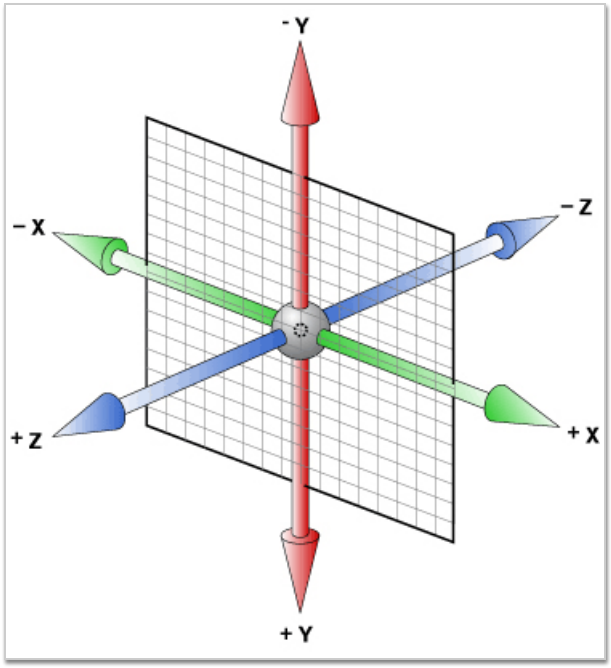
看看下图女帝的动作以及神情，就可以知道名字带D的家伙的厉害！



最近折腾iPad的一些东西，有一些3D效果的交互。有些事情，总以为是遥远的未来，谁知真正发生的时候说来就来，比如说一颗想结婚的心，又比方说在实际项目中折腾3D transform效果。

然而，虽然以前折腾过3D变换效果(webkit)，但都是依葫芦画瓢，囫囵吞枣，真正要轻松实现想要的3D效果，是需要深入理解的，于是，此时的自己苦逼了，泪奔ing……

木有办法，找资料，自己思考学习呗，当我看到下面这张基本图的时候，我的右侧的浓眉毛不由自主抖动了两下，呵，呵呵~~



这个长得像原子核一样的是什么东东？那像章鱼哥一样四处横生的箭头好吓人哦！后面怎么还有一个苍

蝇拍？？ CSS好可怕，我要回去找妈妈……

想必大部分的同行应该跟我一样，没有爱因斯坦爷爷的智商，没有上镜需要把表摘掉的爸爸。因此，那些术语连篇的CSS3 3D transform介绍的资料过于耀眼，无法直视。怎么办？

好吧，佛家有云，我不入地狱谁入地狱。这里，我就从凡人们的视角说说CSS3 3D transform的一些东西，希望说的东西比较亲民，不要吓着大家。

二、首先，情感化认识

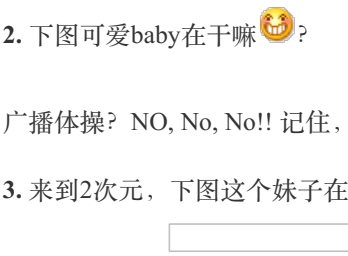
我觉得吧，要想理解一个东西，最好先有一些感性的认识。

CSS3中的3D变换效果，本质上就是我们OOXX时候各种姿势的变换，又称各种体位的变换。

虽然都是成年人，但考虑到仍有不少窝中待守的雏鸟，如果上面的解释想不过来，就想想以下这些：
1. 下图的这些人在干嘛？

跳水？NO, No, No!! 记住，他们不是在跳水，是在做3D变换！！！

2. 下图可爱baby在干嘛？



广播体操？NO, No, No!! 记住，他不是在做操，是在做3D变换！！！

3. 来到2次元，下图这个妹子在这幅姿态称为：

卖萌？NO, No, No!! 记住，他不是在卖萌，是在做3D变换！！！

哈哈哈哈，是否意识到：在显示世界中，一切的动作（包括上面巨乳萌妹所引发的精虫上脑），都是属于3D transform变换。 因此，要学习与理解3D transform变换很简单，一句话，到现实世界找个东西映射一下即可。

三、认识的突破口：rotateX, rotateY, rotateZ

3D transform中有下面这三个方法：

- `rotateX( angle )`
- `rotateY( angle )`
- `rotateZ( angle )`

理解了这三个方法，后面更难懂的`perspective`就好下手了，可以说是突破口！

`rotate` 旋转的意思， `rotateX` 旋转X轴， `rotateY` 旋转Y轴， `rotateZ` 旋转Z轴……

什么X轴/Y轴/Z轴，这几个词从我嘴里一出来，别说你们，我自己都晕了~~

赶快，从现实世界找对应东西理解（参照下面人的旋转）：
邹凯的体操单杠运动是 `rotateX` ；

蔡依林姐姐的钢管舞是 `rotateY` ；

旋转飞刀的特技表演是 `rotateZ` 。

还是理解不过来？好吧，假设你是男的，以你的女朋友举例，假如原本你和她面对面站着，然后你——
从正面将其推到就是 `rotateX` ；

让其原地转个90度欣赏其侧面的丰满曲线就是 `rotateY` ；

把妹子抱到床上侧面躺着就是 `rotateZ` 。

于是，下面CSS世界中的简单3D效果是不是更容易理解了呢？！

四、必不可少的perspective属性

`perspective` 的中文意思是：透视，视角！

`perspective` 属性的存在与否决定了你所看到的是2次元的还是3次元的，也就是是2D transform还是3D transform. 这不难理解，没有透视，不成3D.

我们初中学美术，或者学建筑的同学肯定接触过透视的一些东西：

不过，CSS3 3D transform中的透视的透视点与上面两张示例图是不同的：CSS3 3D transform的**透视点是在浏览器的前方**！

或者这么理解吧：显示器中3D效果元素的透视点在显示器的上方（不是后面），近似就是我们眼睛所在方位！

比方说，一个1680像素宽的显示器中有张美女图片，应用了3D transform，同时，该元素或该元素父辈元素设置的 `perspective` 大小为2000像素。则这张美女多呈现的3D效果就跟你本人在1.2个显示器宽度的地方(1680*1.2≈2000)看到的真实效果一致！！

五、translateZ帮你寻找透视位置

如果说 `rotateX` / `rotateY` / `rotateZ` 可以帮助理解三维坐标，则 `translateZ` 则可以帮你理解透视位置。

我们都知道近大远小的道理，对于没有 `rotateX` 以及 `rotateY` 的元素， `translateZ` 的功能就是让元素在自己的眼前或近或远。比方说，我们设置元素 `perspective` 为201像素，如下：

```
perspective: 201px;
```

则其子元素，设置的 `translateZ` 值越小，则子元素大小越小（因为元素远去，我们眼睛看到的就会变小）； `translateZ` 值越大，该元素也会越来越大，当 `translateZ` 值非常接近201像素，但是不超过201像素的时候（如200像素），该元素的大小就会撑满整个屏幕（如果父辈元素没有类似overflow:hidden的限制的话）。因为这个时候，子元素正好移到了你的眼睛前面，所谓“一叶蔽目，不见泰山”，就是这么回事。当 `translateZ` 值再变大，超过201像素的时候，该元素看不见了——这很好理解：我们是看不见眼睛后面的东西的！

再生动的文字描述也不如一个实例来得直观，您可以狠狠地点击这里：translateZ方法辅助理解perspective视角demo

建议Chrome浏览器下访问，可以使用 `range` 控件，演示效果更赞，如下截图：-100时候最小，200时候超级满屏（垂直方向因特殊布局限制没有显示），250的时候因为元素已经在视点之外，因此是一片空白（看不见）。

六、perspective属性的两种书写

`perspective` 属性有两种书写形式，一种用在舞台元素上（动画元素们的共同父辈元素）；第二种就是用在当前动画元素上，与transform的其他属性写在一起。如下代码示例：

```
.stage {
    perspective: 600px;
}
```

以及：

```
#stage .box {
    transform: perspective(600px) rotateY(45deg);
}
```

您可以狠狠地点击这里：perspective属性的两种书写demo

结果如下缩略图：

从上图我们貌似可以看到，虽然书写的形式，属性名称不一致，但是，效果貌似是一样的~~果真是这样吗？？？

实际上不然，上面的demo上下两个效果之所以会一样，是因为舞台上只有一个元素，因此，发生了巧合，其正好表现一样了。如果，如果舞台上有很多个元素，则两种书写形式的表现差异就会立马显示出来了！

您可以狠狠地点击这里：舞台多元素下的perspective两种书写对比demo

demo页面效果缩略图如下（因背景色随机，可能与下图有差异）：

好吧，图中的效果其实不难理解。上面舞台整个作为透视元素，因此，显然，我们看到的每个子元素的形体都是不一样的；而下面，每个元素都有一个自己的视点，因此，显然，因为rotateY的角度是一样的，因此，看上去的效果也就一模一样了！

关于Chrome浏览器以及透视盲区

在Chrome浏览器下，要想看到完整的3D效果，还需要3D变换元素正好在窗体的垂直居中位置，因此，在Chrome浏览器下，生成了两个位置居中的按钮，帮助您看到想要的效果：

当我们改变第一个 `range` 控件值为200的时候，您会发现右侧第三个元素看不见了：

这不难理解，前面一排门，每个门都是1米，你距离门2米，显示，当所有门都开了45°角的时候，此时，距离中间门右侧的第二个门正好与你的视线平行，这个门的门面显然就什么也看不到。这就是为什么上面右侧第三个门一片空白的元素——特定的视角以及距离形成的视觉盲区。

七、理解perspective-origin

`perspective-origin` 这个属性超级好理解，表示你那双色迷迷的眼睛看的位置。默认就是所看舞台或元素的中心。有时候，我们对中心的位置是不感兴趣的，希望视线放在其他一些地方。比方说：

一图胜千言，屌丝男们这个应该都懂的。

下面为立方体的实际应用透视效果图：

```
perspective-origin: 25% 75%;
```

八、transform-style: preserve-3d

`transform-style` 属性也是3D效果中经常使用的，其两个参数，`flat|preserve-3d`. 前者 `flat` 为默认值，表示平面的；后者 `preserve-3d` 表示3D透视。

`preserve-3d` 符合我们真实世界的思维认识。比方说，你让妹子右转了45度，此时妹子脑袋左转45度想你吐舌卖萌，妹子的脸蛋应该和你是面对面平行的。

应用 `transform-style: preserve-3d` 声明的元素确实是这样表现的，但是，如果使用默认的 `flat` 值，其效果表现——恕我想象力有限——想不通：妹子的脸还是左转45度的，同时脑袋似乎移到了身体以外的地方！

因此，基本上，我们想要根据现实经验实现一些3D效果的时候，`transform-style: preserve-3d` 是少不了的。一般而言，该声明应用在3D变换的兄弟元素们的父元素上，也就是舞台元素。

九、backface-visibility

在显示世界中，我们无法穿过软妹A看到其身后的软妹B或C或D；但是，在CSS3的3D世界中，默认情况

下，我们是可以看到背后的元素（也不知可不可以透视妹子的衣服~）！

因此，为了切合实际，我们常常会这样设置，使后面元素不可见：

```
backface-visibility:hidden;
```

十、实际应用-图片的旋转木马效果

您可以狠狠地点击这里：图片的旋转木马效果demo

建议在足够新版本的FireFox浏览器或Safari浏览器下观看，Chrome可能需要居中定位查看，下图为效果缩略图：

原理：
那些看上去很酷酷的CSS3 3D效果其实就颠来倒去那几个属性（本文提到的这几个），折腾来折腾去，这里这个效果显然也是如此。

首先HTML结构，如下：

```
舞台
    容器
        图片
        图片
        图片
        ...
```

对于舞台，很简单，加个视距，比方说800像素：

```
 perspective: 800px;
```

对于容器，很简单，加个3D视图声明，如下：

```
transform-style: preserve-3d;
```

然后就是图片们了。为了不至于产生类似DNA的螺旋状效果，我们让所有图片 `position:absolute`，公用同一个中心点。

显然，图片旋转木马是类似钢管舞旋转的运动，因此，我们关心的是 `rotateY` 的大小。

因为要正好绕成一个圈，因此，图片 `rotateY` 值正好0~360等分，于是，如果有9张图片，则每个图片的旋转角度累加40(360 / 9 = 40)度即可。因此有：

```
img:nth-child(1) { transform: rotateY(   0deg ); }
img:nth-child(2) { transform: rotateY(  40deg ); }
img:nth-child(3) { transform: rotateY(  80deg ); }
img:nth-child(4) { transform: rotateY( 120deg ); }
img:nth-child(5) { transform: rotateY( 160deg ); }
img:nth-child(6) { transform: rotateY( 200deg ); }
img:nth-child(7) { transform: rotateY( 240deg ); }
img:nth-child(8) { transform: rotateY( 280deg ); }
img:nth-child(9) { transform: rotateY( 320deg ); }
```

这样就好了吗?

No, No, No!!!

想想看那，虽然9个绝色美女每个人的方位不一样，但都站在同一个点上，早就挤作一团，A罩都挤成C了，显然是不行的（见下图只设置rotateY）！我们需要拉开空间~~

如何拉开空间，很简单。

想想看那：9个美女，分别面朝东南西北共9个不同方位，她们只要每个人向前走个4~5步，美女们之间的空间不久拉开了，呈现圆形了！想象一下夜空中，礼花绽开的场景~~

这里的向前走4~5步，聪明的人应该已经知道了，就是本文提到的`translateZ`，当`translateZ`为正值的时候，元素会向其面对的方向走去；如果元素无旋转，就会朝显示器走来！！

现在只剩下一个问题了，美女们要向前走多远呢？？

这个距离是有计算公式滴！

拿本demo距离，每张美女图片的宽度是128像素，因此，有如下理想方位效果图：

上图中红色标注的`r`就是的demo页面中图片要`translateZ`的理想值（该值可以让所有图片无缝围成一个圆）！

`r`的计算很简单，有初中数学水平的人应该都会：

```
r = 64 / Math.tan(20 / 180 * Math.PI) ≈ 175.8
```

demo页面为了好看，图片之间留了点间距，使用的`translateZ`的值为`175.8 + 20 = 195.8`.

最后的最后，要让木马旋转起来，只要让容器每次旋转40度就可以了。

节省篇幅，具体的JavaScript操作代码就不展示了，您有兴趣可以查看demo页面源代码。

理解了旋转木马3D效果实现原理，基本上，其他些3D效果可以轻松驾驭了，因此，本效果还是值得你花功夫看看滴~~

补充于2018-07-10
新增3D动画案例，实现2个3D开门效果，有完成源代码，有兴趣可以点击这里：“CSS3实现3D开门动画效果”。

十一、好吧，结语

理论上，现实世界，及3次元世界中的各种有规律的运动效果都可以使用CSS3 transform 3D方法实现。文章最后的旋转木马效果可以说是各类千奇百怪效果中的沧海一粟~~其他各类有的没有的效果就靠你的大脑就构想了。至于实现嘛，理解了，也就都是小菜。但是，要是不理解，纯粹从网上copy些效果代码，那永远就是copy的命咯！

文章篇幅已经很长了，我的指头也敲出老茧来了，就不再啰嗦什么了。希望本文的嗑叨、卖弄、折腾能够让您学习CSS3 3D transform变换的相关东西更加轻松点！

行文仓促，文中有错误在所难免，欢迎诸位指正。

最后，感谢100offer对本文的大力承包和赞助！

> 《CSS世界》签名版独家发售，包邮，可指定寄语，点击显示购买码

（本篇完） // 想要打赏？点击这里。有话要说？点击这里。

« 近期手机网页项目一些杂碎心得分享

翻译：清除各个浏览器中的数据研究 »

标签： 3d, 3D效果, animate, backface-visibility, css3, perspective, transform, transform-style, transition

发表评论（目前366条评论）

名称 (必须)

邮件地址(不会被公开) (必须)

网站

提交评论

« 先前评论

1. 白说道：
2018年12月29日 11:01

css3 transform 还是你的博客文章入门 深入了解的。看了很多书 一些文章都是概念。
谢谢!

回复

2. **aria说道：**
2018年12月26日 20:45

前端是个难得妹子占比还比较多的IT行当，考虑一下我们看到的感受- –

回复

3. **perryhuang说道：**
2018年12月23日 11:30

讲一个知识有一个关注点够，干嘛要引入那么多关注点了，很反感拿一些不正当图片举例子，适当就好，过犹不及啊

回复

4. 网警说道：
2018年11月17日 16:36

过头了，下不为例！

回复

5. **cc说道：**
2018年11月12日 09:31

通俗易懂

回复

6. **langJS说道：**
2018年09月27日 15:38

看得鸡儿梆硬!老师也是同道中人啊

回复

7. **teeny说道：**
2018年09月25日 09:48

上班时间看这篇文章领导以为我在看什么小黄文

回复

8. **ZeroJsus说道：**
2018年09月20日 15:58

想不到鑫哥 也有这么牛逼的时候

回复

9. **AlenStone说道：**
2018年08月10日 15:58

大佬受我一拜，通俗易懂

回复

10. **houfengqaz说道：**
2018年08月9日 12:53

我用ie查看那个开门动画demo 能够正常运行
为什么旋转的那个就没有反应
ie到底兼不兼容perspective了

回复

11. **ahah说道：**
2018年08月2日 15:22

上班时看这个简直了，别人从我身边一过我就要赶紧关掉。明明是在做项目学习，偏偏搞得跟在看黄色网站一样啊哈哈哈哈。不过学到了很多知识点，解决了困扰我的问题。

回复

> **顶说道：**
> 2018年08月6日 11:22
>
> 必须顶
>
> 回复

> **帅哥胡说道：**
> 2018年08月10日 15:08
>
> 小伙子,看张大大的文章我跟你同感! 不敢叫同桌看都,开的都是小框口…
>
> 回复

12. **tsdd说道：**
2018年06月15日 16:10

最后一次吐槽了，真的难以理解作者的文风，买了一本css世界，看的那是相当难受

回复

13. 亮说道：
2018年06月15日 15:12

看懂部分 够用了 谢谢 再用 再来研究

回复

14. **wangjianye说道：**
2018年06月1日 14:20

哈哈哈，图文并茂，喜欢。

回复

15. 小北方说道：
2018年05月31日 08:58

谢谢分享

回复

16. 测过说道：
2018年05月18日 16:26

博主。。。能不能在写博文的时候，暂时把你脑子里黄色调调关起来先？每次看博主的博客，都要小心翼翼的，跟做贼似的，重点是我是个妹子啊啊啊啊啊啊！！！！嗯，最后一点，文章讲解得很好。

回复

> 但是说道：
> 2018年05月26日 13:58
>
> 漂亮么
>
> 回复
>
> > **benjamin说道：**
> > 2018年05月27日 19:45
> >
> > 我都有点受不了了，特别是，理解perspective-origin的时候
> >
> > 回复

> **dd说道：**
> 2018年06月14日 17:22
>
> 同意。。。面红耳赤的看完了 o(╥﹏╥)o
>
> 回复

> 大胡子说道：
> 2018年07月26日 15:07
>
> 哈哈哈，当小黄书看吧
>
> 回复

> **huoshen说道：**
> 2018年07月31日 12:06
>
> 同意，有点做贼的感觉
>
> 回复

> 妹子说道：
> 2018年08月1日 16:32
>
> 你啊啊啊啊啊啊！！！！嗯，，这个叫是什么意思，，，

回复

顶说道：
2018年08月6日 12:14

顶顶。。。在医院看的，护士从旁边过去，看我的恨尴尬。

回复

17. 如辰说道：
2018年05月5日 09:21

看不懂

回复

18. 上下而求索说道：
2018年04月21日 16:13

易懂易学，前端之路有你这样的领路人真是太幸福了。

回复

« 先前评论

最新文章

» 常见的CSS图形绘制合集
» 粉丝群第1期CSS小测点评与答疑
» 分享三个纯CSS实现26个英文字母的案例
» 小tips: 纯CSS实现打字动画效果
» CSS/CSS3 box-decoration-break属性简介
» CSS :placeholder-shown伪类实现Material Design占位符交互效果
» 从天猫某活动视频不必要的3次请求说起
» CSS vector-effect与SVG stroke描边缩放
» CSS ::backdrop伪元素是干嘛用的?
» 周知：CSS -webkit-伪元素选择器不再导致整行无效

今日热门

» 常见的CSS图形绘制合集 (193)
» 未来必热：SVG Sprite技术介绍 (120)
» 粉丝群第1期CSS小测点评与答疑 (115)
» HTML5终极备忘大全（图片版+文字版） (93)
» 让所有浏览器支持HTML5 video视频标签 (86)
» Selectivizr-让IE6~8支持CSS3伪类和属性选择器 (82)
» CSS3下的147个颜色名称及对应颜色值 (80)
» 视区相关单位vw, vh..简介以及可实际应用场景 (77)
» 写给自己看的display: flex布局教程 (76)
» 小tips: 纯CSS实现打字动画效果 (76)

今年热议

» 《CSS世界》女主角诚寻靠谱一起奋斗之人 (76)
» 不借助Echarts等图形框架原生JS快速实现折线图效果 (64)
» 看，for..in和for..of在那里吵架！ (60)
» 是时候好好安利下LuLu UI框架了！ (47)
» 原来浏览器原生支持JS Base64编码解码 (35)
» 妙法攻略：渐变虚框及边框滚动动画的纯CSS实现 (33)
» 炫酷H5中序列图片视频化播放的高性能实现 (31)
» CSS scroll-behavior和JS scrollIntoView让页面滚动平滑 (30)
» windows系统下批量删除OS X系统.DS_Store文件 (26)
» 写给自己看的display: flex布局教程 (26)

猜你喜欢

- CSS CSS3实现3D开门动画效果
- 小tip: 纯CSS实现视差滚动效果
- Safari 3D transform变换z-index层级渲染异常的研究
- cssSandpaper-兼容IE的CSS3 JavaScript库
- web上渐进使用jQuery Mobile中animate相关CSS
- 基于canvas画布的两个炫酷效果展示
- CSS3 animate实现图片墙3D翻转效果
- 使用CSS3绘制我们的太阳系
- CSS3 Transitions, Transforms和Animation使用简介与应用展示
- 理解CSS3 transform中的Matrix(矩阵)
- 翻译：即将到来的CSS私有前缀灾难