页面动画效果研究  
一、近期使用的动画效果  
简述近期在项目中应用的动画效果  
1、2D变换效果：  
 主要是渐变、位移的变换，称之为过渡效果更合适。从一个状态过渡到另一个状态，实现方法简单：javascript模拟帧、css3实现。模拟帧利用javascript定时器实现，CSS3利用新特性实现。  
 所有的过渡效果在2维平面内进行，在平面内完成位移、透明度、字体大小等CSS数值属性的变换。

渐变：opacity的改变；位移：x/y轴的坐标移动

2、3D立体空间变换效果

所有的变换效果都以假想的三维空间为舞台，涉及到x，y，z以及透视的变换（简单理解为近大远小）

#### 二、动画效果的实现

2.1传统javascript动画的实现（逐帧动画，人眼帧数达到24以上就会感觉流畅）：

动画时长=播放总帧数\*间隔时间

定时器（setTimeout/setInterval）依赖于浏览器内置时钟的刷新，而不同的浏览器刷新频率并不相同，再加上同步代码优先于异步代码执行的特性，无法保证每帧执行的时间间隔，导致了帧数间隔的不可控。

内置时钟的刷新：  
(定时器的计算依靠的是浏览器的内置时钟，而时钟的精确度又取决于时钟更新的频率(Timer resolution)。IE8及其之前的IE版本更新间隔为15.6毫秒。假设你设定的setTimeout延迟为16.7ms，那么它要更新两个15.6毫秒才会该触发延时。这也意味着无故延迟了 15.6 x 2 - 16.7 = 14.5毫秒。)

同步优先于异步：

var curDate = new Date().getTime();

var dif = 0;

setTimeout(function(){

alert(new Date().getTime()-curDate);

},250)

while(dif < 2000){

dif = new Date().getTime()-curDate;

}

1、即使向其传递毫秒为单位的参数，它们也不能达到ms的准确性。这是因为javascript是单线程的，可能会发生阻塞。  
2、没有对调用动画的循环机制进行优化。  
3、没有考虑到绘制动画的最佳时机，只是一味地以某个大致的事件间隔来调用循环。  
其实，使用setInterval或setTimeout来实现主循环，根本错误就在于它们抽象等级不符合要求。我们想让浏览器执行的是一套可以控制各种细节的api，实现如“最优帧速率”、“选择绘制下一帧的最佳时机”等功能。但是如果使用它们的话，这些具体的细节就必须由开发者自己来完成。

由于帧数的间隔不可控，实现动画出现了两种选择：保证动画帧数，对播放时间不作处理--基于帧数的动画；保证动画时长，牺牲掉帧数--基于时间的动画；一般采用丢帧保时的方法。

2.2 保时动画与保帧动画的两个例子

例子：（custome.html）Jquery动画的实现--采用了丢帧保时方法。

2.3 javascript动画并不与jquery动画划等号，javascript动画并不慢。Jquery动画慢的原因：  
 jquery并不是专用的动画引擎库，它的强大是DOM节点的操作  
 jquery动画不可避免的导致重绘  
 jquery采用定时器，定时器本身的局限性

为了完善定时器动画的缺陷，javascript提供了动画API：requestAnimationFrame

requestAnimationFrame,请求帧动画。该方法不在需要我们手动指定动画帧，而是依据浏览器的重绘频率取得合适的时间间隔来运行动画。

特性：

1. 会集中每帧的DOM操作，在一次reflow中完成全部重绘，reflow的时间依据浏览器的刷新频率而定。减少了重绘次数，集中操作，解决了不同浏览器的刷新频率不同的问题。
2. 隐藏、不可见元素，不进行重绘。（更少内存，GPU的使用）

例子：custome.html

2.4 CSS3动画的实现

1. 用于实现动画的属性  
   transition/animation，transform只是CSS3提供的元素变形属性，是静态的。  
   transition：过渡（变换），由一种状态过渡为另外了一种。  
   animation：动画，多个状态之间的转换。  
   小知识点：动画可分为逐帧动画、补间动画。定时器动画是逐帧动画，帧与帧之间没有任何处理；CSS3常用的动画为补间动画，关键帧之间由计算机计算并渲染。  
   例子：step.html
2. 完成3D动画  
   关键属性：perspective(视点)、transform-style:preserve-3d，例子：perspective.html

transform/rotateX/rotateY/translate3d…

如果仅仅只是将这些变形元素应用到DOM元素中，达不到我们所需要的3D效果。在现实空间中，人眼所看到的立体物体，要应用到平面上需要应用到透视学。而在浏览器中是通过perspective属性实现的。

通过CSS3完成3D效果关键属性：transform-style:preserve-3d,perspective

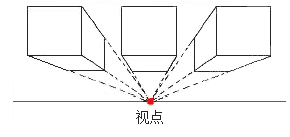
perspective 近大远小的效果  
 preserve-3d保留其子元素的3D转换

perspective透视、视点：

与传统中提到物体透视不同，CSS3中的透视点在浏览器的前方（在浏览器的上方），

相当于三维空间中眼睛的位置。

传统的视点



Css3中的视点：假想浏览器内部为立体空间，眼睛位于浏览器前方向里看，数值则是假想的眼睛距离DOM元素的距离。

浏览器

视点

需要一个理解视点的例子。

知道视点是在浏览器上后，结合视点展示一个立体变换的例子

3、一个简单的3D立体翻转的例子讲解

thirdD.html

1. js与CSS3对比  
   性能：jquery动画不代表javascript的动画性能  
   控制粒度：js能提供更细粒度的控制  
   多状态动画：CSS3对两个状态的切换简单好用，但是涉及到多个状态时则要书写数目繁多的百分比。  
   兼容性：CSS3兼容性并不友好

三、js与css3相结合提高可控性  
 js与CSS3相结合改善了CSS3动画的操作空性。  
 js可监控CSS3的开始（animationStart）和结束（animationEnd/transitionEnd）事件.

关于如何获取CSS3动画变化过程中的状态：getComputedStyle/styleSheets