Js 动画和css动画的研究

以本次项目在电视端的实现效果为例初略解析原生js动画和jq动画及css动画的效果及优劣。

实现方式：

1. 原生js动画

主要代码：

timer1 = setInterval(function(){

var speed = speed0;

if(alpha1 <= 0){

clearInterval(timer1);

$("#bigbg1").css("z-index","-4");

$("#bigbg2").css("z-index","-1");

$("#bigbg3").css("z-index","-2");

$("#bigbg4").css("z-index","-3");

$("#bigbg1").css("opacity","1");

alpha1 = 1;

}else{

alpha1 = (alpha1\*10 - speed\*10)/10;

oImg1.style.opacity = alpha1;

oImg1.style.filter = "alpha(opacity="+ alpha1\*100 +")";

}

},t0);

原理：使用定时器，控制每一个t0时间点的元素的透明度，已达到淡入淡出的效果。

1. jq 动画

$("#myImg1").fadeOut("slow", function () {

$("#myImg1").css("z-index","-4");

$("#myImg2").css("z-index","-1");

$("#myImg3").css("z-index","-2");

$("#myImg4").css("z-index","-3");

$("#myImg1").css("display","block");

});

原理，调用jq的fadein和fadeout方法，实现原理和原生js类似，但在渲染上做了优化，视觉上比原生js顺畅

1. css动画

function fadeIn(e1,e2) {

e1.className = "bigboxs";

e2.className = "myImgs";

};

function fadeOut(e1,e2) {

e1.className = "bigboxs fadeout";

e2.className = "myImgs fadeout";

};

.myImgs{-webkit-transition: opacity 3s linear;-moz-transition: opacity 3s linear;-o-transition: opacity 3s linear;transition: opacity 3s linear;opacity:1;}

.fadein {opacity: 100;filter: alpha(opacity=100);}

.fadeout {opacity: 0;filter: alpha(opacity=0);}

利用css控制元素的透明度从1-0的过渡的方式来实现淡入淡出的效果。视觉上优于原生js和jq的淡入淡出方法。

优劣解析：

从代码的实现可以发现，js实现动画的时候，js能对动画实现极高程度的控制，可以精确到每一个t0时刻的动画效果，如此能做出功能复杂的动画，比如曲线运动,冲击闪烁,视差滚动效果，css只能控制整个过程的完成时间，完成方式，并不能左右某个时间点的动画效果，整体的动画样式单一。但是总体上的淡入淡出效果，确实css动画优于jq的fadein、fadeout优于原生js，在某些低配的机器上，原生js和jq实现的动画，播放时出现卡顿效果严重，css动画却很顺畅。

原理初步解析：

渲染线程分为main thread(主线程)和compositor thread(合成器线程)。

Javascript是在浏览器的主线程（main thread）中运行，而主线程中还有其它需要运行的JavaScript脚本、样式计算、布局、绘制任务等,对其干扰导致线程可能出现阻塞，从而造成丢帧的情况，这个极大影响到视觉上的效果，也就是卡顿。

Css动画在淡入淡出上只是改变了的元素的opacity。css动画在只改变transform和opacity的情况下，整个CSS动画得以在compositor thread完成（而JS动画则会在main thread执行，然后触发compositor进行下一步操作）  
 相关辅助原理

1. 浏览器使用与 requestAnimationFrame 类似的机制，requestAnimationFrame比起setTimeout，setInterval设置动画的优势主要是:1)requestAnimationFrame 会把每一帧中的所有DOM操作集中起来，在一次重绘或回流中就完成,并且重绘或回流的时间间隔紧紧跟随浏览器的刷新频率,一般来说,这个频率为每秒60帧。2)在隐藏或不可见的元素中requestAnimationFrame不会进行重绘或回流，这当然就意味着更少的的cpu，gpu和内存使用量。
2. 在主线程中，维护了一棵Layer树（LayerTreeHost），管理了TiledLayer，在compositor thread，维护了同样一颗LayerTreeHostImpl，管理了LayerImpl，这两棵树的内容是拷贝关系。因此可以彼此不干扰，当Javascript在main thread操作LayerTreeHost的同时，compositor thread可以用LayerTreeHostImpl做渲染。当Javascript繁忙导致主线程卡住时，合成到屏幕的过程也是流畅的。为了实现防假死，鼠标键盘消息会被首先分发到compositor thread，然后再到main thread。这样，当main thread繁忙时，compositor thread还是能够响应一部分消息，例如，鼠标滚动时，加入main thread繁忙，compositor thread也会处理滚动消息，滚动已经被提交的页面部分（未被提交的部分将被刷白）。

Js动画

优点：

1. JavaScript动画控制能力很强, 可以在动画播放过程中对动画进行控制：开始、暂停、回放、终止、取消等等。
2. 动画效果比css3动画丰富,有些动画效果，比如曲线运动,冲击闪烁,视差滚动效果，只有JavaScript动画才能完成
3. CSS3有兼容性问题，而JS大多时候没有兼容性问题

缺点：

1)JavaScript在浏览器的主线程中运行，受到其他JavaScript脚本任务的影响,对其干扰导致线程可能出现阻塞，从而造成丢帧。

2)代码的复杂度高于CSS动画

Css动画

优点：

1）浏览器可以对动画进行优化。compositor thread(合成器线程)

2) 强制使用硬件加速 （通过 GPU 来提高动画性能）

缺点：

1. 运行过程控制较弱,无法附加事件绑定回调函数。CSS动画只能暂停,不能在动画中寻找一个特定的时间点，不能在半路反转动画，不能变换时间尺度，不能在特定的位置添加回调函数或是绑定回放事件,无进度报告
2. 代码冗长。想用 CSS 实现稍微复杂一点动画,最后CSS代码都会变得非常笨重。

## **结论**

 如果动画只是简单的状态切换，不需要中间过程控制，在这种情况下，css动画是优选方案。它可以让你将动画逻辑放在样式文件里面，而不会让你的页面充斥 Javascript 库。然而如果你在设计很复杂的富客户端界面或者在开发一个有着复杂UI状态的 APP。那么你应该使用js动画，这样你的动画可以保持高效，并且你的工作流也更可控。所以，在实现一些小的交互动效的时候，就多考虑考虑CSS动画。对于一些复杂控制的动画，使用javascript比较可靠。

另，CSS动画比JS流畅的前提：

JS在执行一些昂贵的任务

同时CSS动画不触发layout或paint

在CSS动画或JS动画触发了paint或layout时，需要main thread进行Layer树的重计算， 这时CSS动画或JS动画都会阻塞后续操作。

只有如下属性的修改才符合“仅触发Composite，不触发layout或paint”：

backface-visibility、opacity、perspective、perspective-origin、transfrom

所以只有用上了3D加速或修改opacity时，css3动画的优势才会体现出来。

代码相对简单,性能调优方向固定

对于帧速表现不好的低版本浏览器，CSS3可以做到自然降级，而JS则需要撰写额外代码