## 爱心鱼

**requestAnimationFrame**我所做的事情很简单，跟着浏览器的绘制走，如果浏览设备绘制间隔是16.7ms，那我就这个间隔绘制；如果浏览设备绘制间隔是10ms, 我就10ms绘制。这样就不会存在过度绘制的问题，动画不会掉帧，自然流畅的说~~（兼容的代码）

/\* requestAnimationFrame.js

\* by zhangxinxu 2013-09-30

\*/

(function() {

var lastTime = 0;

var vendors = ['webkit', 'moz'];

for(var x = 0; x < vendors.length && !window.requestAnimationFrame; ++x) {

window.requestAnimationFrame = window[vendors[x] + 'RequestAnimationFrame'];

window.cancelAnimationFrame = window[vendors[x] + 'CancelAnimationFrame'] || // name has changed in Webkit

window[vendors[x] + 'CancelRequestAnimationFrame'];

}

if (!window.requestAnimationFrame) {

window.requestAnimationFrame = function(callback, element) {

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

var timeToCall = Math.max(0, 16.7 - (currTime - lastTime));

var id = window.setTimeout(function() {

callback(currTime + timeToCall);

}, timeToCall);

lastTime = currTime + timeToCall;

return id;

};

}

if (!window.cancelAnimationFrame) {

window.cancelAnimationFrame = function(id) {

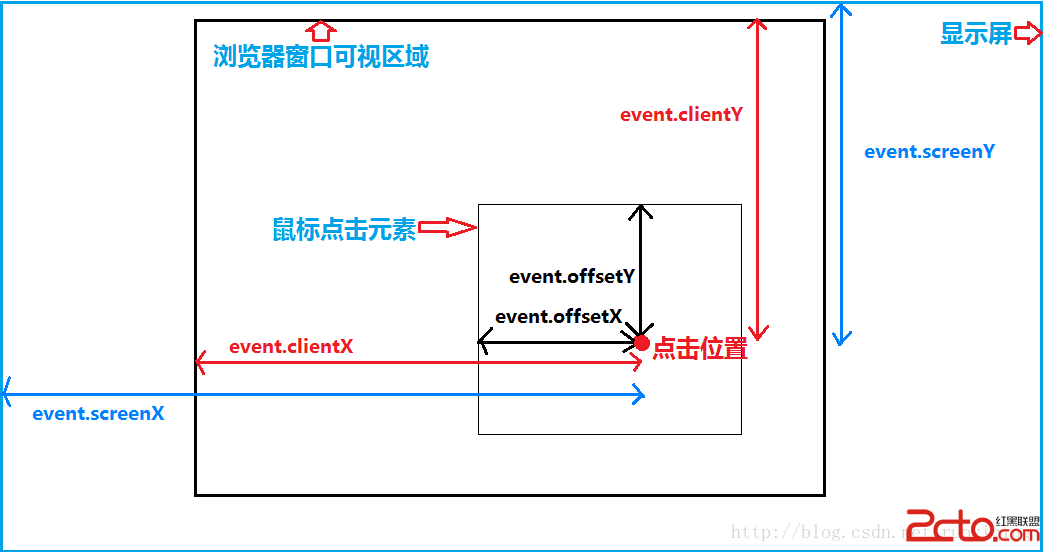
clearTimeout(id);

};

}

}());

chrome：各个参数值



e.pageX——相对整个页面的坐标  
e.layerX——相对当前坐标系的border左上角开始的坐标  
e.offsetX——相对当前坐标系的border左上角开始的坐标//offsetx：设置或者是得到鼠标相对于目标事件的父元素的内边界在x坐标上的位置  
e.clientX——相对可视区域的坐标  
e.x——相对可视区域的坐标

ff：

e.pageX——相对整个页面的坐标  
e.layerX——相对当前坐标系的border左上角开始的坐标  
e.offsetX——无  
e.clientX——相对可视区域的坐标  
e.x——无

opera：

e.pageX——相对整个页面的坐标  
e.layerX——无  
e.offsetX——相对当前坐标系的内容区域左上角开始的坐标  
e.clientX——相对可视区域的坐标  
e.x——相对可视区域的坐标

safari：（这个和chrome是一样的）

e.pageX——相对整个页面的坐标  
e.layerX——相对当前坐标系的border左上角开始的坐标  
e.offsetX——相对当前坐标系的border左上角开始的坐标  
e.clientX——相对可视区域的坐标  
e.x——相对可视区域的坐标

IE9：

e.pageX——相对整个页面的坐标  
e.layerX——相对当前坐标系的border左上角开始的坐标 + 滚动条滚过的距离（这个NB轰轰了····=。=）  
e.offsetX——相对当前坐标系的内容区域左上角开始的坐标  
e.clientX——相对可视区域的坐标  
e.x——相对当前坐标系的border左上角开始

IE8：

e.pageX——无  
e.layerX——无  
e.offsetX——相对当前坐标系的内容区域左上角开始的坐标  
e.clientX——相对可视区域的坐标  
e.x——相对当前坐标系的border左上角开始

IE7：

e.pageX——无  
e.layerX——无  
e.offsetX——相对当前坐标系的内容区域左上角开始的坐标  
e.clientX——相对可视区域的坐标  
e.x——相对当前坐标系的border左上角开始

IE6：

e.pageX——无  
e.layerX——无  
e.offsetX——相对当前坐标系的内容区域左上角开始的坐标  
e.clientX——相对可视区域的坐标  
e.x——相对当前坐标系的border左上角开始

2.PageX和clientX  
pageX指鼠标在页面上的位置，以页面左侧为参考点  
clientX指可视区域内离左侧的距离，以滚动条滚动到的位置为参考点。各个浏览器相同。  
即当有滚动条时clientX  小于  pageX

//ie678不识别pageX  
PageY=clientY+scrollTop-clientTop;(只讨论Y轴,X轴同理,下同)  
页面上的位置=可视区域位置+页面滚动条切去高度-自身border高度,还是直接上图比较清楚

3、screenX  
鼠标在屏幕中的位置，指的是鼠标到电脑屏幕左侧的距离。 各个浏览器相同。  
与clientX的区别是clientX是到浏览器的距离。  
 例如：当网页缩小，拖动到屏幕中间时，screnX 大于 clientX

4、offsetX和layerX  
为了兼容浏览器，layerX是FF、chrome识别，offsetX是除了FF之外。

如果触发元素设置了position，则offsetX等于layerX  
如果页面有滚动条，添加滚动的距离。

layerX:FF特有，是相对于父元素的位置，鼠标相对于“触发事件的元素的层级关系中离该元素最近的，设置了position的父元素”的边界的位置，从border的左上角开始定位，即如果这个父

元素存在border，则坐标原点在border的左上角，而不是内容区域的左上角。  
offsetX:IE特有，鼠标相对于“触发事件的元素”的位置，从内容区域左上角开始定位，不是从border左上角开始！这个属性比较好用，用来判断鼠标点在一个元素中的哪个位置很方便，FF

没有直接替换的属性。

## 注意点：

1. 如果是静态的东西在canvas中则画一次就可以；但是如果是多帧的动画时要放在循环中；
2. 绘制海葵所用到的API：beginPath（） ；closePath（）；strokeStyle（）；stroke（）；

lineWidth（）；lineCap ；globalAlpha（设置透明度）；

1. 绘制果实所用到的API：drawImage（）；
2. 绘制大鱼所用到的API：translate（）；rotate（）；Math.atan2（y，x）；
3. 大鱼吃果实原理：检测大鱼和果实之间的距离；三角函数；
4. drawImage() 方法在画布上绘制图像、画布或视频。

drawImage() 方法也能够绘制图像的某些部分，以及/或者增加或减少图像的尺寸。

JavaScript 语法 1

在画布上定位图像：

context.drawImage(img,x,y);