1. 实现发红包看照片的思路：

首先在最底下放置一张高清图片，css3 filter：blur(10px)设置模糊效果，再在图片上放置canvas,放置一张高清原图片,与原先放置的图片大小完全一样，再用clip()方法剪辑出一个小圆圈出来 则可以实现初始效果。显示图片时 将clip()剪辑区域的半径变大，包围整个图片则可以显示整个高清图片。动画效果可以通过剪辑区域逐渐增加的方式来创建动画效果。

1. Canvas 的save 和 restore() 的作用：

save：用来保存Canvas的状态。save之后，可以调用Canvas的平移、放缩、旋转、错切、裁剪等操作。 restore：用来恢复Canvas之前保存的状态。防止save后对Canvas执行的操作对后续的绘制有影响。

对canvas中特定元素的旋转平移等操作实际上是对整个画布进行了操作，所以如果不对canvas进行save以及restore，那么每一次绘图都会在上一次的基础上进行操作，最后导致错位。比如说你相对于起始点每次30度递增旋转，30，60，90.如果不使用save 以及 restore 就会变成30, 90, 150，每一次在前一次基础上进行了旋转。save是入栈，restore是出栈。

2 canvas画剪辑区域

Eg canvas 画一个圆的剪辑区域

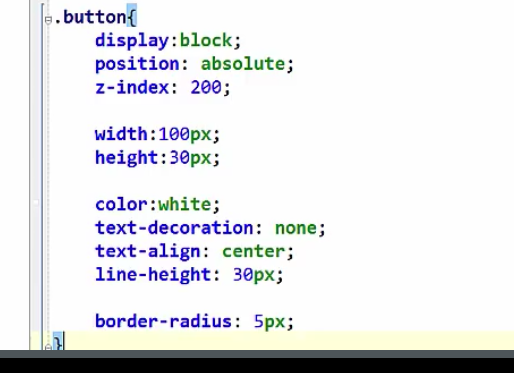
ctx.beginPath();

ctx.arc(clippingRegion.x,clippingRegion.y,clippingRegin.r,0,2\*Math.PI,false);

ctx.clip();

ctx.drawImage(image,0,0);//此时就只会在剪辑的画布上定位图像

3 css里面样式的顺序可以参考：

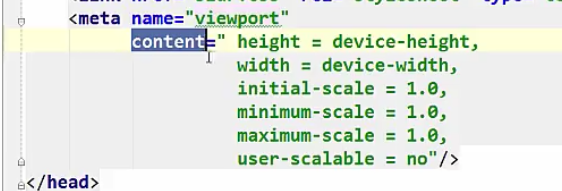
1. 

用setInterval()时要记住使用clearInterval() 来清除动画，把setInterval 的timer 设置为全局变量，在初始化时记得清除interval

4 让圆圈显示在canvas内；

clippingRegion={x:Math.random(0,1)\*(canvas.width-2\*radius)+radius, y:Math.random(0,1)\*(canvas.height-2\*radius)+radius, r:radius};

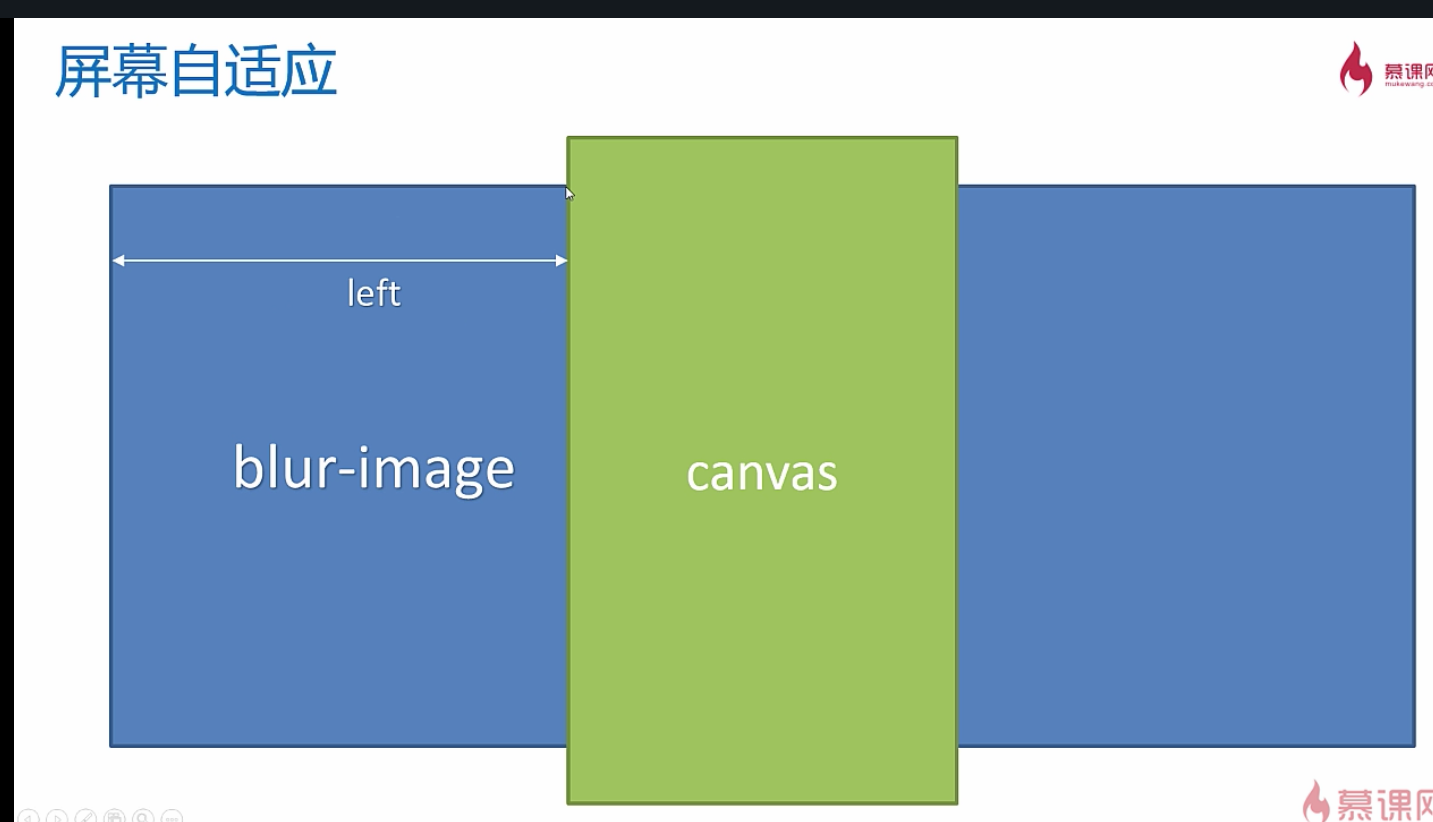
5 自适应屏幕



iPad 可能出现的情况：

leftMargin = (image.width-canvas.width)/2;

topMargin = (image.height-canvas.height)/2;



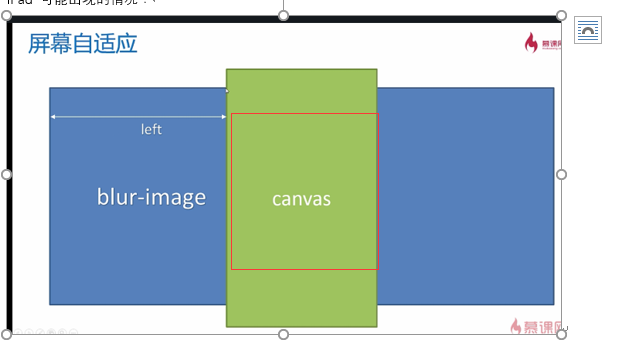
此时

ctx.drawImage(image,leftMargin,-topMargin,canvas.width,canvas.height,0,0,

canvas.width,canvas.height);//在画布上定位图像



假设图片远远大于canvas的大小：



ctx.drawImage(image,leftMargin,topMargin,canvas.width,canvas.height,0,0,canvas.width,canvas.height);//在画布上定位图像