CANVAS

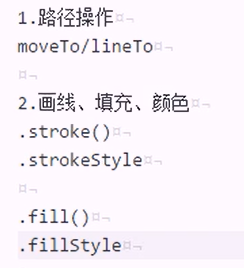
Canvas只能用属性来改变大小，（样式修改只会拉大canvas整体，不会空大绘图区）

Canvas坐标点从画布左上角开始

Canvas里的图形只要画完了，是不能利用left之类的来修改的

Canvas性能高，一部分原因取决于它不保存任何图形的信息

.getContext(‘2d’) 获取图形上下文---绘图所需接口



beginPath –清除之前所有的路径

closePath-闭合路径，封口

.rect(x,y,w,h)矩形 .strokeRect画线矩形 .fillRect 填充矩形

.arc(cx,cy,r,starAng,endAng，是否逆时针)圆心x,圆心y，半径，起始角度，结束角度，

是否逆时针

角度：满圈360度 0度在上边，-90度为弧度

弧度：满圈2PI 计算机使用 0度在右边

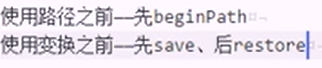
角度转弧度： 角度\*PI/180

ClearReat（x,y,w,h） 清除矩形\

Canvas中的变化都是叠加的

.save()保存当前画布的状态（颜色，线宽，旋转，变换，唯一不包括画布内的图形）

.restore()将还原至上一次保存的状态



Canvas中的变幻执行顺序是反着来的（先执行下面，后执行上面的代码）

Canvas图片加载

精简版：.drawImage(图片,x,y)

完整版：drawImage(图片,图片X,图片Y，图片W，图片H，

画布X，画布Y，画布W，画布H)

Canvas像素级操作

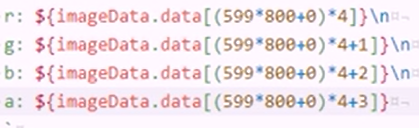
一个像素占4位 r g b a 都是0-255

Canvas是不允许跨域操作图片信息的

.getImageData(x,y,w,h) 获取像素区

.imageData.data[像素位] 获取某一像素信息

例：想获取某一像素，需在像素位中填入该像素前的像素数量，比如800\*600的图片，获取该图片的最后一行第一列的像素位： data[(r\*w+c)\*4]

，

599是前面有599行，800是列数，0是前面有0位

4是rgba

PutImageData(imageData,x,y) 放回修改区

.toDataUrl 导出当前图片(base64)，注意：此方法是canvas自己身上的

requestAnimationFrame 向浏览器请求下一帧的所需的时间，通过递归来时间自动请求