1. 红宝书第15章-使用Canvas绘图

<canvas id="myCanvas" width="200" height="200">A drawing of something</canvas>

<script>

var oCanvas=document.getElementById("myCanvas");

//确定浏览器支持canvas元素

if(oCanvas.getContext){

var context=oCanvas.getContext("2d");

//更多代码

}

</script>

1. 填充和描边

if(oCanvas.getContext){

var context=oCanvas.getContext("2d");

context.strokeStyle="red";

context.fillStyle="#0000ff";

}

fillStyle填充 strokeStyle描边

这2个属性的值可以是字符串、渐变对象、模式对象

1. 绘制矩形

填充矩形：

//绘制红色矩形

context.fillStyle="red";

//矩形的x坐标，矩形的y坐标，矩形的宽度，矩形的高度

context.fillRect(10,10,50,50);

//绘制半透明的蓝色矩形

context.fillStyle="rgba(0,0,255,0.5)";

context.fillRect(30,30,50,50);

描边矩形

//绘制红色描边矩形

context.strokeStyle="red";

//矩形的x坐标，矩形的y坐标，矩形的宽度，矩形的高度

context.strokeRect(10,10,50,50);

//绘制半透明的蓝色描边矩形

context.strokeStyle="rgba(0,0,255,0.5)";

context.strokeRect(30,30,50,50);

清除一个矩形区域

//绘制红色矩形

context.fillStyle="red";

//矩形的x坐标，矩形的y坐标，矩形的宽度，矩形的高度

context.fillRect(10,10,50,50);

//绘制半透明的蓝色矩形

context.fillStyle="rgba(0,0,255,0.5)";

context.fillRect(30,30,50,50);

//清除一个小矩形

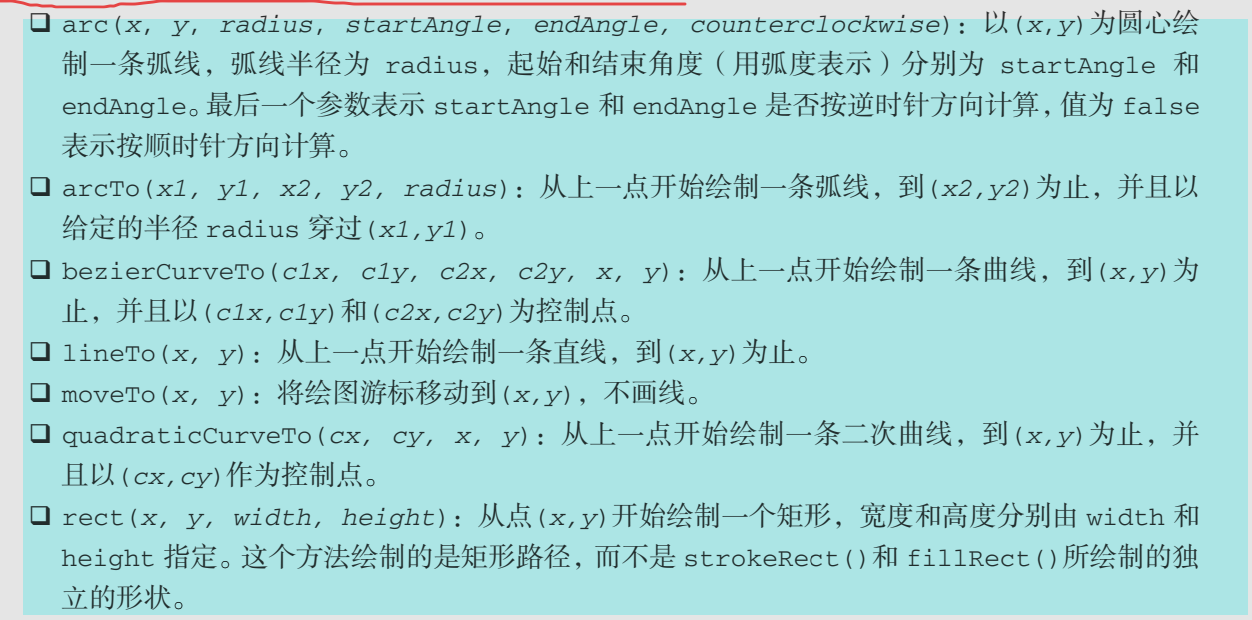
context.clearRect(40,40,10,10);

1. 绘制路径

首先，调用beginPath()

然后，绘制路径

最后，closePath()/fill()/stroke()/clip()



//开始路径

context.beginPath();

//绘制外圆

context.arc(100,100,99,0,2\*Math.PI,false);

//绘制内圆

context.moveTo(194,100);

context.arc(100,100,94,0,2\*Math.PI,false);

//绘制分针

context.moveTo(100,100);

context.lineTo(100,15);

//绘制时针

context.moveTo(100,100);

context.lineTo(35,100);

//描边路径

context.stroke();

context.isPointInPath(x,y);//用于在路径关闭之前确定画布的某一点是否位于路径上

1. 绘制文本

fillText() strokeText()

属性：

font 文本样式、大小及字体

textAlign 文本对齐方式 start center end

textBaseline 文本的基线 top middle bottom等

//正常

context.font="bold 14px Arial";

context.textAlign="center";

context.textBaseline="middle";

context.fillText("12",100,10);

//起点对齐

context.textAlign="start";

context.fillText("12",100,20);

//终点对齐

context.textAlign="end";

context.fillText("12",100,30);

var fontSize=100;

context.font=fontSize+"px Arial";

//在一个140px宽的矩形中绘制文本

while(context.measureText("Hello World").width>140){

fontSize--;

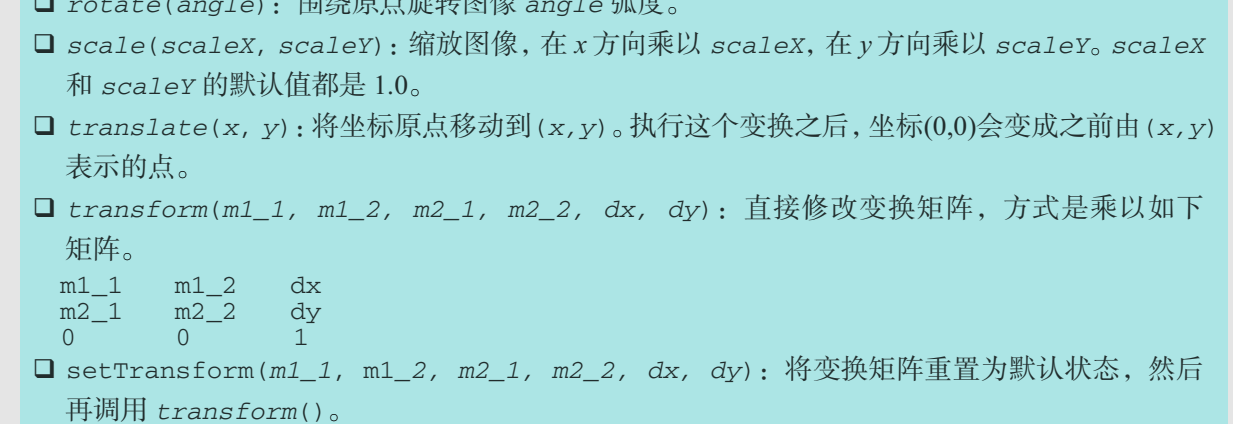
context.font=fontSize+"px Arial";

}

context.fillText("Hello World",0,30);

context.fillText("fontsize:"+fontSize+"px",0,50);

1. 变换



save()保存当前绘图上下文

restore()回到上一个保存的

1. 绘制图像

var image=document.images[0];

//(img,x,y)

context.drawImage(image,10,10);

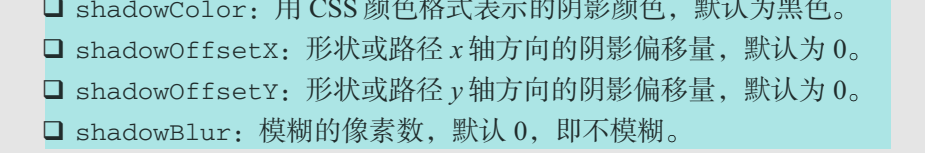
//(img,x,y,width,height)

context.drawImage(image,50,10,20,30);

//(img,x1,y1,width1,height1,x2,y2,width2,height2)

context.drawImage(image,0,10,50,50,0,100,40,60);

1. 阴影



1. 渐变

线性渐变

var gradient=context.createLinearGradient(30,30,70,70);

gradient.addColorStop(0,"white");

gradient.addColorStop(1,"black");

//绘制红色矩形

context.fillStyle="red";

context.fillRect(10,10,50,50);

//绘制渐变矩形

context.fillStyle=gradient;

context.fillRect(30,30,50,50);

径向渐变

//(x1,y1,r1,x2,y2,r2);

var gradient=context.createRadialGradient(55,55,10,55,55,30);

gradient.addColorStop(0,"white");

gradient.addColorStop(1,"black");

//绘制红色矩形

context.fillStyle="red";

context.fillRect(10,10,50,50);

//绘制渐变矩形

context.fillStyle=gradient;

context.fillRect(30,30,50,50);

1. 模式

pattern=context.createPattern(image,”repeat”);

1. 使用图像数据

getImageData(x,y,width,height)取得图像数据

每个imageData对象有3个属性：width height data data是一个数组：红绿蓝透明度

应用：彩色图像变灰白

1. 合成

globalAlpha全局透明度

globlCompositionOperation后绘制的图形怎么与先绘制的图像结合

1. 博客
2. css层叠

优先级：内联样式-id-class/属性选择/伪类-元素/伪元素-结合符/通配选择器

1. 重要声明

!important优先级最高

1. 继承样式

a标签不会继承父元素的文本样式

1. 手风琴效果

每个li有一个初始宽度 25%

hover某li时，此li变宽，其他变窄

离开时，恢复原来的宽度