

后盾人 人人做后盾

www.houdunwang.com

Canvas

后盾人 2011-2017

什么是canvas：

- <canvas> 标签定义图形，比如图表和其他图像。
- <canvas> 标签只是图形容器，您必须使用脚本来绘制图形。
- canvas 其实对于HTML来说很简单，只是一个标签元素而已，自己并没有行为，但却把一个绘图 API 展现给客户端 JavaScript 以使脚本能够把想绘制的东西都绘制到一块画布上，拥有绘制路径，矩形，圆，字符以及图像等功能。所有的标签只是图形的容器，必须使用JavaScript的 API 操作绘图。

标签：

- <canvas id= "canvas" width= "500" height= "500" ></canvas>

getContext

- 返回一个用于在画布上绘图的环境

```
<script type="text/javascript">  
  c = document.getElementById("canvas");  
  context= c.getContext('2d');  
</script>
```

什么是canvas

矩形

- `context.fillRect(x,y,width,height)` 绘制“被填充”的矩形
- `context.strokeRect(x,y,width,height)` 绘制矩形（无填充）
- `context.clearRect(x,y,width,height)` 在给定的矩形内清除指定的像素

颜色、样式

- `context.fillStyle= '#f00f00'` 设置或返回填充绘画的颜色、渐变或模式
- `context.strokeStyle= 'green'` 设置或返回笔触的颜色、渐变或模式
- `context.lineWidth=10` 设置或返回当前的线条宽度
- `context.lineJoin= "边界类型"` bevel:斜角,round:圆角,miter:尖角

canvas方法或属性

路径

- `beginPath()` 开始一条路径，或重置当前路径
- `closePath()` 创建从当前点回到起始点的路径(闭合路径)
- `moveTo(x,y)` 把路径移动到画布中的指定点，不创建线条
- `lineTo(x,y)` 添加一个新点，创建从该点到最后指定点的线条
- `fill()` 填充当前绘图（填充路径）
- `stroke()` 绘制已定义的路径（连线路径）

封闭状态

- `save()` 保存当前环境的状态
- `restore()` 返回之前保存过的路径状态和属性

canvas方法或属性

- `context.scale(scalewidth,scaleheight)` 缩放处理 1=100%
- `context.translate(x,y)` 图形位置处理
- `context.rotate(angle)` 旋转画布,单位：弧度，默认以画布为圆心旋转
- 弧度计算公式：角度*Math.PI/180
- 比如60角度相当于 $60 * \text{Math.PI} / 180$ 弧度

画布控制

- `context.arc(x,y,r,sAngle,eAngle,counterclockwise)` 创建弧/曲线
(用于创建圆形或部分圆)

参数说明：

x 圆的中心的 x 坐标。

y 圆的中心的 y 坐标。

r 圆的半径。

sAngle 起始角，以弧度计。（弧的圆形的三点钟位置是 0 度）

eAngle 结束角，以弧度计。

counterclockwise 可选。False = 顺时针，true = 逆时针。

canvas方法或属性

设置字体属性

- `context.font="40px Arial"`

在画布上绘制“被填充的”文本

- `context.fillText(text,x,y,maxWidth);`

在画布上绘制文本（无填充）

- `context.strokeText(text,x,y,maxWidth)`

设置对齐方式

- `context.textAlign= "left | right | center"`

设置文字基线

- `context.textBaseline= "top | middle | bottom" ;`

获取文本宽度

- `context.measureText(text);`

文本控制

向画布上绘制图像、画布或视频

语法 1: 在画布上定位图像：

```
context.drawImage(img,画布x坐标,画面y坐标);
```

语法 2: 在画布上定位图像，并规定图像的宽度和高度

```
context.drawImage(img,画布x坐标,画面y坐标,图片width,图片height);
```

语法 3: 剪切图像，并在画布上定位被剪切的部分

```
context.drawImage(img,sx,sy,swidth,sheight,x,y,width,height);
```

参数：

img 规定要使用的图像、画布或视频。

sx 可选。开始剪切的 x 坐标位置。

sy 可选。开始剪切的 y 坐标位置。

swidth 可选。被剪切图像的宽度。

sheight可选。被剪切图像的高度。

x 可选。在画布上放置图像的 x 坐标位置。

y 可选。在画布上放置图像的 y 坐标位置。

width可选。要使用的图像的宽度。（伸展或缩小图像）

height 可选。要使用的图像的高度。（伸展或缩小图像）

图像控制

在水平和垂直方向重复图像

- `context.createPattern(image,"repeat|repeat-x|repeat-y|no-repeat")`

获取画布矩形区域像素信息

- `context.getImageData(x,y,width,height)`

返回的Image对象的数据属性返回一个对象，该对象包含指定的 ImageData 对象的图像数据

对于 ImageData 对象中的每个像素，都存在着四方面的信息，即 RGBA 值：

R - 红色 (0-255)

G - 绿色 (0-255)

B - 蓝色 (0-255)

A - alpha 通道 (0-255; 0 是透明的，255 是完全可见的)

把图像数据（从指定的 ImageData 对象）放回画布

- `context.putImageData(imgData,x,y,dirtyX,dirtyY,dirtyWidth,dirtyHeight);`

参数说明：

imgData 规定要放回画布的 ImageData 对象。

x 在画布上放置图像的位置。

y 在画布上放置图像的位置。

图像控制

- width 返回 ImageData 对象的宽度
- height 返回 ImageData 对象的高度
- data 返回一个对象，包含ImageData 对象的图像数据
- toDataURL() 获得图片base64加密的数据

图像控制
