note.md

# HTML5新特性 -- Unit03

# 1.CanvasRenderingContext2D接口

## 1.1 绘制文本

### • textAlign 属性

textAlign属性用于获取/设置文本的水平对齐方式，其语法结构是：

//设置
  
CanvasRenderingContext2D.textAlign = 'left|center|right'
  
   
//获取
  
variable = CanvasRenderingContext2D.textAlign

## 1.2 路径

路径（path）是将预先定义的坐标点顺序连接形成的图形。

路径在进行描边或填充之前在画布不可见

### • 路径的绘制步骤

A、通过beginPath()方法开始一条新路径

B、通过moveTo()方法定义路径起点

C、定义路径的内容(如rect()方法用于绘制矩形路径，arc()方法用于绘制圆弧等)

D、通过stroke()或fill()方法进行描边或填充

### • beginPath()方法

beginPath()方法用于清空之前的子路径，开始一个新的路径，语法结构是：

CanvasRenderingContext2D.beginPath()

### • moveTo()方法

moveTo()方法用于移动新路径的起点到指定的位置，其语法结构是：

CanvasRenderingContext2D.moveTo(x,y)

### • lineTo()方法

lineTo()方法实现使用直线连接路径终点，语法结构是：

CanvasRenderingContext2D.lineTo(x,y)

### • arc()方法

arc()方法用于绘制圆弧路径，其语法结构是：

CanvasRenderingContext2D.arc(x,y,半径,起始弧度,结束弧度)

圆弧的起点和终点用弧度表示

弧度的计算公式为： 角度 \* Math.PI / 180

如360度用 360 X Math.PI / 180

示例代码如下：

<canvas id="canvas"></canvas>
  
<script>
  
 let canvasEle = document.getElementById('canvas');
  
 let ctx = canvasEle.getContext('2d');
  
 canvasEle.width = 800;
  
 canvasEle.height = 480;
  
 ctx.beginPath();
  
 ctx.moveTo(400,240);
  
 ctx.arc(400,240,100,0, 2 \* Math.PI);
  
 ctx.fillStyle = '#f00';
  
 ctx.fill();
  
</script>

### • closePath()方法

closePath()方法用于返回当前路径的起点，语法结构是：

CanvasRenderingContext2D.closePath()

### • stroke()方法

stroke()方法用于根据当前的描边样式绘制当前路径，语法结构是：

CanvasRenderingContext2D.stroke()

### • fill()方法

fill()方法用于根据当前的填充样式绘制当前路径，语法结构是：

CanvasRenderingContext2D.fill()

### • clearRect()方法

clearRect()方法用于擦除画布指定区域的内容，语法结构是：

CanvasRenderingContext2D.clearRect(x,y,width,height)

动画示例代码如下：

<canvas id="canvas"></canvas>
  
<script>
  
 //颜色数组
  
 let colors = ['#589635', '#128469', '#753684', '#568430', '#865474', '#935475'];
  
 let canvasEle = document.getElementById('canvas');
  
 let ctx = canvasEle.getContext('2d');
  
 canvasEle.width = 800;
  
 canvasEle.height = 480;
  
 //X轴与Y轴的起始坐标
  
 let x = Math.floor(Math.random() \* canvasEle.width);
  
 let y = Math.floor(Math.random() \* canvasEle.height);
  
   
 if (x > canvasEle.width - 50) {
  
 x = canvasEle.width - 50;
  
 }
  
   
 if (y > canvasEle.height - 50) {
  
 y = canvasEle.height - 50;
  
 }
  
   
 //X轴移动的距离
  
 let xDistance = 1;
  
 //Y轴移动的距离
  
 let yDistance = 2;
  
   
 window.setInterval(() => {
  
 x += xDistance;
  
 y += yDistance;
  
 ctx.clearRect(0, 0, canvasEle.width, canvasEle.height);
  
 ctx.fillRect(x, y, 50, 50);
  
 //最右侧
  
 if (x > canvasEle.width - 50) {
  
 xDistance = -1;
  
 let color = colors[Math.floor(Math.random() \* colors.length)];
  
 ctx.fillStyle = color;
  
 }
  
 //最左侧
  
 if (x < 0) {
  
 xDistance = 1;
  
 let color = colors[Math.floor(Math.random() \* colors.length)];
  
 ctx.fillStyle = color;
  
 }
  
 //最底部
  
 if (y > canvasEle.height - 50) {
  
 yDistance = -2;
  
 let color = colors[Math.floor(Math.random() \* colors.length)];
  
 ctx.fillStyle = color;
  
 }
  
 //最顶部
  
 if (y < 0) {
  
 yDistance = 2;
  
 let color = colors[Math.floor(Math.random() \* colors.length)];
  
 ctx.fillStyle = color;
  
 }
  
 }, 15);
  
</script>

# 2.window对象

## • requestAnimationFrame()

requestAnimationFrame()方法用于定时循环操作，主要用于按帧对于网页进行重绘，其优势在于充分利用显示器的刷新频率，所以不会出现丢帧、卡顿等现象。而且一旦页面没有处于当前标签，则自动停止刷新，以节省CPU、GPU等资源。语法结构是：

//W3C建议
  
window.requestAnimationFrame(callback)
  
//Chrome、Opera、safari
  
window.webkitRequestAnimationFrame(callback)
  
//Firefox
  
window.mozRequestAnimationFrame(callback)
  
//Internet Explorer
  
window.msRequestAnimationFrame(callback)

返回值为整数，可以传入window.cancelAnimationFrame()方法中，用于取消回调函数的执行

## • cancelAnimationFrame()

cancelAnimationFrame()方法用于取消先前通过requestAnimationFrame()方法中生成ID，语法结构是：

//W3C建议
  
window.cancelAnimationFrame(callback)
  
//Chrome、Opera、safari
  
window.webkitCancelAnimationFrame(callback)
  
//Firefox
  
window.mozCancelAnimationFrame(callback)
  
//Internet Explorer
  
window.msCancelAnimationFrame(callback)