# Canvas

. 什么是 Canvas

就是 HTML 5 给出的一个可以展示绘图内容的标签.

最早是 苹果公司 提出的该标签.

threejs.org 里面有汽车等 3D 的仿真

## 基本使用

1> 提供 Canvas 标签即可. 默认就会占据 300 \* 150 的区域

2> 利用 html 属性为它设置宽高. 不要使用 CSS 来设置.

使用 属性设置 canvas 标签的宽高, 实际上相当于增加了 canvas 画布的 像素

但是如果使用 CSS 来设置画布的大小, 那么不会增加像素点, 只是将像 素扩大了

3> Canvas 只能展示绘图的内容. 但是不能进行绘图

利用 Canvas 找到绘图工具

每一个 Canvas 都有一套工具, 利用工具可以在当前 Canvas 上进行绘图

使用语法 canvas.getContext( '2d' ) 就返回一个在当前画布上绘图的工 具集

这个工具集专门绘制 平面图形. 如果要绘制 立体的图形需要传入 ' webgl'

绘制 平面图形的 对象是 CanvasRenderingContext2D 类型的

## 绘图的常用方法

ctx.moveTo( x, y ) 将绘图的起始点设置为 x,y

ctx.lineTo( x, y ) 从当前位置, 绘制直线到 x,y

ctx.stroke() 就是将刚刚绘制的所有的点联系起来. 就可以看到图形了.

绘制方法

ctx.fill()调用 fill 方法会将所有的点连接起来, 并构成一个封闭的图形结构

如果所有的描点没有构成封闭结构, 也会将开始的起点, 与最终的点

连接起来, 构成一个闭合的图形, 并在里面填充颜色(默认黑色)

闭合路径

closePath()

lineWidth 设置绘制图形的线宽

closePath 与 直接使用 lineTo 闭合是有区别的

虚线

ctx.setLineDash( 数组 )

ctx.getLineDash()

ctx.lineDashOffset = 值

数组中存储的数字就是分别表示 实线部分与空白部分的长度 [ 10 ]

如何设置描边与填充的颜色

ctx.strokeStyle 设置描边的颜色

ctx.fillStyle 设置填充的颜色

## 绘制形状

-> 矩形

ctx.rect( x, y, width, heigth ) 描边, 需要 stroke 或 fill

ctx.strokeRect( x, y, w, h )

ctx.fillRect( x, y, w, h )

ctx.clearRect( x, y, w, h ) 清除该矩形区域的内容

-> 清除整个画布

ctx.clearRect( 0, 0, cas.width, cas.height );

cas.width = cas.width;

-> 圆弧

ctx.arc( x, y, r, startAngle, endAngle, clockwise )

ctx.arcTo() 了解

3. 弧度制

为了更好的计算角度, 我们该角度提供一个新的定义, 用 PI 作为单位

将单位圆的一个整圈( 360 度 )记作 2 倍 的 PI.

这样的度量表示就是弧度制的表示方法.

60 度 PI / 3

45 度 PI / 4

30 度 PI / 6

学会进行转换

2 PI 刚好是一圈

一圈又是 360 度

2 PI 比上 360 度 = 弧度 比上 对应的角度

angle 角度

radian 弧度

function toAngle ( radian ) {

return radian \* 180 / Math.PI;

}

function toRadian ( angle ) {

return angle \* Math.PI / 180;

}

4. 角度的坐标

水平向右的角度是 0 度, 或 0 弧度

顺时针是正方向, 逆时针是负方向

练习: 绘制出, 圆心在 canva 正中心, 半径为 100, 角度从 -60度 到 120 度的圆弧

5. 如果没有当前位置, 绘制圆弧是没有任何问题

但是如果有了当前位置, 绘制圆弧的时候会将当前位置连接到圆弧上

6. 计算在圆弧上的点的坐标

7. 根据固定到起始点到 圆心, 结合圆弧和 closePath 方法可以绘制扇形( 楔形 Wedge)

8. 动态的通过动画, 一点一点的添加角度, 然后绘制一整个圆

## 绘制文字

ctx.fillText( 文本内容, x, y )

ctx.strokeText( 文本内容, x, y );

常用的属性

ctx.font = '30px 黑体'

ctx.textAlign left, center, right. start, end

ctx.textBaseline top, middle, bottom. hanging, ideographics, alphabetic

ctx.measureText( 文本 ) 获取当前文字的字体设置下, 文字的宽度对象

## 图片绘制

ctx.drawImage()

有三种调用形式

1> ctx.drawImage( img, x, y ) 将 image 绘制到 x, y 表示的位置

2> ctx.drawImage( img, x, y, width, height ) 将 img 绘制到一个矩形区域内

3> ctx.drawImage( img, sx, sy, sw, sh, x, y, w, h )

将图片部分内容绘制在该区域

将图片 img 的 sx, sy, （在图片上选中的位置）

sw, sh （选中图片区域的大小大小）

x, y, （在画布上的起始点）

w, h （在画布上的宽高）的矩形区域内.

2. 计时器模型

var id = setInterval(function () {

if ( 条件 ) {

clearInterval( id );

}

// 继续执行我的内容

}, 20);

3. 变换的概念

计算机绘图是利用坐标进行绘图. 绘制任何图形都和坐标系的结构息息相关.

所谓的变换就是一套数学公式, 可以记录坐标轴的变化方式.

利用坐标轴的变换可以绘制出, 根据不同坐标轴特点而形成的图形.

基本的 api

ctx.translate() 平移变换

ctx.rotate() 旋转变换

ctx.scale() 伸缩变换

4. 封装绘图对象

Line

Rect

Circle x, y, radius, strokeStyle, fillStyle, lineWidth

Arc

function Line( config ) {

}

Line.prototype = {

stroke: function () {

}

}

5. canvas 的状态

在 Canvas 中凡是设置了属性效果, 都会延续到后面一次修改

Canvas 在创建出来的时候, 是有一个默认的状态的

我希望每次修改状态的时候 都是不影响原来默认状态的

每次画完图时, 我都会新建一个状态, 然后绘制完成后

恢复到原有状态

ctx.save() 将当前状态保存

ctx.restore() 将保存的状态恢复

状态栈

6. 在 canvas 绘制的时候允许使用 canvas 绘制 canvas

ctx.darwImage( img, ... )

此时 img 可以是 图片, 还可以是 canvas, 甚至是 video

Konva 是一个完全面向对象的框架

将所有的东西都看做是对象: 图片, 直线, 矩形, ...

将整个canvas看做成舞台(stage)

在舞台上放一个层, 那么将所有的图形放在这个层中

命名空间

var num = 123;

function foo() {}

// 污染全局作用域

var Itcast = {};

Itcast.num = 123;

Itcast.foo = function () {};