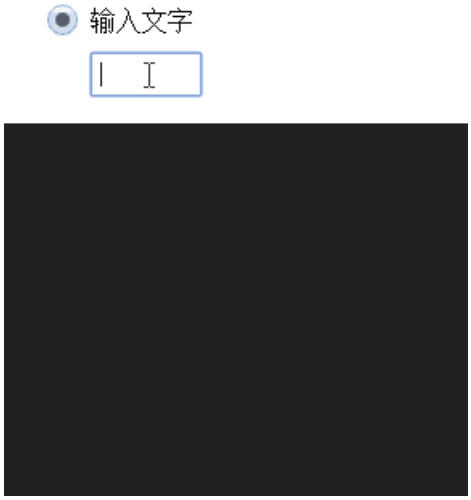
canvas getImageData与任意字符图形点、线动效实现

这篇文章发布于 2017年12月9日，星期六，22:20，归类于 Canvas相关。 阅读 19110 次, 今日 10 次 8 条评论

by zhangxinxu from http://www.zhangxinxu.com/wordpress/?p=6594
本文可全文转载，但需得到原作者书面许可，同时保留原作者和出处，摘要引流则随意。

一、动态字符或动态图像的连线动画效果

首先见下面gif效果截屏：



也就是我在输入框中输入一个字符`'5'`，然后回车，页面上就会以连线的方式把这个字符`'5'`给呈现出来。

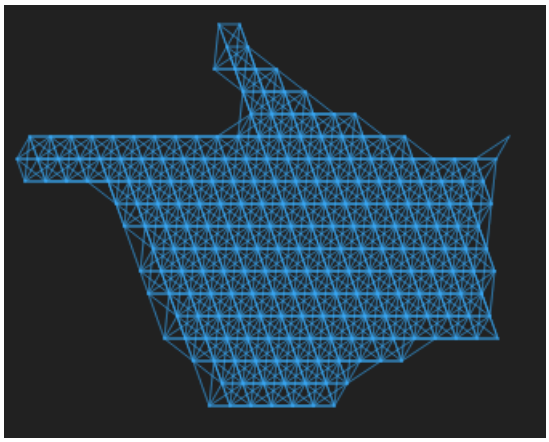
上面演示效果有对应的demo页面，您可以狠狠的点击这里：任意字符图形连线生成动效demo

不仅仅是字符，我们还可以选择本地图片进行图形生成。

例如有一个手指PNG图标如下图效果：

在demo页面中，我们选择这张图：

然后效果为：



下面简单讲下实现原理，以及拓展出来其它好玩东西。

二、canvas getImageData与动态字符图形信息获取

这种动态内容动效实现主要难点在于如果准确获得图形坐标。

实现原理如下：

1. 字符或图形绘制在canvas上；
2. 使用getImageData方法获取canvas所有像素点信息；
3. 以固定间隙遍历这些像素点，非透明像素点位置作为动效关键点坐标；
4. 拿到这个动效关键点坐标，做连线动画效果，或者实现其它效果。

以上面的demo为例，我们看下上面4步原理是如何走下来的。

1. 绘制

如果是字符，使用 `fillText()` 方法把字符绘制在canvas画布上；如果是图片，使用 `drawImage()` 方法把图片绘制在canvas画布上。

`fillText()` 语法如下：

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

其中， `text` 表示需要绘制的文本内容， `x, y` 为文本绘制的坐标位置， `maxWidth` 是可选参数，我是这段文本所占据的最大宽度。

`canvas` 中的文本绘制和CSS/HTML中的文本呈现有不少区别，其中最典型的区别之一就是它不会自动换行，这里的 `maxWidth` 并不是表示文本超过这个宽度会自动换行，而是文本会自动收缩变窄，就像被门挤了一样，保证一行显示同时宽度不超过这个限制（下图示意）。

`drawImage()` 语法如下：

```
context.drawImage(img, dx, dy);
context.drawImage(img, dx, dy, dWidth, dHeight);
context.drawImage(img, sx, sy, sWidth, sHeight, dx, dy, dWidth, dHeight);
```

具体解释（包含如何本地图片绘制到canvas上）可以参见这篇文章“canvas实现图片前端JS压缩并上传”第三段，这里不赘述。

2. 像素点信息获取

这里主要介绍下`getImageData()`方法，语法如下：

```
context.getImageData(sx, sy, sw, sh);
```

其中，`sx, sy`表示你想获取的像素信息区域的左上角坐标，`sw, sh`表示你想获取的像素信息区域的宽度和高度。

例如一个canvas的尺寸是320*320，则`context.getImageData(10, 10, 100, 100)`，就是表示获取canvas左上角`10px, 10px`宽度`100px`高度`100px`的矩形区域的像素点信息。

`getImageData()`方法返回的是一个`ImageData`对象，包括`width`，`height`以及`data`三个属性。通常我们对像素点进行处理，需要的是后面的`data`属性。

例如：

```
var imageData = context.getImageData(0, 0, canvas.width, canvas.height);
```

则`imageData.data`就是我们需要的像素点信息数组，是个一维数组，类型为`Uint8ClampedArray`，Uint表示自然数，`8`可以理解为数值范围不超过`2`的`8`次方，也就是不超过`256`。所以这个数组中所有的值范围都是`0~255`。

`imageData.data`这个数组表示的是像素点的RGBA颜色信息，由于类型为一维数组，因此，RGBA颜色信息被按顺序作为4个独立的数组项依次排列的。

加上一张图片，只有两个像素点，分别是纯黑色和纯白色，使用CSS3的RGBA颜色表示就是rgba(0,0,0,1)以及rgba(255,255,255,1)，其中RGB颜色和canvas中的ImageData范围一致，是`0~255`，但这个与透明度相关的A需要转换下，`0->0`，`1->255`。

所以，如果我们使用`getImageData()`获取这两个像素点信息，则`imageData.data`数组如下所示：

```
[0,0,0,255,255,255,255,255]
```

共8项，每4项表示一个像素点信息。

3. 关键坐标获取

算法如下，水平和垂直方向均以固定间隙去读取`imageData`像素点信息，如果是完全不透明的像素点，则作为我们需要的关键坐标保存下来。

相关JS代码如下：

```
var imgData = context.getImageData(0,0,width,height).data;
// 间隙大小为13
var gap = 13;
var pos = [];
var x = 0, y = 0, index = 0;
for (var i=0; i<imgData.length; i+=(4*gap)) {
    if (imgData[i+3] == 255) {
        // 塞入此时的坐标
        pos.push({
            x: x,
            y: y
        });
```

```
    }
    index = Math.floor(i / 4);
    x = index % width;
    y = Math.floor(index / width);
    if (x >= width - gap) {
        i += gap * 4 * width;
    }
}

// pos就是我们需要的关键点坐标集合
```

这里的间隙大小选取还是值得一说的，也就是这里的`gap`变量值应该选多大，根据研究和测试发现，我们的间隙选取质数时候的呈现效果要好一些，`13`就是一个非常棒的质数。

为什么要选择质数呢？因为质数是不能因式分解的自然数，随着不断循环，其“自然随机性”要比非质数高，这种以质数作为循环周期来增加“自然随机性”的策略有一种专门的称谓，叫做“蝉原则”。我之前有专门写文章介绍过，有兴趣可以阅读下：“蝉原则”与CSS3随机多背景随机圆角等效果。

4. 坐标与动效呈现

有了固定间隔的关键点坐标，下面就是动效呈现了，此时，可供我们发挥的空间就很大了。

例如，我们可以借助SVG的`stroke`描边动画来轻松实现我们的连线动画效果。

三、SVG描边动画与连线动效

关于SVG路径描边动画技术可以参见我之前写的“纯CSS实现帅气的SVG路径描边动画效果”一文。

所以，我们需要点CSS配合：

```
line {
    stroke: rgba(59,166,241,.8);
    stroke-width: .5px;
    transition: stroke-dashoffset 2s ease-in-out;
}
```

然后，根据坐标，我们创建SVG直线就好了，使用`<line>`元素，原理如下（以一根线示意）：

```
var line = document.createElementNS('http://www.w3.org/2000/svg', "line");
// 设置起止坐标
line.setAttribute('x1', x1);
line.setAttribute('y1', y1);
line.setAttribute('x2', x2);
line.setAttribute('y2', y2);
// 设置直线偏移到看不见
line.style.strokeDasharray = distance + " " + distance;
line.style.strokeDashoffset = distance;

svg.appendChild(line);

// 然后设置线可以看见，动画交给CSS3
window.requestAnimationFrame(function() {
    lines.style.strokeDashoffset = 0;
})
```

以上就是SVG实现连线动画的原理，就这样结束了？

稍等，还差一步，那就是不是所有的点和点都可以作为线连接，例如，字符`'5'`的左上角点和右下角点就不能连在一起，否则效果会乱套的。所以我们需要对一些坐标进行过滤，有些可以连线，有些不行，判断的标准是两个点之间的距离。如果两个点之间的距离超过一定的限制，我们就认为它并不是构成图形所必须的连线，于是整条`line`就舍弃掉。

例如本demo中，有设置：

```
var maxDistance = 31, minDistance = 0;
```

也就是两个点之间的距离，如果超过`31`像素，则认为这两个点不适合连线。

具体JS代码可以参考demo页面源代码。

虽然使用SVG实现任意字符图形的连线动画相对简单些，但是在实际应用的时候，发现有一个致命的问题，那就是性能问题，如果我们的连线数目超过两百，甚至更多，则在移动端，尤其一些Android设备，能明显感觉到卡顿。

所以，实际应用的时候，还是建议使用`canvas`技术来绘制我们的连线。

四、canvas绘制与动态内容的连线动效

这里介绍一下如何使用canvas实现我们的连线动效，前3步原理都是一样的，不一样的是线的绘制。

canvas绘制线的相关JS如下：

```
context.beginPath();
context.moveTo(x1, y1);
context.lineTo(x2, y2);
context.lineWidth = 0.5;
context.strokeStyle = 'rgba(59,166,241,0.5)';
context.stroke();
```

然后套用canvas 2D动效实现的常用套路即可！参见这里：“canvas 2D炫酷动效的实现套路”。

如果你效果急用，没时间学习，没关系，有现成的demo页面可供参考，您可以狠狠的点击这里：canvas实现任意图片和文字的连线动画demo

例如，我输入一个爱心字符：

性能非常流畅。

下图是该demo地址的二维码，有兴趣可以手机扫个码，感受下性能。

五、扩展：基于关键点坐标的canvas点动效

上面介绍和展示的都是基于关键点坐标实现的线动画，实际上有了坐标，我们可以做的事情非常多，例如我们可以实现一些“点”动画效果。

百闻不如一见，可以围观下面的效果实现（GIF截屏）：

上面的“线动画”每次都是重新绘制的，而这里的“点动画”前后图形变换是联动的，字符变化可以联动，字符变图像也是可以联动。

您可以狠狠地点击这里：canvas实现的任意图片和文字点变换动效demo

注释良好，未压缩的JS代码就在页面上。

例如，这里的点动画效果，我们稍微改改就可以实现一个超酷的倒计时动画效果！

具体实现也是套用canvas 2D动效实现的常用套路，可参见“canvas 2D炫酷动效的实现套路”一文。

六、使用许可申明和结语

本文所有JS均采用MIT许可，也就是你**可以编辑和再加工甚至商用，但是，务必保留作者和描述这两行注释信息**。

除了尊重原作者外，日后接手维护你代码的其他同事遇到不明白之处也知道去哪里寻根溯源，答疑解惑。

其它

我们其实还可以进一步拓展，本文是选区的是非透明点作为动效关键点。实际上，我们还可以基于灰度、基于色彩确定我们的动效关键点。

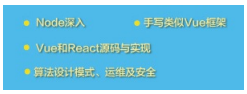
或者配合一些算法，例如查找边缘，找到人物过物体的边缘坐标，然后实现一些酷酷的效果等。

以上~

感谢阅读，欢迎交流！

> 《CSS世界》签名版独家发售，包邮，可指定寄语，点击显示购买码

（本篇完） // 想要打赏？点击这里。有话要说？点击这里。



« 照片位图转SVG矢量图片JS工具primitive.js等简介

用3D LUT滤镜我做了个在线专业电影级别照片调色工具 »

猜你喜欢

- 纯CSS实现帅气的SVG路径描边动画效果
- “蝉原则”与CSS3随机多背景随机圆角等效果
- 纯前端实现可传图可字幕台词定制的GIF表情生成器
- 小tips: 使用JS检测用户是否安装某font-family字体
- 我对CSS vertical-align的一些理解与认识（一）
- 不借助Echarts等图形框架原生JS快速实现折线图效果
- 团购类网站倒计时的js实现
- CSS3+js实现多彩炫酷旋转圆环时钟效果
- 3种纯CSS实现中间镂空的12色彩虹渐变圆环方法
- canvas文本绘制自动换行、字间距、竖排等实现
- 小tips: 0学习成本实现HTML元素粘滞融合效果

分享到：

1

标签： canvas, drawImage, fillText, getImageData, line, stroke-dashoffset, SVG, 倒计时, 蝉原则

发表评论（目前8条评论）

名称 (必须)

邮件地址(不会被公开) (必须)

网站

提交评论

1. **Jscss说道：**
   2018年12月26日 10:16

   i += gap * 4 * width;这是啥意思？i不是数组下标嘛， 万一width是1000， i就瞬间加了4w?

   回复

2. **craazy说道：**
   2018年01月16日 14:13

   多个字的需要怎么处理呢

   回复

3. **6K说道：**
2018年01月4日 19:46

非常强

回复

4. **zq说道：**
2017年12月13日 20:17

厉害厉害！

回复

5. **libing_cheer说道：**
2017年12月11日 11:30

很棒哈哈。
如果输入比较复杂的字，比如龘，显示会模糊的。

回复

6. **方正说道：**
2017年12月11日 03:03

太牛了！！

回复

7. **DeepKolos说道：**
2017年12月9日 23:09

膜拜大神~~

回复

8. **风海流说道：**
2017年12月9日 22:45

性能真的不错，手机上也非常流畅！

回复

最新文章

» 常见的CSS图形绘制合集
» 粉丝群第1期CSS小测点评与答疑
» 分享三个纯CSS实现26个英文字母的案例
» 小tips: 纯CSS实现打字动画效果
» CSS/CSS3 box-decoration-break属性简介
» CSS :placeholder-shown伪类实现Material Design占位符交互效果
» 从天猫某活动视频不必要的3次请求说起
» CSS vector-effect与SVG stroke描边缩放
» CSS ::backdrop伪元素是干嘛用的?
» 周知：CSS -webkit-伪元素选择器不再导致整行无效

今日热门

» 常见的CSS图形绘制合集 (190)
» 未来必热：SVG Sprite技术介绍 (119)
» 粉丝群第1期CSS小测点评与答疑 (115)
» HTML5终极备忘大全（图片版+文字版） (93)
» 让所有浏览器支持HTML5 video视频标签 (86)
» Selectivizr-让IE6~8支持CSS3伪类和属性选择器 (82)

» CSS3下的147个颜色名称及对应颜色值 (79)
» 视区相关单位vw, vh..简介以及可实际应用场景 (76)
» 写给自己看的display: flex布局教程 (76)
» 小tips: 纯CSS实现打字动画效果 (76)

今年热议

» 《CSS世界》女主角诚寻靠谱一起奋斗之人 (76)
» 不借助Echarts等图形框架原生JS快速实现折线图效果 (64)
» 看，for..in和for..of在那里吵架！ (60)
» 是时候好好安利下LuLu UI框架了！ (47)
» 原来浏览器原生支持JS Base64编码解码 (35)
» 妙法攻略：渐变虚框及边框滚动动画的纯CSS实现 (33)
» 炫酷H5中序列图片视频化播放的高性能实现 (31)
» CSS scroll-behavior和JS scrollIntoView让页面滚动平滑 (30)
» windows系统下批量删除OS X系统.DS_Store文件 (26)
» 写给自己看的display: flex布局教程 (26)

猜你喜欢

- 纯CSS实现帅气的SVG路径描边动画效果
- “蝉原则”与CSS3随机多背景随机圆角等效果
- 纯前端实现可传图可字幕台词定制的GIF表情生成器
- 小tips: 使用JS检测用户是否安装某font-family字体
- 我对CSS vertical-align的一些理解与认识（一）
- 不借助Echarts等图形框架原生JS快速实现折线图效果
- 团购类网站倒计时的js实现
- CSS3+js实现多彩炫酷旋转圆环时钟效果
- 3种纯CSS实现中间镂空的12色彩虹渐变圆环方法
- canvas文本绘制自动换行、字间距、竖排等实现
- 小tips: 0学习成本实现HTML元素粘滞融合效果

Designed & Powerd by zhangxinxu
Copyright© 2009-2019 张鑫旭-鑫空间-鑫生活
鄂ICP备09015569号