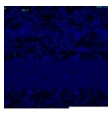
瞎折腾：把JS,CSS任意文本文件加密成一张图片

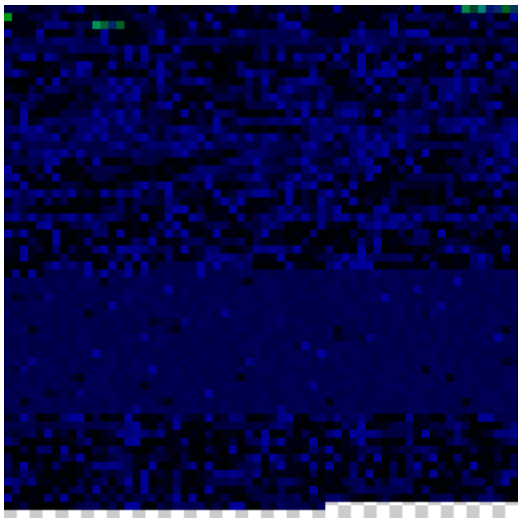
这篇文章发布于 2017年12月24日，星期日，16:04，归类于 JS实例。 阅读 22317 次, 今日 11 次 28 条评论

一、这是一张包含特殊信息的加密图片

64像素*64像素原始加密图：



放大5倍呈现后：



这张64*64见方的图片实际上包含了一段4K大小的JavaScript脚本的信息。

相信如果不知道这张图片的加密算法，就算是爱因斯坦也很难短时间破解之。

二、JavaScript代码的图片加密与解密

加解密的原理如下：

- 加密：获取文本字符的Unicode编码，范围是 0 – 65535 之间的整数。然后把这个整数转化成RGB色值，每一个字符对应一个像素值颜色，然后绘制成图片。
- 解密：读取图片的像素信息，转化成对应的Unicode编码，再转换成对应字符。

加解密的媒介canvas

加密的图片生成，解密的图片信息读取，都是借助 `canvas` 实现的。

`canvas` API中的 `putImageData` 可以用来生成图片； `canvas` API中的 `getImageData` 可以用来读取图片中的像素信息。

对于通过web生成的数据，我们处理相对简单一些，因为都在同一个web平台上。

例如我们发表了一篇需要付费阅读的文章，可以这么处理：

1. 将文章文本转换成RGB颜色信息绘制在canvas画布上；
2. 通过 `canvas` 的 `toDataURL()` 方法，将画布信息转为 `base64` 格式，发送给后端，后端以图片格式进行信息接收（或直接base64地址入库——大小都差不多）；
3. 当需要呈现文章的时候，通过读取这张图片信息进行解密，得到文章内容。然后再以画布的形式显示文章内容，这样页面源代码和实时DOM都没有文章文字信息。可以大大提高盗版的门槛。

如果是本地开发的源代码想要加密呢？

则需要借助本地工具对这些文件进行图片格式转换，对于前端开发同学而言，可以自己写一个node.js小工具。

下面这段node.js代码就是我自己写自己用的，大家有兴趣可以作为参考：

```
const { createCanvas, loadImage } = require('canvas');
const fs = require('fs');
const file2Img = function (filename) {
    fs.readFile(filename, {
        encoding: 'utf8'
    }, function (err, data) {
        let length = data.length;
        let size = Math.ceil(Math.sqrt(length));
        // 绘制canvas
        const canvas = createCanvas(size, size);
        const ctx = canvas.getContext('2d');
        // 获取透明像素数据
        var imgData = ctx.getImageData(0, 0, size, size);
        // 透明像素数据替换为实色数据
        var index = 0;
        for (var start=0; start<size*size; start++) {
            let charCode = data.charCodeAt(start);
            if (Number.isNaN(charCode) == false) {
                let hex = (charCode + '').padStart(6, '0');
                for (var i=0; i<6; i+=2) {
                    imgData.data[index++] = parseInt('0x' + hex.slice(i, i + 2));
                }
                // 透明度
                imgData.data[index++] = 255;
            }
        }
        // 使用新颜色信息绘制
        ctx.putImageData(imgData, 0, 0);

        // 保存的PNG文件名
        var imgFilename = filename.split('.')[0] + '.png';
        let stream = canvas.pngStream();
        // 输出PNG流
        let out = fs.createWriteStream(__dirname + '/' + imgFilename);
        stream.on('data', function(chunk) {
          out.write(chunk);
        });
        stream.on('end', function(){
          console.log('The '+ imgFilename +' stream ended');
        });
    });
};
```

使用示意，命名一个file2img.js文件，复制并粘贴上面JS代码，代码最后加上下面一句（js文件名换成你希望转换的）：

```
file2Img('colorful-min.js');
```

然后执行：

```
node file2img
```

此时，就会把colorful-min.js变成colorful-min.png图片了。

本地实现麻烦之处
看代码量，似乎觉得本地文件转图片也是洒洒水的程度。

事非经过不知难，本地文件转图片，麻烦的其实并不是代码实现本身，而是`node-canvas`模块的安装（node-canvas项目地址见这里），尤其在windows系统下的安装，我看网上还有人折腾了一整天才弄好的。

我自己折腾了好几个小时，完全按照官方的步骤，一个个的安装，甚至为了安装node-gyp，还下载了占用了3.29G的Microsoft Visual C++ Build Tools，结果还是狗屎。

什么binding.gyp not found，c2373 __pfnDliNotifyHook2之类错误。

我跟大家讲，官方的安装教程，很不靠谱，漏了一个非常重要的东西。

第1步并不是安装node-gyp，而是升级Node.js到最新版。

我折腾这么久，就是因为Node.js还是老版本的原因，  。

然后，我使用的是2.0的用法，默认安装还是1.x版本，因此，canvas包安装时候，要走下面：

```
npm install canvas@next
```

三、解密图片并执行实际案例

下面这段JS可以对图片进行解密并直接执行其对应的JavaScript脚本：

```
var jsParseImg=function(l,g){var e=document.createElement("img"),f=document.createElement(
"canvas");e.onload=function(){f.width=this.width;f.height=this.height;var a=f.getContext("
2d");a.drawImage(this,0,0,this.width,this.height);for(var a=a.getImageData(0,0,this.width,
this.height),e=a.data.length,h=[],b=0;b<e;b+=4){var k=a.data[b].toString(16),c=a.data[b+1]
.toString(16),d=a.data[b+2].toString(16);a.data[b+2].toString(16);1==c.length&&(c="0"+c);1
==d.length&&(d="0"+d);0!=Number(k+c+d)&&h.push(String.fromCharCode(Number(k+
c+d)))}window.eval(h.join(""));g&&g()};e.src=l};
```

语法如下：

```
jsParseImg(imgurl, callback);
```

其中，`imgurl`只图片的URL地址，可以是普通协议地址，也可以是base64地址，甚至是Blob地址；`ca`

`llback` 为解密并解析成功后的回调。

眼见为实，您可以狠狠地点击这里：JS解析解密图片并执行demo

在原demo页面中，炫彩背景的 `colorfulBackground()` 方法是在 `colorful-min.js` 中声明的；但是，在这里，有的仅仅是一个 `colorful-min.png` 图片请求。

但，这并不妨碍我们执行：

```
jsParseImg('colorful-min.png', function () {
    colorfulBackground({
      container: document.getElementById('container'),
      animation: false,
      size: [400, 800]
    });
});
```

来实现我们想要的效果。

想必那些喜欢扒代码的伸手党，看了这个页面之后，一定是一脸懵逼。

四、结束语

有小伙伴可能会担心，加密和解密的东西呀，最宝贵的就是算法了，你这里基本上原封不动就暴露了，那基本上就没啥用了呀。

这个其实大可不必担心。

本文所展示的颜色处理只是抛砖引玉，属于最基本最简单的，从头往后线性的颜色绘制。

如果在实际项目中应用，肯定就不会想这么简单的策略了。

颜色信息的起点可以从中间，圆环的形式；或者逐行绘制的方式。

或者更近一步，每一个像素点藏更多的信息。因为字符最大Unicode编码是65535，但是一个像素点RGB

A所能包含的数值范围信息是256^4，也就是4294967296，完全不是一个数量级的。如果采用像素通道组合方式，一个像素点藏2个字符信息，是完全可行的。一个像素通道范围 `0~255`，其中两个组合，则范围大小256*256为65536，排除掉认为是无效像素信息的 `0,0` 组合正好涵盖65535。

恩，有时间可以再试试。

这样，图片的尺寸可以进一步缩小。

本文展示的JS和生成突破尺寸对比为：3.94K VS 5.33K。静态资源成本是有所增加的，如何权衡，还要看大家自己。

在前端领域，是没有百分之百的加密的，只要花足够多的时间和耐心，总会窥探到到门路的。而且对前端技术这种东西，都是开放的，尤其web这种项目，你说你JS代码镶了金镶了钻，非加密不可，想必多半会一笑而过吧。所以，平常的压缩混淆已经足够了。

所以标题才有“瞎折腾”字样。自己玩一玩试验试验，实际上实际开发的估计很少使用。但是用的少，并不一定没有价值，说不定遇到特殊项目特殊场景，就能高光一把。

感谢阅读！

> 《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。

« 小tips: 0学习成本实现HTML元素粘滞融合效果　　借助HTML5 details,summary无JS实现各种交互效果 »

猜你喜欢

- JS检测PNG图片是否有透明背景、抠图等相关处理
- 小tips:使用canvas在前端实现图片水印合成
- SVG <foreignObject>简介与截图等应用
- 小tip: 使用CSS将图片转换成模糊(毛玻璃)效果
- CSS3 box-shadow盒阴影图形生成技术
- 图片主色获取脚本rgbaster.js小介绍小使用
- canvas getImageData与任意字符图形点、线动效实现
- 解决canvas图片getImageData,toDataURL跨域问题
- 小tips: 使用JS检测用户是否安装某font-family字体
- 高富帅seajs使用示例及spm合并压缩工具露脸
- canvas图形绘制之星空、噪点与烟雾效果

标签： canvas, getImageData, putImageData, rgba, toDataURL, 加密, 压缩, 解密

发表评论（目前28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. 多疑说道：
2018年06月9日 23:25

多次提到“伸手党”，请问您对“伸手党”关键词是怎么定义的？

回复

2. 静静的说道：
2018年05月28日 15:48

无论加密的再好 运行的时候 eval前加一个 console.log(h.join(“”)) 就把代码读出来了啊…… 解png的这部分代码有办法加密吗？

回复

3. 阳阳说道：
2018年04月16日 15:16

看到大神的文章，资料分享，感觉到国内程序员还是真心有专注于技术的传播与分享的存在的。

回复

4. 路人说道：
2018年03月7日 14:51

网页排版可以整体居中，用户体验会好点

回复

5. 猪猪侠邓欣宜说道：
2018年03月6日 17:11

很好

回复

6. 梅世祺说道：
2018年02月17日 03:30

大佬解释的好清楚

回复

7. **reyona说道：**
2018年01月31日 08:32

类似炒鸡难装的还有node.js环境下操作oracledb的模块……曾经装了一个星期才装上去……

回复

> **ugrg说道：**
> 2018年02月11日 12:22
>
> 哈哈,所有nodejs下难装的模块应该都node-gyp这个东西有关,不过可以试试直接按装
> npm install –global –production windows-build-tools
> 这东西按装速度很慢,不过装好后一般都可以直接能用了.
>
> 回复
>
> > **jwei说道：**
> > 2018年04月3日 10:16
> >
> > 厉害厉害
> >
> > 回复

8. **nicholasurey说道：**
2018年01月22日 11:34

感觉和二维码的思路有点像啊

回复

9. **刘瑶168设计切图工作室说道：**
2018年01月17日 22:45

大神，牛逼啦！

回复

10. **刘昭廷说道：**
2018年01月11日 14:59

我买啦鑫神的书《CSS世界》啦，我是个很少买书看得人，因为电子版到处都有下载，但是我一看是鑫神的，二话不说就下单了！支持鑫神著作，希望有更多著作出现！

回复

11. **来语直搜说道：**
2018年01月9日 00:12

敬佩多年潜心研究一样东西的人！

回复

12. **路人甲说道：**
2018年01月8日 17:16

很久之前就看您的博客，这次花了一个月从最09年一直到17年12月的博客，估计消化的都不到10%，但是确实吸收了很多奇淫的技巧和思路，非常感谢您对我帮助，以后有时间会再来重新过一遍的。以前没留过言，现在真的想说一声感谢….

回复

13. **素材火说道：**
2018年01月8日 10:24

这个牛啊

回复

14. **lg说道：**
2018年01月6日 14:57

很有收获

回复

15. **一点点说道：**
2018年01月4日 11:22

新书有电子版的吗

回复

16. **上官说道：**
2018年01月2日 09:56

买新书赠个PDF版撒，车上也看看

回复

17. **IT技术社区说道：**
2017年12月28日 23:11

不错，来支持支持！

回复

18. **大少说道：**
2017年12月27日 21:37

张哥，你好，经常看你的博客文章，写得特别深入，详细，谢谢了。不过看你的博客时发现一个问题就是代码块的换行有问题，我这段时间刚好做了一个markdown转换工具Md2All,支持markdown语法和html,支持“一键排版”的css模板选择和自定义css,还有80多种代码风格的highlight,能生成自定义的html文件用于博客中，你有空可以试试，有意见随时提，希望能帮上点忙.网址：http://md.aclickall.com

回复

19. **wingmeng说道：**
2017年12月26日 09:59

恭喜旭哥出书！已买，不解释

回复

20. **horan说道：**
2017年12月25日 17:12

呃，关于坑爹的 node-gyp，可以考虑这个：
npm install –global –production windows-build-tools

回复

21. **猫咪君-VRlie说道：**
2017年12月25日 10:39

旭哥，祝你新书大卖。看了目录之后我觉得我也要买一本O(∩_∩)O

回复

22. 孙老四说道：
2017年12月25日 08:13

祝书大卖！

回复

23. 依韵宵音说道：
2017年12月25日 06:13

书上架了这么没见发微博或文章宣传下呢。
已购，是你让我感受到了css还能这么玩。

回复

24. 个人博客说道：
2017年12月24日 17:22

感谢分享、这个起到压缩效果吗?

回复

张 鑫旭说道：
2017年12月24日 21:43

不能。

回复

25. 怡红公子说道：
2017年12月24日 16:17

屈屈老师之前也写过类似的文章 https://imququ.com/post/code2png-encoder.html

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 (179)
» 粉丝群第1期CSS小测点评与答疑 (113)
» 未来必热：SVG Sprite技术介绍 (111)
» HTML5终极备忘大全（图片版+文字版） (85)
» 让所有浏览器支持HTML5 video视频标签 (83)
» Selectivizr-让IE6~8支持CSS3伪类和属性选择器 (80)
» CSS3下的147个颜色名称及对应颜色值 (78)
» 小tips: 纯CSS实现打字动画效果 (73)
» 写给自己看的display: flex布局教程 (70)
» 分享三个纯CSS实现26个英文字母的案例 (70)

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

- JS检测PNG图片是否有透明背景、抠图等相关处理
- 小tips:使用canvas在前端实现图片水印合成
- SVG <foreignObject>简介与截图等应用
- 小tip: 使用CSS将图片转换成模糊(毛玻璃)效果
- CSS3 box-shadow盒阴影图形生成技术
- 图片主色获取脚本rgbaster.js小介绍小使用
- canvas getImageData与任意字符图形点、线动效实现
- 解决canvas图片getImageData,toDataURL跨域问题
- 小tips: 使用JS检测用户是否安装某font-family字体
- 高富帅seajs使用示例及spm合并压缩工具露脸
- canvas图形绘制之星空、噪点与烟雾效果