

02

OPEN ORIENTED

凹凸实验室

在CSS中使用内联SVG的探索

网页嵌入SVG的六种方式

- ``
- `<iframe src= 'mysvg.svg' ></iframe>`
- `<embed src= 'mysvg.svg' ></embed>`
- `<object type="image/svg+xml" data= "mySVG.svg" ></object>`
- HTML内联`<svg>`
- css的资源引用：`background-image: url(mysvg.svg)`

```
<div class="seckill_banner">
  <div class="seckill_banner_title">
    <svg viewBox="0,0,332,48">
      <path id="textPath" d="
m0.749997,39.860256c92.99817,20.99958
250.99507,-15.99969 327.99356,1.99996'
      opacity="0.5" stroke-opacity="null"
      stroke-width="1" stroke="none" fill="
      none"/>
      <text fill="#fff" font-size="28" x="0"
      y="0">
        <textPath xlink:href="#textPath"
        text-anchor="middle" startOffset="
        50%">
          利益点利益点利益点
        </textPath>
      </text>
    </svg>
  </div>
</div>
```

优点

- 样式和脚本不受SVG文件限制，可以放在当前文档任意位置
- 可以当前文档直接修改
- 移植方便
- 没有跨域问题

缺点

- 造成HTML臃肿
- 造成HTML臃肿
- 造成HTML臃肿

CSS中使用内联SVG可以解决HTML节点臃肿

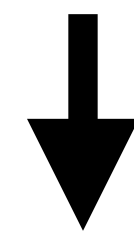
以下是我设想的代码：

```
background: url(  
  <svg xmlns="http://www.w3.org/2000/svg" width="393.969" height="28.219"  
  viewBox="0 0 393.969 28.219">  
    <path d="M0.969,11.313 C-20.207,69.711 314.183,-47.984 393.969,24.312 "/>  
  </svg>  
)
```

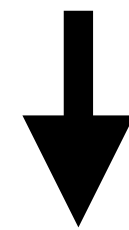
事实上，没有任何浏览器支持以上代码

CSS引用SVG的正确方式：

```
background: url(  
  <svg xmlns="http://www.w3.org/2000/svg" width="393.969" height="28.219"  
  viewBox="0 0 393.969 28.219">  
    <path d="M0.969,11.313 C-20.207,69.711 314.183,-47.984 393.969,24.312 "/>  
  </svg>  
)
```



demo.svg



background: url(demo.svg)

```
background: url(data:image/  
svg+xml;base64,PHN2ZyB4bWxucz0iaHR0cDovL3d3dy53My5vcmcvMjA5Lzdn  
ciIHZpZXdCb3g9IjAgMCA3NTAgNDAwIj4gCQkJPHBhdGggZD0iTTAsMCBMNzUwL  
DAgTDc1MCw0MDAgTDM4OCw0MDAgTDM3NSwzODcsIEwzNjIsNDAwIEwwLDQyL  
MFoiIGZpbGw9IiMwZjAiPjwvcGF0aD4gCQk8L3N2Zz4=);
```

Data URI scheme

Data URI 模式在1995 年被首次提议。规范对它的描述为：“允许将小块数据内联为‘立即数’，数据就在其URL 自身之中”。

语法如下：

```
data: [<media type>] [;base64], <data>
```


Data URI scheme

```
data: [<media type>] [;base64], <data>
```

data: — URI的协议头

media type — 表示数据呈现的格式，即指定嵌入数据的MIME。默认为 text/plain

;base64 — base64编码扩展，为可选项

<data> — 编码后的文件正体。如果不使用base64扩展，那正体将会使用URL 编码，即我们熟悉的“%xx”形式

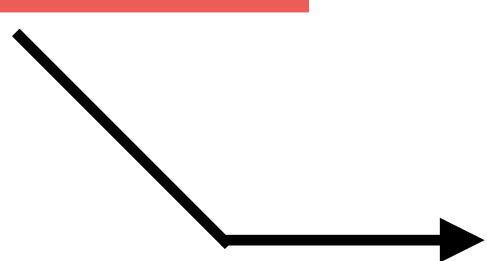
<data>即是编码后的文本内容

在SCSS中实现一个编码函数就可以实现 svg 内联了。

使用SASS来实现文本编码 — URL编码

URL编码又叫百分号编码，它的编码很简单：

$$\% \quad + \quad \underline{2xHex} \quad = \quad \text{ASCII字符}$$

 ascii对应的十六进制数值

使用SASS来实现文本编码 — URL编码

SASS的内置函数大全：

<http://sass-lang.com/documentation/Sass/Script/Functions.html>

主要函数是以下两个：

`str_index(string, substring)`

`str_slice(string, start_at, end_at = nil)`



使用SASS来实现文本编码 — URL编码

<https://github.com/leeenx/escapeSvg>





%XX too low too simple?

使用SASS来实现文本编码 — base64

Base64字符即US-ASCII子集的64个字符，如下：

```
'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H',  
'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',  
'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X',  
'Y', 'Z', 'a', 'b', 'c', 'd', 'e', 'f',  
'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n',  
'o', 'p', 'q', 'r', 's', 't', 'u', 'v',  
'w', 'x', 'y', 'z', '0', '1', '2', '3',  
'4', '5', '6', '7', '8', '9', '+', '/'
```

简单地说，使用4个base64字符表示3个ascii字符

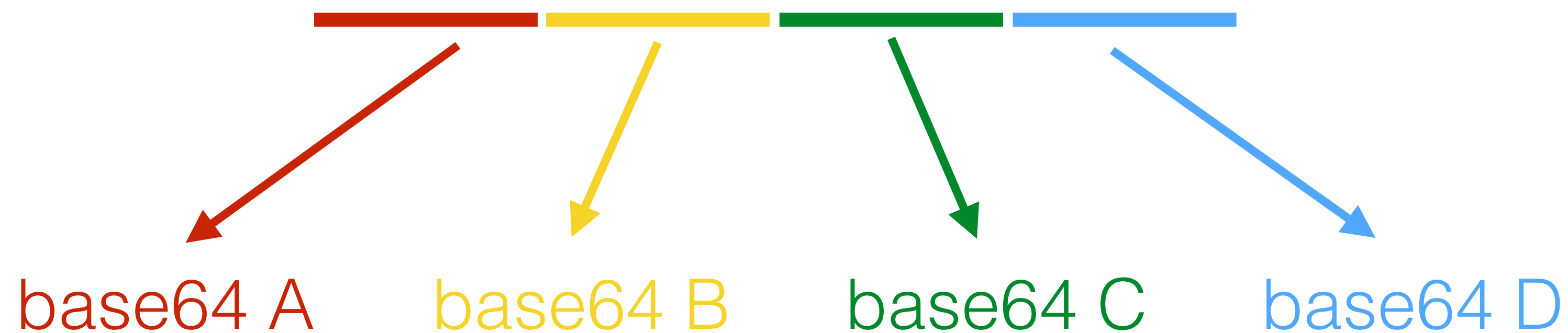
使用SASS来实现文本编码 — base64

base64的编码原理：

ASCII 1
hhhhhhhh

ASCII 2
eeeeeeee

ASCII 3
xxxxxxxx



使用SASS来实现文本编码 — base64

对应的base64二进制串不足6位时，使用0来补全
base64不足四个一组时，使用使用字符 = 来占位。



“me” base64编码的过程如下：

me -> 0110110101100101 -> 011011 + 010110 + 010100 -> bWU =

使用SASS来实现文本编码 — base64

SASS没有获取ASCII码的函数，更没有转base64的函数。为了实现 base64 编码，需要手工建立这两个对照map。

```
$base64Codes: (  
  'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H',  
  'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',  
  'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X',  
  'Y', 'Z', 'a', 'b', 'c', 'd', 'e', 'f',  
  'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n',  
  'o', 'p', 'q', 'r', 's', 't', 'u', 'v',  
  'w', 'x', 'y', 'z', '0', '1', '2', '3',  
  '4', '5', '6', '7', '8', '9', '+', '/',  
  '='  
);
```

```
95 $asciiMap: (  
96   " ": "00001001",  
97   " ": "00100000",  
98   " ": "00100001",  
99   "\": "00100010",  
00   "#": "00100011",  
01   "$": "00100100",  
02   "%": "00100101",  
03   "&": "00100110",  
04   "'": "00100111",  
05   "(": "00101000",
```

使用SASS来实现文本编码 — base64

<https://github.com/leeenx/svgToBase64>



当前页面二维码

SVG内联的应用

- <http://jdc.jd.com/fd/h5/1612/double12/climax.html#4> TAB打孔
- <http://jdc.jd.com/fd/h5/1612/store/preheat.html#0> 渐变边框
- <http://jdc.jd.com/fd/h5/1701/ny/nyone.html> 伪元素 after/before 不够用？

tab打孔的实现：

<https://github.com/leeenx/tabpunch>



当前页面二维码

THANKS
FOR YOUR WATCHING



OPEN ORIENTED

凹凸实验室