SVG 教程

SVG 意为可缩放矢量图形（Scalable Vector Graphics）。

SVG 使用 XML 格式定义图像。

现在开始学习 SVG!

SVG 实例

```
<html>
<body>
<h1>My first SVG</h1>
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<circle cx="100" cy="50" r="40" stroke="black"
stroke-width="2" fill="red" />
</svg>
</body>
</html>
```

尝试一下 »

点击 "尝试一下" 按钮查看运行结果.

SVG 实例

我们可以在线编辑 SVG 实例，并且在线查看运行结果!

在线实例

SVG 参考手册

在菜鸟教程中， 我们为您提供完整的 SVG 参考手册，其中列出了所有 W3C 推荐标准（SVG Version 1.1）中的 SVG 的元素。

SVG 参考手册

SVG 在线编辑器

首页 HTML CSS JS 本地书签

SVG 简介

SVG 是使用 XML 来描述二维图形和绘图程序的语言。

学习之前应具备的基础知识：

继续学习之前，你应该对以下内容有基本的了解：

- HTML
- XML 基础

如果希望首先学习这些内容，请在本站的首页选择相应的教程。

什么是SVG?

- SVG 指可伸缩矢量图形 (Scalable Vector Graphics)
- SVG 用来定义用于网络的基于矢量的图形
- SVG 使用 XML 格式定义图形
- SVG 图像在放大或改变尺寸的情况下其图形质量不会有所损失
- SVG 是万维网联盟的标准
- SVG 与诸如 DOM 和 XSL 之类的 W3C 标准是一个整体

SVG 是 W3C 推荐标准

SVG 于 2003 年 1 月 14 日成为 W3C 推荐标准。

如需阅读更多有关 W3C 的 SVG 活动的信息，请访问我们的 W3C 教程。

SVG 的历史和优势

在 2003 年一月，SVG 1.1 被确立为 W3C 标准。

参与定义 SVG 的组织有：Sun公司（已被Oracle公司收购）、Adobe、苹果公司、IBM 以及柯达。

与其他图像格式相比，使用 SVG 的优势在于：

- SVG 可被非常多的工具读取和修改（比如记事本）
- SVG 与 JPEG 和 GIF 图像比起来，尺寸更小，且可压缩性更强。
- SVG 是可伸缩的
- SVG 图像可在任何的分辨率下被高质量地打印
- SVG 可在图像质量不下降的情况下被放大
- SVG 图像中的文本是可选的，同时也是可搜索的（很适合制作地图）
- SVG 可以与 Java 技术一起运行
- SVG 是开放的标准
- SVG 文件是纯粹的 XML

SVG 的主要竞争者是 Flash。

与 Flash 相比，SVG 最大的优势是与其他标准（比如 XSL 和 DOM）相兼容。而 Flash 则是未开源的私有技术。

查看 SVG 文件

Internet Explorer9，火狐，谷歌Chrome，Opera和Safari都支持SVG。

IE8和早期版本都需要一个插件 - 如Adobe SVG浏览器，这是免费提供的。

创建SVG文件

由于SVG是XML文件，SVG图像可以用任何文本编辑器创建，但它往往是与一个绘图程序一起使用，如Inkscape，更方便地创建SVG图像。

SVG 实例

简单的 SVG 实例

一个简单的SVG图形例子：

这里是SVG文件（SVG文件的保存与SVG扩展）：

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <circle cx="100" cy="50" r="40" stroke="black"
  stroke-width="2" fill="red" />
</svg>
```

SVG 代码解析：

第一行包含了 XML 声明。请注意 standalone 属性！该属性规定此 SVG 文件是否是"独立的"，或含有对外部文件的引用。

standalone="no" 意味着 SVG 文档会引用一个外部文件 - 在这里，是 DTD 文件。

第二和第三行引用了这个外部的 SVG DTD。该 DTD 位于 "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd"。该 DTD 位于 W3C，含有所有允许的 SVG 元素。

SVG 代码以 <svg> 元素开始，包括开启标签 <svg> 和关闭标签 </svg> 。这是根元素。width 和 height 属性可设置此 SVG 文档的宽度和高度。version 属性可定义所使用的 SVG 版本，xmlns 属性可定义 SVG 命名空间。

SVG 的 <circle> 用来创建一个圆。cx 和 cy 属性定义圆中心的 x 和 y 坐标。如果忽略这两个属性，那么圆点会被设置为 (0, 0)。r 属性定义圆的半径。

stroke 和 stroke-width 属性控制如何显示形状的轮廓。我们把圆的轮廓设置为 2px 宽，黑边框。

fill 属性设置形状内的颜色。我们把填充颜色设置为红色。

关闭标签的作用是关闭 SVG 元素和文档本身。

注释：所有的开启标签必须有关闭标签！

SVG 在 HTML 页面

SVG 文件可通过以下标签嵌入 HTML 文档：<embed>、<object> 或者 <iframe>。

SVG的代码可以直接嵌入到HTML页面中，或您可以直接链接到SVG文件。

使用 <embed> 标签

<embed>:

优势：所有主要浏览器都支持，并允许使用脚本

缺点：不推荐在HTML4和XHTML中使用（但在HTML5允许）

语法：

```
<embed src="circle1.svg" type="image/svg+xml" />
```

结果：

使用 <object> 标签

<object>:

优势：所有主要浏览器都支持，并支持HTML4，XHTML和HTML5标准

缺点：不允许使用脚本。

语法：

```
<object data="circle1.svg" type="image/svg+xml"></object>
```

结果：

使用 <iframe> 标签

<iframe>:

优势：所有主要浏览器都支持，并允许使用脚本

缺点：不推荐在HTML4和XHTML中使用（但在HTML5允许）

语法：

```
<iframe src="circle1.svg"></iframe>
```

结果：

直接在HTML嵌入SVG代码

在Firefox、Internet Explorer9、谷歌Chrome和Safari中，你可以直接在HTML嵌入SVG代码。

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<circle cx="100" cy="50" r="40" stroke="black" stroke-width="2" fill="red" />
</svg>
```

尝试一下 »

链接到SVG文件

您还可以用<a>标签链接到一个SVG文件：链接到SVG文件

您还可以用<a>标签链接到一个SVG文件：

```
<a href="circle1.svg">View SVG file</a>
```

结果：

查看 SVG 文件

SVG 实例 SVG 矩形

点我分享笔记

反馈/建议

SVG 在 HTML 中 SVG 圆形

SVG <rect>

SVG Shapes

SVG有一些预定义的形状元素，可被开发者使用和操作：

- 矩形 <rect>
- 圆形 <circle>
- 椭圆 <ellipse>
- 线 <line>
- 折线 <polyline>
- 多边形 <polygon>
- 路径 <path>

下面的章节会为您讲解这些元素，首先从矩形元素开始。

SVG 矩形 - <rect>

实例 1

<rect> 标签可用来创建矩形，以及矩形的变种：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect width="300" height="100"
  style="fill:rgb(0,0,255);stroke-width:1;stroke:rgb(0,0,0)"/>
</svg>
```

尝试一下 »

对于Opera用户： 查看SVG文件（右键单击SVG图形预览源）。

代码解析：

rect 元素的 width 和 height 属性可定义矩形的高度和宽度

style 属性用来定义 CSS 属性

CSS 的 fill 属性定义矩形的填充颜色（rgb 值、颜色名或者十六进制值）

CSS 的 stroke-width 属性定义矩形边框的宽度

CSS 的 stroke 属性定义矩形边框的颜色

实例 2

让我们看看另一个例子，它包含一些新的属性：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="50" y="20" width="150" height="150"
  style="fill:blue;stroke:pink;stroke-width:5;fill-opacity:0.1;
  stroke-opacity:0.9"/>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

代码解析：

x 属性定义矩形的左侧位置（例如，x="0" 定义矩形到浏览器窗口左侧的距离是 0px）

y 属性定义矩形的顶端位置（例如，y="0" 定义矩形到浏览器窗口顶端的距离是 0px）

CSS 的 fill-opacity 属性定义填充颜色透明度（合法的范围是：0 - 1）

CSS 的 stroke-opacity 属性定义轮廓颜色的透明度（合法的范围是：0 - 1）

实例 3

定义整个元素的不透明度：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="50" y="20" width="150" height="150"
  style="fill:blue;stroke:pink;stroke-width:5;opacity:0.5"/>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

CSS opacity 属性用于定义了元素的透明值 (范围: 0 到 1)。

实例 4

最后一个例子，创建一个圆角矩形：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="50" y="20" rx="20" ry="20" width="150" height="150"
  style="fill:red;stroke:black;stroke-width:5;opacity:0.5"/>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

rx 和 ry 属性可使矩形产生圆角。

SVG 在 HTML 中 SVG 圆形

SVG 矩形 SVG 椭圆

SVG <circle>

SVG 圆形 - <circle>

<circle> 标签可用来创建一个圆：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <circle cx="100" cy="50" r="40" stroke="black"
  stroke-width="2" fill="red"/>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

代码解析：

cx和cy属性定义圆点的x和y坐标。如果省略cx和cy，圆的中心会被设置为(0, 0)

r属性定义圆的半径

SVG 矩形 SVG 椭圆

点我分享笔记

反馈/建议

Copyright © 2013-2018 菜鸟教程 runoob.com All Rights Reserved. 备案号：闽ICP备15012807号-1

SVG <ellipse>

SVG 椭圆 - <ellipse>

实例 1

<ellipse> 元素是用来创建一个椭圆：

椭圆与圆很相似。不同之处在于椭圆有不同的x和y半径，而圆的x和y半径是相同的：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<ellipse cx="300" cy="80" rx="100" ry="50"
style="fill:yellow;stroke:purple;stroke-width:2"/>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

代码解析：

CX属性定义的椭圆中心的x坐标

CY属性定义的椭圆中心的y坐标

RX属性定义的水平半径

RY属性定义的垂直半径

实例 2

下面的例子创建了三个累叠而上的椭圆：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<ellipse cx="240" cy="100" rx="220" ry="30" style="fill:purple"/>
<ellipse cx="220" cy="70" rx="190" ry="20" style="fill:lime"/>
<ellipse cx="210" cy="45" rx="170" ry="15" style="fill:yellow"/>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

实例 3

下面的例子组合了两个椭圆（一个黄的和一个白的）：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<ellipse cx="240" cy="50" rx="220" ry="30" style="fill:yellow"/>
<ellipse cx="220" cy="50" rx="190" ry="20" style="fill:white"/>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

SVG 圆形 SVG 直线

点我分享笔记

反馈/建议

SVG 椭圆 SVG 多边形

SVG <line>

SVG 直线 - <line>

<line> 元素是用来创建一个直线：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <line x1="0" y1="0" x2="200" y2="200"
  style="stroke:rgb(255,0,0);stroke-width:2"/>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件右键单击SVG图形预览源）。

x1 属性在 x 轴定义线条的开始

y1 属性在 y 轴定义线条的开始

x2 属性在 x 轴定义线条的结束

y2 属性在 y 轴定义线条的结束

SVG 椭圆 SVG 多边形

点我分享笔记

反馈/建议

Copyright © 2013-2018 菜鸟教程 runoob.com All Rights Reserved. 备案号：闽ICP备15012807号-1

SVG <polygon>

SVG 多边形 - <polygon>

实例 1

<polygon> 标签用来创建含有不少于三个边的图形。

多边形是由直线组成，其形状是"封闭"的（所有的线条 连接起来）。

polygon来自希腊。 "Poly" 意味 "many"， "gon" 意味 "angle".

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <polygon points="200,10 250,190 160,210"
  style="fill:lime;stroke:purple;stroke-width:1"/>
</svg>
```

尝试一下 »

对于Opera用户： 查看SVG文件（右键单击SVG图形预览源）。

代码解析：

points 属性定义多边形每个角的 x 和 y 坐标

实例 2

下面的示例创建一个四边的多边形：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <polygon points="220,10 300,210 170,250 123,234"
  style="fill:lime;stroke:purple;stroke-width:1"/>
</svg>
```

尝试一下 »

对于Opera用户： 查看SVG文件（右键单击SVG图形预览源）。

实例 3

使用 <polygon> 元素创建一个星型:

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <polygon points="100,10 40,180 190,60 10,60 160,180"
  style="fill:lime;stroke:purple;stroke-width:5;fill-rule:nonzero;" />
</svg>
```

尝试一下 »

对于Opera用户： 查看SVG文件（右键单击SVG图形预览源）。

实例 4

改变 fill-rule 属性为 "evenodd":

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <polygon points="100,10 40,180 190,60 10,60 160,180"
  style="fill:lime;stroke:purple;stroke-width:5;fill-rule:evenodd;" />
</svg>
```

尝试一下 »

对于Opera用户： 查看SVG文件（右键单击SVG图形预览源）。

1 篇笔记

#1

写笔记

SVG的图形填充规则通过fill-rule属性来指定。

fill-rule

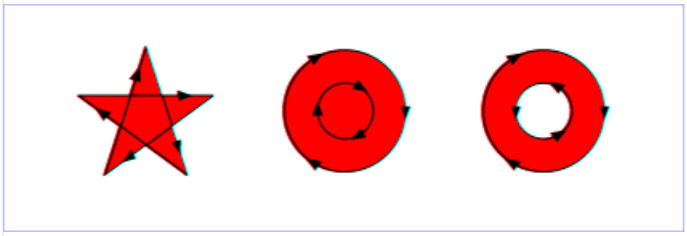
有效值:	nonzero | evenodd | inherit
默认值:	nonzero
应用于:	shape形状类元素 和 文字内容类元素
可继承:	是
比例:	无
媒体:	可见
动画可用:	是

fill-rule 属性用于指定使用哪一种算法去判断画布上的某区域是否属于该图形“内部”（内部区域将被填充）。对一个简单的无交叉的路径，哪块区域是“内部”是很直观清除的。但是，对一个复杂的路径，比如自相交或者一个子路径包围另一个子路径，“内部”的理解就不那么明确了。

fill-rule 属性提供两种选项用于指定如何判断图形的“内部”:

nonzero

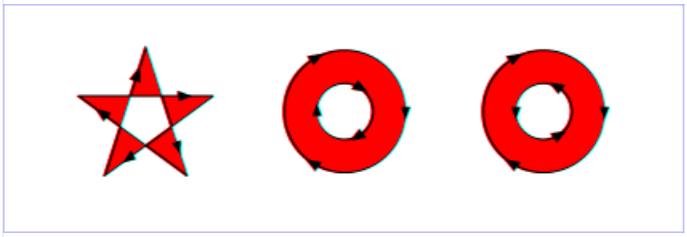
字面意思是“非零”。按该规则，要判断一个点是否在图形内，从该点作任意方向的一条射线，然后检测射线与图形路径的交点情况。从0开始计数，路径从左向右穿过射线则计数加1，从右向左穿过射线则计数减1。得出计数结果后，如果结果是0，则认为点在图形外部，否则认为在内部。下图演示了nonzero规则:



点击查看示例SVG文件 (仅适用于支持SVG的浏览器)

evenodd

字面意思是“奇偶”。按该规则，要判断一个点是否在图形内，从该点作任意方向的一条射线，然后检测射线与图形路径的交点的数量。如果结果是奇数则认为点在内部，是偶数则认为点在外部。下图演示了evenodd 规则:



点击查看示例SVG文件 (仅适用于支持SVG的浏览器)

提示: 上述解释未指出当路径片段与射线重合或者相切的时候怎么办，因为任意方向的射线都可以，那么只需要简单的选择另一条没有这种特殊情况的射线即可。

没有水的木1年前
(2017-09-08)

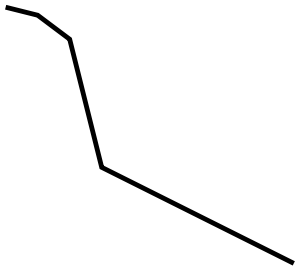
反馈/建议

SVG <polyline>

SVG 曲线 - <polyline>

实例 1

<polyline> 元素是用于创建任何只有直线的形状：



下面是SVG代码：

实例

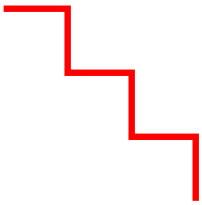
```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<polyline points="20,20 40,25 60,40 80,120 120,140 200,180"
style="fill:none;stroke:black;stroke-width:3" />
</svg>
```

尝试一下 »

F对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

实例 2

只有直线的另一个例子：



下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<polyline points="0,40 40,40 40,80 80,80 80,120 120,120 120,160" style="fill:white;stroke:red;stroke-width:4" />
</svg>
```

尝试一下 »

F对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

SVG <path>

SVG 路径 - <path>

<path> 元素用于定义一个路径。

下面的命令可用于路径数据：

M = moveto

L = lineto

H = horizontal lineto

V = vertical lineto

C = curveto

S = smooth curveto

Q = quadratic Bézier curve

T = smooth quadratic Bézier curveto

A = elliptical Arc

Z = closepath

注意：以上所有命令均允许小写字母。大写表示绝对定位，小写表示相对定位。

实例 1

上面的例子定义了一条路径，它开始于位置150 0，到达位置75 200，然后从那里开始到225 200，最后在150 0关闭路径。

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<path d="M150 0 L75 200 L225 200 Z" />
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

实例 2

下面的例子创建了一个二次方贝塞尔曲线，A 和 C 分别是起点和终点，B 是控制点：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<path id="lineAB" d="M 100 350 l 150 -300" stroke="red"
stroke-width="3" fill="none" />
<path id="lineBC" d="M 250 50 l 150 300" stroke="red"
stroke-width="3" fill="none" />
<path d="M 175 200 l 150 0" stroke="green" stroke-width="3"
fill="none" />
<path d="M 100 350 q 150 -300 300 0" stroke="blue"
stroke-width="5" fill="none" />
<!-- Mark relevant points -->
<g stroke="black" stroke-width="3" fill="black">
<circle id="pointA" cx="100" cy="350" r="3" />
<circle id="pointB" cx="250" cy="50" r="3" />
<circle id="pointC" cx="400" cy="350" r="3" />
</g>
<!-- Label the points -->
<g font-size="30" font="sans-serif" fill="black" stroke="none"
text-anchor="middle">
<text x="100" y="350" dx="-30">A</text>
<text x="250" y="50" dy="-10">B</text>
<text x="400" y="350" dx="30">C</text>
</g>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

复杂吗？是的！！由于在绘制路径时的复杂性，强烈建议使用SVG编辑器来创建复杂的图形。

SVG 多边形

SVG 曲线

点我分享笔记

反馈/建议

SVG <text>

SVG 文本 - <text>

<text> 元素用于定义文本。

实例 1

写一个文本：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <text x="0" y="15" fill="red">I love SVG</text>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

实例 2

旋转的文字：

下面是SVG代码：

实例

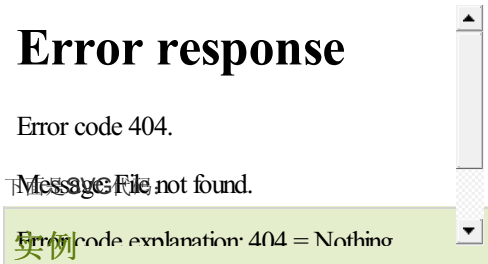
```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <text x="0" y="15" fill="red" transform="rotate(30 20,40)">I love SVG</text>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

实例 3

路径上的文字：



下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1"
xmlns:xlink="http://www.w3.org/1999/xlink">
   <defs>
```

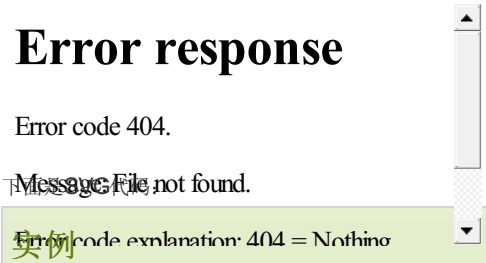
```
    <path id="path1" d="M75,20 a1,1 0 0,0 100,0" />
  </defs>
  <text x="10" y="100" style="fill:red;">
    <textPath xlink:href="#path1">I love SVG I love SVG</textPath>
  </text>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

实例 4

元素可以安排任何分小组与<tspan> 元素的数量。每个<tspan> 元素可以包含不同的格式和位置。几行文本(与 <tspan> 元素):



下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <text x="10" y="20" style="fill:red;">Several lines:
    <tspan x="10" y="45">First line</tspan>
    <tspan x="10" y="70">Second line</tspan>
  </text>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

实例 5

作为链接文本（ <a> 元素）：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1"
xmlns:xlink="http://www.w3.org/1999/xlink">
  <a xlink:href="http://www.w3schools.com/svg/" target="_blank">
    <text x="0" y="15" fill="red">I love SVG</text>
  </a>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

SVG 曲线

SVG Stroke 属性

点我分享笔记

反馈/建议

SVG Stroke 属性

SVG Stroke 属性

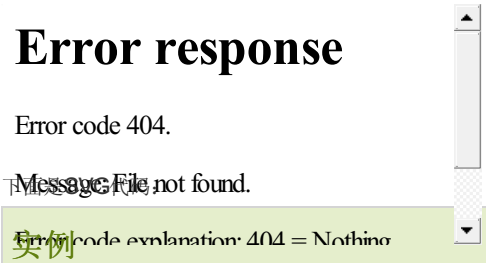
SVG提供了一个范围广泛stroke 属性。在本章中，我们将看看下面：

- stroke
- stroke-width
- stroke-linecap
- stroke-dasharray

所有stroke属性，可应用于任何种类的线条，文字和元素就像一个圆的轮廓。

SVG stroke 属性

Stroke属性定义一条线，文本或元素轮廓颜色：



实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <g fill="none">
    <path stroke="red" d="M5 20 l215 0" />
    <path stroke="blue" d="M5 40 l215 0" />
    <path stroke="black" d="M5 60 l215 0" />
  </g>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

SVG stroke-width 属性

Tstroke- width属性定义了一条线，文本或元素轮廓厚度：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <g fill="none" stroke="black">
    <path stroke-width="2" d="M5 20 l215 0" />
    <path stroke-width="4" d="M5 40 l215 0" />
    <path stroke-width="6" d="M5 60 l215 0" />
```

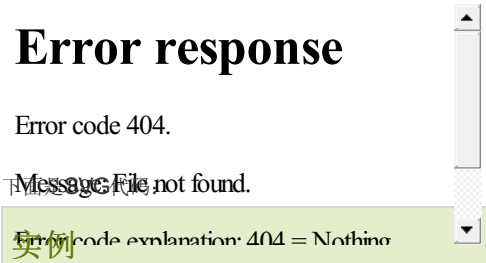
```
  </g>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

SVG stroke-linecap 属性

strokelinecap属性定义不同类型的开放路径的终结：



实例

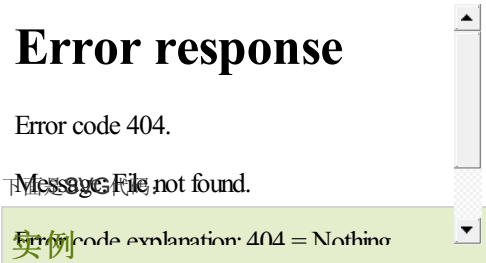
```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <g fill="none" stroke="black" stroke-width="6">
    <path stroke-linecap="butt" d="M5 20 l215 0" />
    <path stroke-linecap="round" d="M5 40 l215 0" />
    <path stroke-linecap="square" d="M5 60 l215 0" />
  </g>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

SVG stroke-dasharray 属性

strokedasharray属性用于创建虚线：



实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <g fill="none" stroke="black" stroke-width="4">
    <path stroke-dasharray="5,5" d="M5 20 l215 0" />
    <path stroke-dasharray="10,10" d="M5 40 l215 0" />
    <path stroke-dasharray="20,10,5,5,5,10" d="M5 60 l215 0" />
  </g>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

SVG 文本

SVG 滤镜

点我分享笔记

反馈/建议

SVG 滤镜

SVG滤镜用来增加对SVG图形的特殊效果。

SVG 滤镜

在本教程中，我们将仅展示一个可能采用的特殊效果。基础知识展示后，你已经学会使用特殊效果，你应该能够适用于其他地方。这里的关键是给你一个怎样做SVG的想法，而不是重复整个规范。

SVG可用的滤镜是：

- feBlend - 与图像相结合的滤镜
- feColorMatrix - 用于彩色滤光片转换
- feComponentTransfer
- feComposite
- feConvolveMatrix
- feDiffuseLighting
- feDisplacementMap
- feFlood
- feGaussianBlur
- feImage
- feMerge
- feMorphology
- feOffset - 过滤阴影
- feSpecularLighting
- feTile
- feTurbulence
- feDistantLight - 用于照明过滤
- fePointLight - 用于照明过滤
- feSpotLight - 用于照明过滤

除此之外，您可以在每个 SVG 元素上使用多个滤镜！

SVG 模糊效果

注意： Internet Explorer和Safari不支持SVG滤镜！

<defs> 和 <filter>

所有互联网的SVG滤镜定义在<defs>元素中。<defs>元素定义短并含有特殊元素（如滤镜）定义。

<filter>标签用来定义SVG滤镜。<filter>标签使用必需的id属性来定义向图形应用哪个滤镜？

SVG <feGaussianBlur>

实例 1

<feGaussianBlur> 元素是用于创建模糊效果：

fegaussianblur

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <defs>
    <filter id="f1" x="0" y="0">
      <feGaussianBlur in="SourceGraphic" stdDeviation="15" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
  fill="yellow" filter="url(#f1)" />
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

代码解析：

<filter>元素id属性定义一个滤镜的唯一名称

<feGaussianBlur>元素定义模糊效果

in="SourceGraphic"这个部分定义了由整个图像创建效果

stdDeviation属性定义模糊量

<rect>元素的滤镜属性用来把元素链接到"f1"滤镜

SVG 阴影

注意： Internet Explorer和Safari不支持SVG滤镜！

<defs> 和 <filter>

所有互联网的SVG滤镜定义在<defs>元素中。<defs>元素定义短并含有特殊元素（如滤镜）定义。

<filter>标签用来定义SVG滤镜。<filter>标签使用必需的id属性来定义向图形应用哪个滤镜？

SVG <feOffset>

实例 1

<feOffset>元素是用于创建阴影效果。我们的想法是采取一个SVG图形（图像或元素）并移动它在xy平面上一点儿。

第一个例子偏移一个矩形（带<feOffset>），然后混合偏移图像顶部（含<feBlend>）：

feoffset

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <defs>
    <filter id="f1" x="0" y="0" width="200%" height="200%">
      <feOffset result="offOut" in="SourceGraphic" dx="20" dy="20" />
      <feBlend in="SourceGraphic" in2="offOut" mode="normal" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
  fill="yellow" filter="url(#f1)" />
</svg>
```

尝试一下 »

对于Opera用户： 查看SVG文件（右键单击SVG图形预览源）。

代码解析：

<filter>元素id属性定义一个滤镜的唯一名称

<rect>元素的滤镜属性用来把元素链接到"f1"滤镜

实例 2

现在，偏移图像可以变的模糊（含 <feGaussianBlur>）：

feoffset2

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <defs>
    <filter id="f1" x="0" y="0" width="200%" height="200%">
      <feOffset result="offOut" in="SourceGraphic" dx="20" dy="20" />
      <feGaussianBlur result="blurOut" in="offOut" stdDeviation="10" />
      <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
```

```
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
  fill="yellow" filter="url(#f1)" />
</svg>
```

尝试一下 »

对于Opera用户： 查看SVG文件（右键单击SVG图形预览源）。

代码解析：

<feGaussianBlur>元素的stdDeviation属性定义了模糊量

实例 3

现在，制作一个黑色的阴影：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <defs>
    <filter id="f1" x="0" y="0" width="200%" height="200%">
      <feOffset result="offOut" in="SourceAlpha" dx="20" dy="20" />
      <feGaussianBlur result="blurOut" in="offOut" stdDeviation="10" />
      <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
  fill="yellow" filter="url(#f1)" />
</svg>
```

尝试一下 »

对于Opera用户： 查看SVG文件（右键单击SVG图形预览源）。

代码解析：

<feOffset>元素的属性改为"SourceAlpha"在Alpha通道使用残影，而不是整个RGBA像素。

实例 4

现在为阴影涂上一层颜色：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <defs>
    <filter id="f1" x="0" y="0" width="200%" height="200%">
      <feOffset result="offOut" in="SourceGraphic" dx="20" dy="20" />
      <feColorMatrix result="matrixOut" in="offOut" type="matrix"
      values="0.2 0 0 0 0 0 0.2 0 0 0 0 0 0.2 0 0 0 0 0 1 0" />
      <feGaussianBlur result="blurOut" in="matrixOut" stdDeviation="10" />
      <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
  fill="yellow" filter="url(#f1)" />
</svg>
```

尝试一下 »

对于Opera用户： 查看SVG文件（右键单击SVG图形预览源）。

代码解析：

<feColorMatrix>过滤器是用来转换偏移的图像使之更接近黑色的颜色。 '0.2'矩阵的三个值都获取乘以红色，绿色和蓝色通道。降低其值带来的颜

色至黑色（黑色为0）

SVG 模糊效果 – 高斯模糊　　SVG 渐变 – 线性

1 篇笔记
#1

写笔记

变换矩阵的定义和说明

feColorMatrix 的 matrix 是一个 4*5 的矩阵。前面 4 列是颜色通道的比例系数，最后一列是常量偏移。

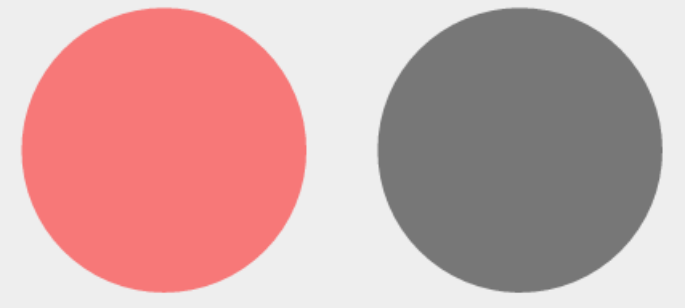
$$\begin{bmatrix} rr & rg & rb & ra & c1 \\ gr & gg & gb & ga & c2 \\ br & bg & bb & ba & c3 \\ ar & ag & ab & aa & c4 \end{bmatrix} \bullet \begin{bmatrix} r \\ g \\ b \\ a \\ 1 \end{bmatrix} = \begin{bmatrix} r*rr+g*rg+b*rb+a*ra+c1*1 \\ r*gr+g*gg+b*gb+a*ga+c2*1 \\ r*br+g*bg+b*bb+a*ba+c3*1 \\ r*ar+g*ag+b*ab+a*aa+c4*1 \end{bmatrix}$$

feColorMatrix矩阵乘法公式说明 - http://techbrood.com

上面公式中的 rr 表示 red to red 系数，以此类推。c1~c4 表示常量偏移。
第一个 4*5 矩阵为变换矩阵，第二个单列矩阵为待变换对象的像素值。右侧单列矩阵为矩阵 1 和 2 的点积结果。
这个变换矩阵看起来比较复杂，在实践上常使用一个简化的对角矩阵，即除了 rr/gg/bb/aa 取值非零外，其余行列取值为 0，这就退化成了简单的各颜色通道的独立调整。
feColorMatrix的语法:

```
<filter id="f1" x="0%" y="0%" width="100%" height="100%">
  <feColorMatrix
     result="original" id="c1" type="matrix"
     values="1 0 0 0 0
             0 1 0 0 0
             0 0 1 0 0
             0 0 0 1 0" />
</filter>
```

上述feColorMatrix过滤器的类型值为matrix，除此之外，还有saturate（饱和度）和hueRotate（色相旋转），取值比较简单，这里不做说明。
显然当变换矩阵为单位对角矩阵时，变换结果和原值相等。
我们可以尝试调整比例系数，比如把rr的值设置为0，即去除图像中的red颜色通道含量：



尝试一下 »

smilenow1年前 (2017-09-12)

反馈/建议

SVG 渐变 - 线性

SVG 渐变

渐变是一种从一种颜色到另一种颜色的平滑过渡。另外，可以把多个颜色的过渡应用到同一个元素上。

SVG渐变主要有两种类型：

- Linear
- Radial

SVG 线性渐变 - <linearGradient>

<linearGradient>元素用于定义线性渐变。

<linearGradient>标签必须嵌套在<defs>的内部。<defs>标签是definitions的缩写，它可对诸如渐变之类的特殊元素进行定义。

线性渐变可以定义为水平，垂直或角渐变：

- 当y1和y2相等，而x1和x2不同时，可创建水平渐变
- 当x1和x2相等，而y1和y2不同时，可创建垂直渐变
- 当x1和x2不同，且y1和y2不同时，可创建角形渐变

实例 1

定义水平线性渐变从黄色到红色的椭圆形：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <defs>
    <linearGradient id="grad1" x1="0%" y1="0%" x2="100%" y2="0%">
      <stop offset="0%" style="stop-color:rgb(255,255,0);stop-opacity:1" />
      <stop offset="100%" style="stop-color:rgb(255,0,0);stop-opacity:1" />
    </linearGradient>
  </defs>
  <ellipse cx="200" cy="70" rx="85" ry="55" fill="url(#grad1)" />
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

代码解析：

- <linearGradient>标签的id属性可为渐变定义一个唯一的名称
- <linearGradient>标签的X1，X2，Y1，Y2属性定义渐变开始和结束位置
- 渐变的颜色范围可由两种或多种颜色组成。每种颜色通过一个<stop>标签来规定。offset属性用来定义渐变的开始和结束位置。
- 填充属性把 ellipse 元素链接到此渐变

实例 2

定义一个垂直线性渐变从黄色到红色的椭圆形：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <defs>
    <linearGradient id="grad1" x1="0%" y1="0%" x2="0%" y2="100%">
      <stop offset="0%" style="stop-color:rgb(255,255,0);stop-opacity:1" />
      <stop offset="100%" style="stop-color:rgb(255,0,0);stop-opacity:1" />
    </linearGradient>
  </defs>
  <ellipse cx="200" cy="70" rx="85" ry="55" fill="url(#grad1)" />

</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

实例 3

定义一个椭圆形，水平线性渐变从黄色到红色并添加一个椭圆内文本：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <defs>
    <linearGradient id="grad1" x1="0%" y1="0%" x2="100%" y2="0%">
      <stop offset="0%" style="stop-color:rgb(255,255,0);stop-opacity:1" />
      <stop offset="100%" style="stop-color:rgb(255,0,0);stop-opacity:1" />
    </linearGradient>
  </defs>
  <ellipse cx="200" cy="70" rx="85" ry="55" fill="url(#grad1)" />
  <text fill="#ffffff" font-size="45" font-family="Verdana" x="150" y="86">
  SVG</text>
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

代码解析：

<text> 元素是用来添加一个文本

□ SVG 阴影　　SVG 渐变- 放射性 □

□点我分享笔记

反馈/建议

□

SVG 渐变- 放射性

SVG 放射性渐变 - <radialGradient>

<radialGradient>元素用于定义放射性渐变。

<radialGradient>标签必须嵌套在<defs>的内部。<defs>标签是definitions的缩写，它可对诸如渐变之类的特殊元素进行定义。

实例 1

定义一个放射性渐变从白色到蓝色椭圆：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <defs>
    <radialGradient id="grad1" cx="50%" cy="50%" r="50%" fx="50%" fy="50%">
      <stop offset="0%" style="stop-color:rgb(255,255,255);
      stop-opacity:0" />
      <stop offset="100%" style="stop-color:rgb(0,0,255);stop-opacity:1" />
    </radialGradient>
  </defs>
  <ellipse cx="200" cy="70" rx="85" ry="55" fill="url(#grad1)" />
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

代码解析：

<radialGradient>标签的 id 属性可为渐变定义一个唯一的名称

CX，CY和r属性定义的最外层圆和Fx和Fy定义的最内层圆

渐变颜色范围可以由两个或两个以上的颜色组成。每种颜色用一个<stop>标签指定。offset属性用来定义渐变色开始和结束

填充属性把ellipse元素链接到此渐变

实例 2

定义放射性渐变从白色到蓝色的另一个椭圆：

下面是SVG代码：

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <defs>
    <radialGradient id="grad1" cx="20%" cy="30%" r="30%" fx="50%" fy="50%">
      <stop offset="0%" style="stop-color:rgb(255,255,255);
      stop-opacity:0" />
      <stop offset="100%" style="stop-color:rgb(0,0,255);stop-opacity:1" />
    </radialGradient>
  </defs>
  <ellipse cx="200" cy="70" rx="85" ry="55" fill="url(#grad1)" />
</svg>
```

尝试一下 »

对于Opera用户：查看SVG文件（右键单击SVG图形预览源）。

反馈/建议

SVG 实例

在线实例

下面的例子是把SVG代码直接嵌入到HTML代码中。

谷歌Chrome，火狐，Internet Explorer9，和Safari都支持。

注意： 下面的例子将不会在Opera运行，即使Opera支持SVG - 它也不支持SVG在HTML代码中直接使用。

SVG 实例

SVG基本形状

一个圆

矩形

不透明矩形

一个矩形不透明2

一个带圆角矩形

一个椭圆

累叠而上的三个椭圆

两个椭圆

一条线

三角形

四边形

一个星星

另一个星星

折线

另一个折线

路径

二次贝塞尔曲线

编写文字

旋转文本

路径上文字

几行文字

文本链接

定义一条线，文本或轮廓颜色（stroke）

定义一条线宽度，文本或轮廓（stroke-width）

stroke-linecap属性定义不同类型的开放路径的终结：

定义虚线（stroke-dasharray）

SVG滤镜

feGaussianBlur - 模糊效果

feOffset - 偏移一个矩形，然后混合偏移图像顶部

feOffset - 模糊偏移图像

feOffset - 使阴影变黑色

feOffset - 为阴影涂上一层颜色

一个feBlend滤镜

一个feColorMatrix滤镜

一个feComponentTransfer滤镜

feOffset、feFlood、feComposite、feMerge 以及 feMergeNode

一个feMorphology滤镜

滤镜1

滤镜2

滤镜3

滤镜4

滤镜5

滤镜6

SVG渐变

水平线性渐变从黄色到红色的椭圆形

垂直线性渐变从黄色到红色椭圆形

水平线性渐变从黄色到红色并添加一个椭圆内文本

放射性渐变从白色到蓝色椭圆

放射性渐变从白色到蓝色的另一个椭圆

SVG杂项

重复用 5 秒时间淡出的矩形

矩形会变大并改变颜色

会改变颜色的三个矩形

沿一个运动路径移动的文本

沿一个运动路径移动、旋转并缩放的文本

沿一个运动路径移动、旋转并缩放的文本 + 逐步放大并改变颜色的矩形

□ SVG 渐变- 放射性　　SVG 参考手册 □

SVG 参考手册

SVG 元素

元素	说明	属性
<a>	创建一个SVG元素周围链接	xlink:show xlink:actuate xlink:href target
<altGlyph>	允许对象性文字进行控制，来呈现特殊的字符数据	x y dx dy rotate glyphRef format xlink:href
<altGlyphDef>	定义一系列象性符号的替换	id
<altGlyphItem>	定义一系列候选的象性符号的替换	id
<animate>	随时间动态改变属性	attributeName="目标属性名称" from="起始值" to="结束值" dur="持续时间" repeatCount="动画时间将发生"
<animateColor>	定义随着时间的推移颜色转换	by="相对偏移值" from="起始值" to="结束值"
<animateMotion>	使元素沿着动作路径移动	calcMode="动画的插补模式。可以是'discrete', 'linear', 'paced', 'spline'" path="运动路径" keyPoints="沿运动路径的对象目前时间应移动多远" rotate="应用旋转变换" xlink:href="一个URI引用<path>元素，它定义运动路径"
<animateTransform>	动画上一个目标元素变换属性，从而使动画控制平移，缩放，旋转或倾斜	by="相对偏移值" from="起始值" to="结束值" type="类型的转换其值是随时间变化。可以是 'translate', 'scale', 'rotate', 'skewX', 'skewY'"
<circle>	定义一个圆	cx="圆的x轴坐标" cy="圆的y轴坐标" r="圆的半径". 必需. + 显现属性：颜色，FillStroke，图形
<clipPath>	用于隐藏位于剪切路径以外的对象部分。定义绘制什么和什么不绘制的模具被称为剪切路径	clip-path="引用剪贴路径和引用剪贴路径交叉" clipPathUnits="userSpaceOnUse'或'objectBoundingBox"。第二个值childern一个对象的边框，会使用掩码的一小部分单位（默认："userSpaceOnUse"）"

<color-profile>	指定颜色配置文件的说明（使用CSS样式文件时）	local="本地存储颜色配置文件唯一ID" name="" rendering-intent="auto|perceptual|relative-colorimetric|saturation|absolute-colorimetric" xlink:href="ICC配置文件资源URI"
<cursor>	定义一个独立于平台的自定义光标	x="x轴左上角光标（默认为0）" y="y轴的左上角光标（默认为0）" xlink:href="使用光标图像URI
<defs>	引用的元素容器	
<desc>	对 SVG 中的元素的纯文本描述 - 并不作为图形的一部分来显示。用户代理会将其显示为工具提示	
<ellipse>	定义一个椭圆	cx="椭圆x轴坐标" cy="椭圆y轴坐标" rx="沿x轴椭圆形的半径"。必需。 ry="沿y轴长椭圆形的半径"。必需。 + 显现属性：颜色，FillStroke，图形
<feBlend>	使用不同的混合模式把两个对象合成在一起	mode="图像混合模式：normal|multiply|screen|darken|lighten" in="标识为给定的滤镜原始输入：SourceGraphic | SourceAlpha | BackgroundImage | BackgroundAlpha | FillPaint | StrokePaint | <filter-primitive-reference>" in2="第二输入图像的混合操作"
feColorMatrix	SVG滤镜。适用矩阵转换	
feComponentTransfer	SVG 滤镜。执行数据的 component-wise 重映射	
feComposite	SVG 滤镜	
feConvolveMatrix	SVG 滤镜	
feDiffuseLighting	SVG 滤镜	
feDisplacementMap	SVG 滤镜	
feDistantLight	SVG滤镜。定义一个光源	
feFlood	SVG滤镜	
feFuncA	SVG 滤镜。feComponentTransfer 的子元素	
feFuncB	SVG 滤镜。feComponentTransfer 的子元素	
feFuncG	SVG 滤镜。feComponentTransfer 的子元素	

feFuncR	SVG 滤镜。feComponentTransfer 的子元素	
feGaussianBlur	SVG滤镜。执行高斯模糊图像	
feImage	SVG滤镜。	
feMerge	SVG滤镜。建立在彼此顶部图像层	
feMergeNode	SVG 滤镜。feMerge的子元素	
feMorphology	SVG 滤镜。 对源图形执行"fattening" 或者 "thinning"	
feOffset	SVG滤镜。相对其当前位置移动图像	
fePointLight	SVG滤镜	
feSpecularLighting	SVG滤镜	
feSpotLight	SVG滤镜	
feTile	SVG滤镜	
feTurbulence	SVG滤镜	
filter	滤镜效果的容器	
font	定义字体	
font-face	描述一种字体的特点	
font-face-format		
font-face-name		
font-face-src		
font-face-uri		
foreignObject		
<g>	用于把相关元素进行组合的容器元素	id="该组的名称" fill="该组填充颜色" opacity="该组不透明度" + 显现属性: All
glyph	为给定的象形符号定义图形	
glyphRef	定义要使用的可能的象形符号	
hkern		

<image>	定义图像	x="图像的左上角的x轴坐标" y="图像的左上角的y轴坐标" width="图像的宽度". 必须. height="图像的高度". 必须. xlink:href="图像的路径". 必须. + 显现属性: Color, Graphics, Images, Viewports
<line>	定义一条线	x1="直线起始点x坐标" y1="直线起始点y坐标" x2="直线终点x坐标" y2="直线终点y坐标" + 显现属性: Color, FillStroke, Graphics, Markers
<linearGradient>	定义线性渐变。通过使用矢量线性渐变填充对象，并可以定义为水平，垂直或角渐变。	id="id 属性可为渐变定义一个唯一的名称。引用必须" gradientUnits="'userSpaceOnUse' or 'objectBoundingBox'.使用视图框或对象，以确定相对位置矢量点。（默认为'objectBoundingBox）" gradientTransform="适用于渐变的转变" x1="渐变向量x启动点（默认0%）" y1="渐变向量y启动点（默认0%）" x2="渐变向量x的终点。（默认100%）" y2="渐变向量y的终点。（默认0%）" spreadMethod="'pad' or 'reflect' or 'repeat'" xlink:href="reference to another gradient whose attribute values are used as defaults and stops included. Recursive"
<marker>	标记可以放在直线，折线，多边形和路径的顶点。这些元素可以使用marker属性的"marker-start"，"marker-mid"和"marker-end"，继承默认情况下或可设置为"none"或定义的标记的URI。您必须先定义标记，然后才可以通过其URI引用。任何一种形状，可以把标记放在里面。他们绘制元素时把它们附加到顶部	markerUnits="strokeWidth'或'userSpaceOnUse"。如果是strokeWidth"那么使用一个单位等于一个笔划宽度。否则，标记尺度不会使用同一视图单位作为引用元素（默认为'strokeWidth'）" refx="标记顶点连接的位置（默认为0）" refy="标记顶点连接的位置（默认为0）" orient="'auto'始终显示标记的角度。 "auto"将计算某个角度使得X轴一个顶点的正切值（默认为0） markerWidth="标记的宽度（默认3）" markerHeight="标记的高度（默认3）" viewBox="各点"看到"这个SVG绘图区域。由空格或逗号分隔的4个值。(min x, min y, width, height)" + presentation attributes: All

<mask>	度屏蔽是一种不透明度值的组合和裁剪。像裁剪，您可以使用图形，文字或路径定义掩码的部分。一个掩码的默认状态是完全透明的，也就是裁剪平面的对面的。在掩码的图形设置掩码的不透明部分	maskUnits="'userSpaceOnUse' or 'objectBoundingBox'.设定裁剪面是否是相对完整的视窗或对象（默认：'objectBoundingBox'）" maskContentUnits="第二个掩码相对对象的图形位置使用百分比'userSpaceOnUse'或'objectBoundingBox'（默认：'userSpaceOnUse'）" x="裁剪面掩码（默认值：-10%）" y="裁剪面掩码（默认值：-10%）" width="裁剪面掩码（默认是：120%）" height="裁剪面掩码（默认是：120%）"
metadata	指定元数据	
missing-glyph		
mpath		
<path>	定义一个路径	d="定义路径指令" pathLength="如果存在，路径将进行缩放，以便计算各点相当于此值的路径长度" transform="转换列表" + 显现属性: Color, FillStroke, Graphics, Markers
<pattern>	定义坐标，你想要的视图显示和视图的大小。然后添加到您的模式形状。该模式命中时重复视图框的边缘（可视范围）	id="用于引用这个模式的唯一ID。"必需的。 patternUnits="userSpaceOnUse'或'objectBoundingBox"。第二个值 X，Y，width，height 一个会使用模式对象的边框的小部分，单位（%）。" patternContentUnits="'userSpaceOnUse'或 'objectBoundingBox"' patternTransform="允许整个表达式进行转换" x="模式的偏移量，来自左上角（默认为0）" y="模式的偏移量，来自左上角（默认为0）" width="模式平铺的宽度（默认为100%）" height="模式平铺的高度（默认为100%）" viewBox="各点"看到"这个SVG绘图区域。由空格或逗号分隔的4个值。(min x, min y, width, height)" xlink:href="另一种模式，其属性值是默认值以及任何子类可以继承。递归"
<polygon>	定义一个包含至少三边图形	points="多边形的点。点的总数必须是偶数"。必需的。 fill-rule="FillStroke演示属性的部分" + 显现属性: Color, FillStroke, Graphics, Markers
<polyline>	定义只有直线组成的任意形状	points=折线上的"点"。必需的。 + 显现属性: Color, FillStroke, Graphics, Markers

<radialGradient>	定义放射性渐变。放射性渐变创建一个圆圈	gradientUnits="'userSpaceOnUse' or 'objectBoundingBox'. 使用视图框或对象以确定相对位置的矢量点。（默认为'objectBoundingBox）" gradientTransform="适用于渐变的变换" cx="渐变的中心点（数字或% - 50%是默认）" cy="渐变的中心点。（默认50%）" r="渐变的半径。（默认50%）" fx="渐变的焦点。（默认0%）" fy="渐变的焦点。（默认0%）" spreadMethod="'pad' or 'reflect' or 'repeat'" xlink:href="引用到另一个渐变，其属性值作为默认值。递归"
<rect>	定义一个矩形	x="矩形的左上角的x轴" y="矩形的左上角的y轴" rx="x轴的半径（round元素）" ry="y轴的半径（round元素）" width="矩形的宽度"。必需的。 height="矩形的高度"。必需的。 + 显现属性: Color, FillStroke, Graphics
script	脚本容器。（例如ECMAScript）	
set	设置一个属性值指定时间	
<stop>	渐变停止	offset="偏移停止（0到1/0%到100%）". 参考 stop-color="这个stop的颜色" stop-opacity="这个Stop的不透明度 (0到1)"
style	可使样式表直接嵌入SVG内容内部	
<svg>	创建一个SVG文档片段	x="左上角嵌入（默认为0）" y="左上角嵌入（默认为0）" width="SVG片段的宽度（默认为100%）" height="SVG片段的高度（默认为100%）" viewBox="点"seen"这个SVG绘图区域。由空格或逗号分隔的4个值。(min x, min y, width, height)" preserveAspectRatio="'none'或任何'xVALYVAL'的9种组合,VAL是"min"，"mid"或"max"。（默认情况下none）" zoomAndPan="'magnify' or 'disable'.Magnify选项允许用户平移和缩放您的文件（默认Magnify ）" xml="最外层<svg>元素都需要安装SVG和它的命名空间： xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" xml:space="preserve"" + 显现属性: All
switch		
symbol		

<text>	定义一个文本	x="列表的X-轴的位置。在文本中在第n个字符的位置在第n个x轴。如果后面存在额外的字符，耗尽他们最后一个字符之后放置的位置。 0是默认" y="列表的Y轴位置。（参考x）0是默认" dx="在字符的长度列表中移动相对最后绘制标志符号的绝对位置。（参考x）" dy="在字符的长度列表中移动相对最后绘制标志符号的绝对位置。（参考x）" rotate="一个旋转的列表。第n个旋转是第n个字符。附加字符没有给出最后的旋转值" textLength="SVG查看器将尝试显示文本之间的间距/或字形调整的文本目标长度。（默认：正常文本的长度）" lengthAdjust="告诉查看器，如果指定长度就尝试进行调整用以呈现文本。这两个值是'spacing'和'spacingAndGlyphs'" + 显现属性: Color, FillStroke, Graphics, FontSpecification, TextContentElements
textPath		
title	对 SVG 中的元素的纯文本描述 - 并不作为图形的一部分来显示。用户代理会将其显示为工具提示	
<tref>	引用任何SVG文档中的<text>元素和重用	相同的<TEXT>元素
<tspan>	元素等同于<text>，但可以在内部嵌套文本标记以及内部本身	Identical to the <text> element + in addition: xlink:href="引用一个<TEXT>元素"
<use>	使用URI引用一个<G>,<svg>或其他具有一个唯一的ID属性和重复的图形元素。复制的是原始的元素，因此文件中的原始存在只是一个参考。原始影响到所有副本的任何改变。	x="克隆元素的左上角的x轴" y="克隆元素的左上角的y轴" width="克隆元素的宽度" height="克隆元素的高度" xlink:href="URI引用克隆元素" + 显现属性: All
view		
vkern		

SVG 实例

点我分享笔记

反馈/建议