SVG

http://www.w3.org/TR/SVG11/

浏览器支持情况，查看can i use

Canvas 和 SVG 的区别：

SVG

SVG 是一种使用 XML 描述 2D 图形的语言。 SVG 基于 XML，这意味着 SVG DOM 中的每个元素都是可用的。您可以为某个元素附加 JavaScript 事件处理器。 在 SVG 中，每个被绘制的图形均被视为对象。如果 SVG 对象的属性发生变化，那么浏览器能够自动重现图形。

特点：

不依赖分辨率

最适合带有大型渲染区域的应用程序（比如谷歌地图）

复杂度高会减慢渲染速度（任何过度使用 DOM 的应用都不快）

不适合游戏应用

应用场景：高保真度的复杂图像比如建筑和工程图，组织结构图，生物图

Canvas

Canvas 通过 JavaScript 来绘制 2D 图形。

Canvas 是逐像素进行渲染的。

在 canvas 中，一旦图形被绘制完成，它就不会继续得到浏览器的关注。如果其位置发生变化，那么整个场景也需要重新绘制，包括任何或许已被图形覆盖的对象。

特点：

依赖分辨率

弱的文本渲染能力

能够以 .png 或 .jpg 格式保存结果图像

最适合图像密集型的游戏，其中的许多对象会被频繁重绘

应用场景： 在一个较小的空间绘制大量的对象 比如非交互式的实时数据可视化，比如气象图

针对视频的像素级操作

在高级场景下组合使用HTML5 Canvas和SVG

有些情况下可以组合使用HTML5Canvas和SVG来达到最佳效果。例如，一个基于 canvas的游戏可以利用矢量编辑程序生成的SVG图来实现雪碧图以达到伸缩性和减少下载量的目的。又或

者一个绘图程序可以利用SVG来构建用户界面并使用嵌入的canvas元素用来绘图。

SVG使用

浏览器直接打开

在html里面加入svg标签

用img标签插入 src

用css引入

svg基本结构

一个独立的SVG文件，他主要包括： XML声明 svg根元素 包括一个用来描述SVG的XML声明空间。

最小的SVG结构：

```
<?xml version="1.0"?>   XML声明方式
<svg xmlns="http://www.w3.org/2000/svg"> SVG的XML声明空间
<!-- SVG代码 -->
</svg>
```

创建基本形状

rect 创建矩形

x 属性定义矩形的左侧位置

y 属性定义矩形的顶端位置

width 和 height定义矩形的高度和宽度

rx 和 ry 属性可使矩形产生圆角。

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 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 属性定义笔触颜色的透明度（合法的范围是：0 - 1）

stroke-linecap 不同类型的开放路径的终结 (butt round square)

lineCap属性

lineCap 定义上下文中线的端点，可以有以下 3 个值。 butt：默认值，端点是垂直于线段边缘的平直边缘。 round：端点是在线段边缘处以线宽为直径的半圆。 square：端点是在选段边缘处以线宽为长、以一半线宽为宽的矩形。 stroke-dasharray 用于创建虚线

circle 创建圆形

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

创建椭圆

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

line 创建线条

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

polyline 创建折线

points 元素是用于创建任何只有直线的形状

有无逗号均可以

polygon 创建多边形

与polyline类似，会自动闭合线段

‘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:

```
<polygon points="100,10 40,180 190,60 10,60 160,180"
             style="fill:red;stroke:purple;stroke-width:5;fill-
rule:nonzero;" />

<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>
```

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:

path 创建路径

M = moveto(M X,Y)： 将画笔移动到指定的坐标位置

L = lineto(L X,Y)： 画直线到指定的坐标位置

H = horizontal lineto(H X)：画水平线到指定的X坐标位置

V = vertical lineto(V Y)：画垂直线到指定的Y坐标位置

C = curveto(C X1,Y1,X2,Y2,ENDX,ENDY)：三次贝赛曲线

S = smooth curveto(S X2,Y2,ENDX,ENDY)

Q = quadratic Belzier curve(Q X,Y,ENDX,ENDY)：二次贝赛曲线

T = smooth quadratic Belzier curveto(T ENDX,ENDY)：映射

A = elliptical Arc(A RX,RY,XROTATION,FLAG1,FLAG2,X,Y)：弧线

Z = closepath()：关闭路径

注意：坐标轴为以(0,0)为中心，X轴水平向右，Y轴水平向下。 所有指令大小写均可。大写绝对定位，参照全局坐标系；小写相对定位，参照父容器坐标系 指令和数据间的空格可以省略 同一指令出现多次可以只用一个

A指令

允许不闭合。可以想像成是椭圆的某一段，共七个参数：

A RX,RY,XROTATION,FLAG1,FLAG2,X,Y

RX,RY指所在椭圆的半轴大小 XROTATION指椭圆的X轴与水平方向顺时针方向夹角，可以想像成一个水平的椭圆绕中心点顺时针旋转XROTATION的角度。 FLAG1只有两个值，1表示大角度弧线，0为小角度弧线。 FLAG2只有两个值，确定从起点至终点的方向，1为顺时针，0为逆时针 X,Y为终点坐标

二次贝塞尔曲线 M X0 Y0 Q X1 Y1 X Y

三次贝塞尔曲线 M X0 Y0 C X1 Y1 X2 Y2 ENDX ENDY

光滑贝塞尔曲线

T 是 Q的光滑 版本 S是C的光滑版本

g元素将多个元素组织在一起

由g元素编组在一起的可以设置样式

SVG 文本

和创建文本

```
<text x="100" y="100" fill="red" stroke="blue" text-anchor="middle"
startOffset="30%" style="font-family: 'Adobe Caslon Pro';font-size: 30px">
        <tspan>青云欢迎你</tspan>
        <tspan dx="10" dy="10">郑州欢迎你</tspan>
    </text>
```

x 和 y 属性 - 定位标准 通过 dx dy 来设置错落的文字

x、text-anchor文本中间位置 确定排列起始位置 dx, dy 属性 切线和法线方向的偏移 x, dx ,startOffset 在切线上进行偏移

运用tspan可以控制法线上的偏移，且具有向后影响性 可以将tspan的tspan值返回

textPath

超出路径将不会渲染 xlink:href 来指定 startOffset 来指定在路径上的起始位置。 标签

```
 <defs>
    <path id="p1" d="M 10 100 Q 100 50 200 100" fill="none" stroke="blue"
stroke-width="3"/>
    </defs>
    <text>
        <textPath xlink:href="#p1">
            前方的路啊,慢慢走啊
         </textPath>
    </text>
    <text dy="30">
        <textPath  startOffset="-10%" xlink:href="#p1">前方的路啊,慢慢走啊
</textPath>
    </text>
```

超链接

```
    <a xlink:href="https://www.baidu.com" xlink:title="圆形" target="_blank">
        <circle cx="100" cy="100" r="50" style="fill:grey"/>
        <text x="80" y="100" font-size="12">Or here</text>
    </a>
```

可以添加到任意的图形上

xlink:href 指定连接地址

xlink:title 指定连接提示

target 指定打开目标

渐变

linearGradient 可用来定义 SVG 的线性渐变。

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。

线性渐变可被定义为水平、垂直或角形的渐变：

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

```
 <defs>
        <linearGradient id="l1" x1="0" y1="0" x2="1" y2="1">
            <stop offset="0" stop-color="red"/>
            <stop offset="0.5" stop-color="green"/>
            <stop offset="1" stop-color="blue"/>
        </linearGradient>
    </defs>
    <rect height="100" width="200" x="50" y="50" stroke="blue"
fill="url(#l1)"></rect>
```

radialGradient 用来定义放射性渐变。

radialGradient标签的 id 属性可为渐变定义一个唯一的名称，cx、cy 和 r 属性定义外圈，而 fx 和 fy 定义内圈 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 标签来规定。offset 属性用来定义渐变的开始和结束位置。

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%）

```
<svg width="500" height="500">
<defs>
    <radialGradient id="r1" cx="0.2" cy="0.4" r="0.5" fx="0.5" fy="0.5">
        <stop offset="0" style="stop-color: red"/>
        <stop offset="0.5" style="stop-color: green"></stop>
        <stop offset="1" style="stop-color: blue"></stop>
    </radialGradient>

</defs>
<ellipse rx="100" ry="50" cx="50" cy="50" stroke="blue" fill="url(#r1)"/>
  </svg>
```

滤镜

标签必须嵌套在 标签内。 标签是 definitions 的缩写，它允许对诸如滤镜等特殊元素进行定义。 元素id属性定义一个滤镜的唯一名称 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 元素的滤镜属性用来把元素链接

到"f1"滤镜

```
<svg width="1000" height="1000">
<defs>
    <filter id="f1" x="0" y="0">
        <feGaussianBlur in="SourceGraphic" stdDeviation="30"/>
    </filter>
</defs>
<rect x="100" y="100" height="200" width="200" stroke="green"    stroke-
width="3" fill="red" filter="url(#f1)"/>
```

SVG动画

SVG动画基于 SMIL（Synchronized Multimedia Integration Language - 同步多媒体集成语言） 来实现

animate 定义动画的实时属性 持续时间、属性和属性值

attributeName 属性：动画属性的名称可以是元素直接暴露的属性，可以是CSS属性。

attributeType 属性：取值："auto" | "XML" | "CSS"

auto：默认值，自动识别（他会优先当成 CSS 处理，如果无法识别，则直接当成 XML 类别处理。如果不能确定，建议不设置 attributeType 值）

from, to, by, values 属性：

from：动画的起始值（可选，当不写的时候会取默认值）

to：指定动画的结束值

by：动画的相对变化值

values：可以是分号分隔的一个或多个值（即：一组数值。但只有一个值时，无法体现动画效果）

begin 动画开始的时间, end 属性 用分号分隔的一组值，例 beigin="1s;3s" 表示 1s 时执行一次，3s 再执行一次（如果之前动画没结束，会立即停止后续效果，从头开始）

dur 属性：动画的持续时间

让动画停留在最后一帧

fill 属性： 表示动画间隙的填充方式。取值："remove" | "freeze"。 remove：默认值。表示动画结束后直接回到动画开始状态。 freeze：冻结的意思。表示动画结束后保持动画结束时的状态。 主要注意一点：SVG fill 属性和 animate fill 属性是两个独立的属性。

动画无限循环

repeatCount 属性：

表示动画执行次数。可以是一个数字，也可以是 "indefinite" 无限循环。

另外还有一个 repeatDur 属性：表示动画重复的总时间。即：repeatDur = "1s" 那么动画执行到一半时就会暂停。

多个动画同时执行

增加一个 标签，让元素能够同时实现2个效果

为每个 标签添 id ，然后 下一个动画的 begin 值为上一个的 id.end

http://www.tuicool.com/articles/Qbma2mN

http://www.zhangxinxu.com/study/201408/svg-viewbox-explain.html

Note:

from, to, by, values 属性

如果动画的起始值与元素的默认值是一样的，from 参数可以省略。

不考虑values的前提下，to, by 两个参数至少需要有一个出现，否则动画效果没有。to 表示绝对值，by 表示相对值。拿位移距离，如果 from 是 0，to 值为 100 则表示元素从 0 移动到 100 这个位置。但是，如果 by 值是200，则表示移动到 100+200=300 这个位置。

如果 to, by 同时出现，只识别to。