02

OPEN ORIENTED

凹 凸 实 验 室

聊聊SVG 基本形状转换那些事

– – pfan

02

为什么要把SVG 基本形状转换为path呢？

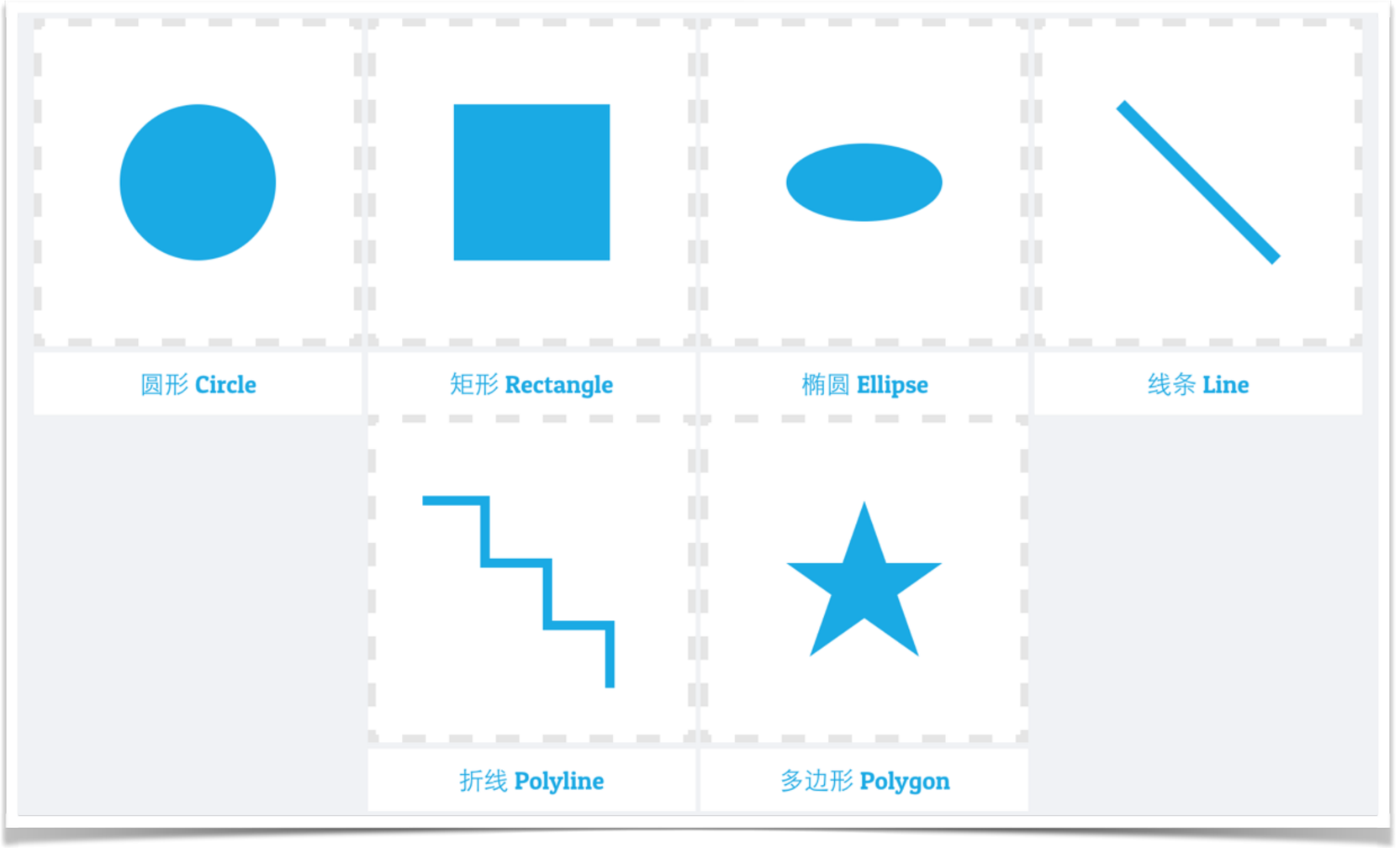
一、path路径方便于做路径动画

二、基本形状转换为path路径，可以压缩代码量

分享
目录

SVG基本元素

SVG 的提供了rect、circle、ellipse、line、polyline、polygon 六种基本形状，而 path 的功能更为强大，这六种基本形状均可由 path 实现。



SVG路径path

SVG 的 路径 <path> 功能非常强大，前面介绍的六个基本图形均可由路径实现，路径是由一些命令来控制的，每一个命令对应一个字母，并且区分大小写，大写主要表示绝对定位，小写表示相对定位。

```
// 以下两个等价
d='M 10 10 20 20'      // (10, 10) (20 20) 都是绝对坐标
d='M 10 10 L 20 20'

// 以下两个等价
d='m 10 10 20 20'      // (10, 10) 绝对坐标, (20 20) 相对坐标
d='M 10 10 l 20 20'
```

SVG路径path常见命令

command	描述
`M(m) x y`	移动到(x, y) (小写表示相对于上个坐标的位移, 下同)
`L(l) x y`	画一条直线到(x, y)
`H(h) x`	水平画一条直线到 x
`V(v) y`	竖直画一条直线到 y
`A(a) rx ry x-axis-rotation large-arc sweep x y`	画一段到(x,y)的椭圆弧. 椭圆弧的 x, y 轴半径分别为 rx,ry. 椭圆相对于 x 轴旋转 x-axis-rotation 度. large-arc=0表明弧线小于180读, large-arc=1表示弧线大于180度. sweep=0表明弧线逆时针旋转, sweep=1表明弧线顺时间旋转. 具体解释看如何绘制椭圆弧
`Q(q) cx cy x y`	从当前点画一条到(x, y)的二次贝塞尔曲线, 曲线的控制点为(cx, cy). 关于二次贝塞尔曲线请看[二次贝塞尔曲线详解](#articleHeader6
`T(t) x y`	此命令只能跟在一个 Q 命令使用, 假设 Q 命令生成曲线 s, T 命令的作用是从 s 的终点再画一条到(x y)的二次贝塞尔曲线, 曲线的控制点为 s 控制点关于 s 终点的对称点. T 命令生成的曲线会非常平滑
`C(c) cx1 cy1 cx2 cy2 x y`	从当前点画一条到(x, y)的三次贝塞尔曲线, 曲线的开始控制点和终点控制点为别为 (cx1, cy1), (cx2, cy2). 关于三次贝塞尔曲线请看三次贝塞尔曲线详解
`S(s) cx2 cy2 x y`	此命令只能跟在 C 命令后使用, 假设 C 命令生成曲线 s, S 命令的作用是再画一条到 (x, y)的三次贝塞尔曲线, 曲线的终点控制点是 (cx2, cy2), 曲线的开始控制点是 s 的终点控制点关于 s 终点的对称点.

SVG基本元素path路径转换原理

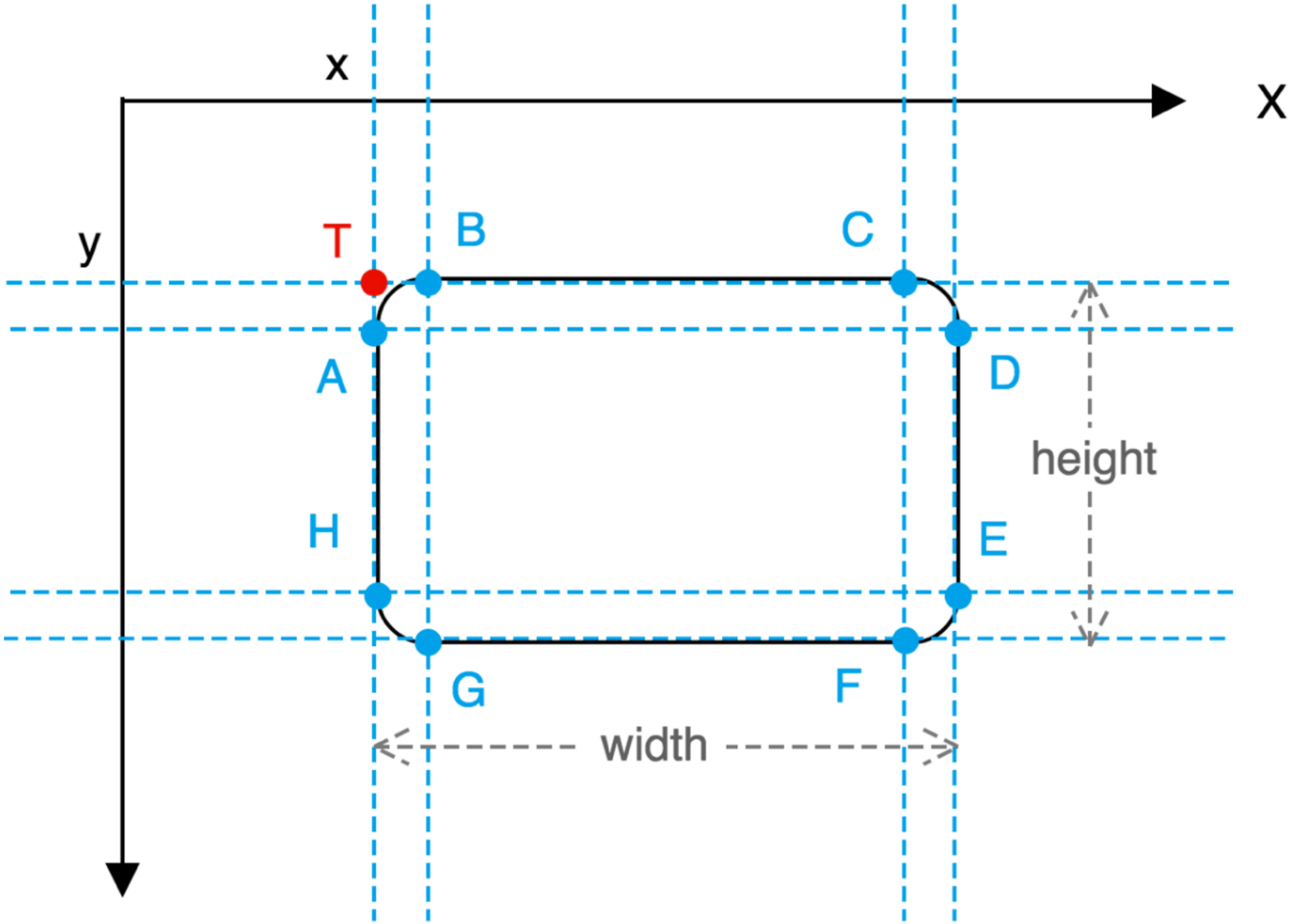
1.<rect> 矩形转换path

```
<svg with="200" height="200" viewBox="0 0 200 200" version="1.1" xmlns="http://www.w3.org/2000/svg">
  <rect x="50" y="50" width="100" height="100" rx="4" ry="2" fill="#fff"></rect>
</svg>
```

2.<circle> 矩形转换path

```
<svg width="200" height="200" viewBox="0 0 200 200">
  <circle cx="100" cy="100" r="50" fill="#fff"></circle>
</svg>
```

<rect> 矩形转换path

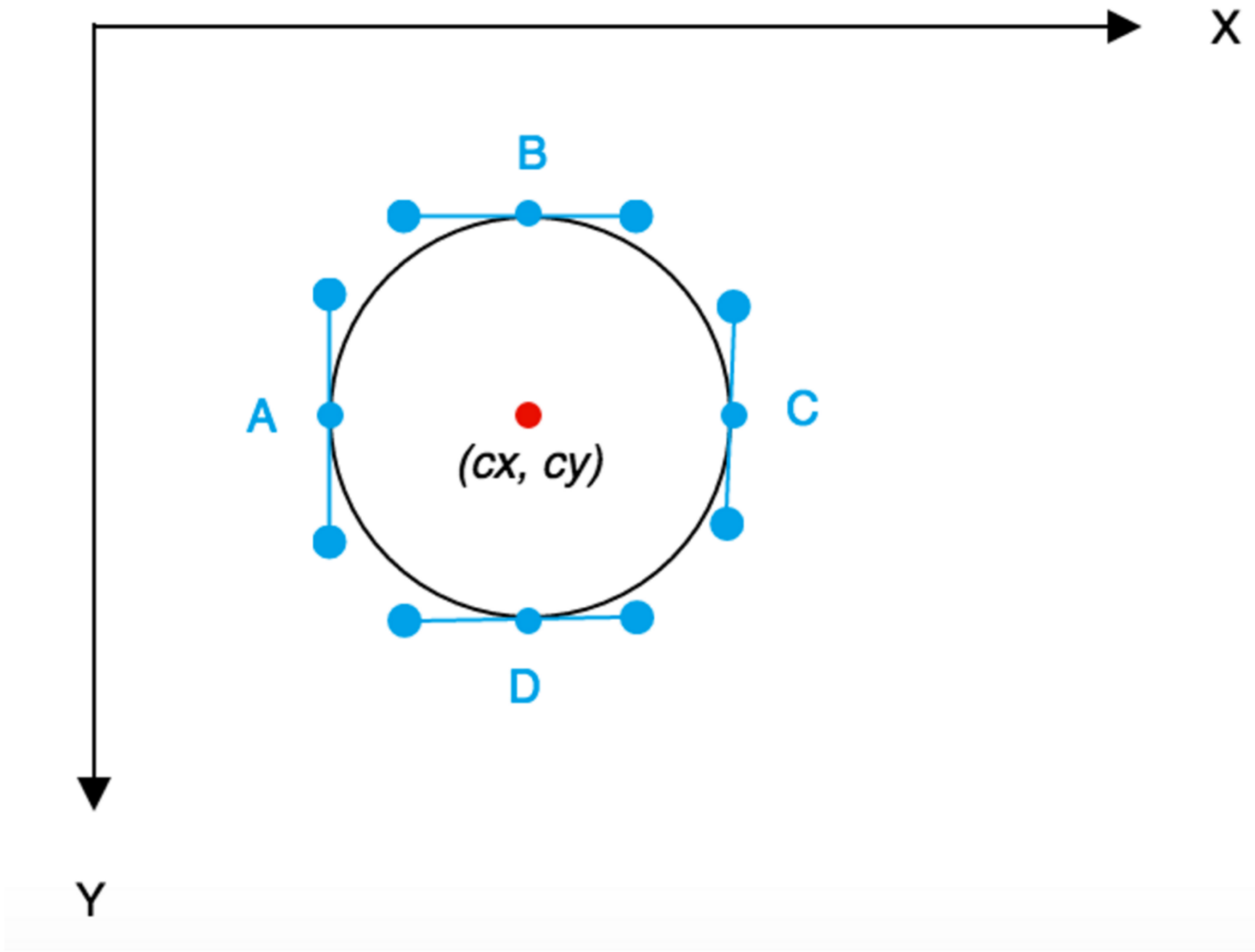


<rect> 矩形转换path


```
// 不含有rx,ry 或 rx,ry有一个值为0
path =
    'M' + x + ' ' + y +
    'h' + width +
    'v' + height +
    'h' + -width +
    'z';


// 含有rx,ry 且都不为0
path =
    'M' + x + ' ' + (y+ry) +
    'a' + rx + ' ' + ry + ' 0 0 1 ' + rx + ' ' + (-ry) +
    'h' + (width - rx - rx) +
    'a' + rx + ' ' + ry + ' 0 0 1 ' + rx + ' ' + ry +
    'v' + (height - ry -ry) +
    'a' + rx + ' ' + ry + ' 0 0 1 ' + (-rx) + ' ' + ry +
    'h' + (rx + rx -width) +
    'a' + rx + ' ' + ry + ' 0 0 1 ' + (-rx) + ' ' + (-ry) +
    'z';
```

<circle> 矩形转换path



<circle> 矩形转换path


```
var cx = node.getAttribute('cx') ,
    cy = node.getAttribute('cy') ,
    r = node.getAttribute('r') ;

var path =
    'M' + (cx-r) + ' ' + cy +
    'a' + r + ' ' + r + ' 0 1 0 ' + 2*r + ' 0' +
    'a' + r + ' ' + r + ' 0 1 0 ' + (-2*r) + ' 0' +
    'z';
```

演示分析

https://github.com/pfan123/convertpath

参考资料

- 路径
- svg图形、路径、变换总结
- 探究 dataURI 中使用 SVG 正确姿势

THANKS

FOR YOUR WATCHING

02

OPEN ORIENTED

凹 凸 实 验 室