in context

and metafun xl

Introduction

This document is about using svg, an xml based format for describing graphics and colorful font shapes in ConTEXt. It's one of the external figure formats. Where we can use MetaPost for all kind of systematic graphics, bitmap images and artistic outlines come from outside. Inclusion of svg using the methods discussed here is quite efficient and will work for many graphics, but when it doesn't you can always fall back on a conversion by Inkscape. It's work in progress anyway.

The document is made for viewing on the screen and has a bunch of examples taken from websites. We might add more in due time. The cover page has the svg logo taken from Wikipedia but with some details added. It's not a nice cover image but it will do for our purpose. Feel free to suggest additional examples.

Hans Hagen
Hasselt NL
October 2019+

The svg format

1 What it is

The Scalable Vector Graphics format (svg) showed up around the turn of this century. I remember looking into it and wondering to what extent it was a fresh development and not some kind of application format turned xml. Most elements are empty elements and data lives in attributes. What I found most puzzling is that a path definition was an attribute and not just content, especially because it can be a pretty large blob of numbers and commands. Anyway, at that time I played a bit with conversion but in the end decided to just consider it an external format for which conversion to (say) pdf by an external program was a reasonable. At some point that external program became Inkscape and ConTEXt uses that to convert svg images to pdf runtime (with caching).

In the meantime edition one turned edition two and the advance of html and css has crept features into the format, thereby not making it look better. But, because viewers support rendering svg, we now also see graphics showing up. The ones that I have to deal with are educational graphics, and when you look into the files, they can be curiously inconsistent in the way parts of graphics are made. For instance, the numbers along an axis of a mathematical graphic can be a mix of references to a font (`<text/>`), references to symbols `<symbol/>` that have paths (`<path/>`) or just paths `<path/>`. Using a tool that can spit out something structured doesn't mean that all its users will structure.

The svg format provides lines, rectangles, circles, ellipses, polylines, polygons and paths. Paths are defines as a sequence of moves, lines, cubic and quadratic curves, arcs, collected in the `d` attribute (a funny short name compared to the length of its content and the verbosity of other attribute names). They can be open or closed, and use different winding rules. Positions are absolute or relative. This all leaves a lot of room for error and confusion. When a path looks bad, it can be produced bad, or the interpretation can be bad. Interpretation can even be such that errors are catched which makes it hard to figure out what is really wrong. And as usual, bugs (and supposed catches) can become features in the end. So it might take a while before this kind of support in ConTEXt becomes stable but once it is, normally we're okay for a while. And, one nice side effect of xml is that it can't really crash processing as it's just data.

2 Color fonts

Then color fonts showed up in OpenType and svg is one of the used sub-formats in that. Again it was convenient enough to

rely on Inkscape to do the conversion to pdf blobs, but after a while I decided that a more native (built-in) support start making sense. A lot had happened since 2000, most noticeably the arrival of LuaTEX and ConTEXt MkIV followed by LuaMetaTEX and ConTEXt lmtx, so a more direct support because more feasible. A more direct support has the advantage that we don't need to call an external program and cache the results (think of Emoji fonts with thousands of glyphs in svg format). Also, direct conversion makes it possible to tweak colors and such, simply because the data goes through the ConTEXt internals as part of the typesetting process. So, as a prelude to the ConTEXt 2019 meeting a preliminary converter was made, color font support was partially redone, and afterward the converter got completed to the level needed for embedding more fancy graphics, including relabeling.

3 In practice

In the end all is about paths or glyphs, plus some optional clipping and transformations. The rendering is controlled by attributes: color, transparency, line thickness, the way lines join and end, etc. Now, in the original specification that was done only with attributes, which is a clean and robust way of doing it, but later styles and classes were introduced and we now have a whole chain to consider when resolving a to be used attribute.

- attributes explicitly set by keys to an element
- attributes set in the `style` attribute
- attributes set via one or more `class` assignments
- attributes set for the specific element
- attributes inherited from an ancestor (somewhat vague)
- redundant (nested) attributes (text styling)

Where examples are often hand codes and therefore look ok, graphics that get generated can look quite horrible: the same parameters being set with different methods, even inconsistently, to mention one. But also, graphics can be read in, tweaked and saved again which in itself generates artifacts, etc. One can of course argue that xml is not for human consumption but personally I tend to conclude that when a source file looks bad, the likelyhood is great that what it encodes looks bad too. And for instance Inkscape provides ways to inspect and tweak the xml in the editor.

4 The conversion

This brings us to the conversion. As we need pdf operators one method is to directly go from svg to pdf. There is the issue

of fonts, but as we delegate that to TEX anyway, because that is kind of an abstraction. Such a conversion is comparable with going from MetaPost to pdf. However, for practical reasons an intermediate step has been chosen: we go from svg to MetaPost first. This has the benefit that we need little code for color and transparency because MetaPost (read: MetaFun) already deals with that. We also don't need that much for text, as we deal with that in MetaPost too, and that way we can even overload and reposition for instance labels in graphics relatively easy.

Another advantage of the intermediate step is that we can combine svg graphics with MetaPost code. Of course we can already combine external graphics with MetaPost, but there is a big advantage here: the output is quite efficient. When we transform paths and pens in MetaPost, the end result is often just a path, but where we to do a direct conversion to pdf, we would either have to do calculations on our own, or we would have to use lots of transformation directives. In the end, especially because MetaPost is so fast, the indirect route pays of well (and I haven't even optimized it yet).

5 Remark

In the perspective if using TEX and MetaPost it makes sense to keep an eye on consistency. You can make quite structured svg images if you want to. When you use a graphical editor you can even consider using a normal text editor to clean up the code occasionally. The cleaner the code, the more predictable the outcome will become. Looking at the code might also give an impression of what features not to use of use differently. Of course this makes most sense in situations where there are many graphics and long-term (re)use is needed.

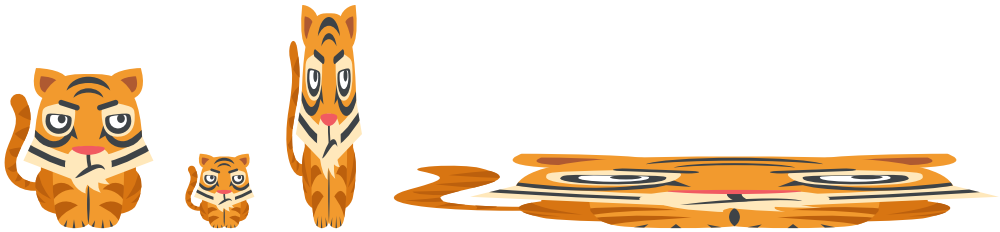
Embedding graphics

1 External figures

At least for now, the default svg inclusions is done via an external converter but you can use the internal one by specifying a conversion. The next example demonstrates that it works like any external figure:

```
\startcombination[4*1]
    {\externalfigure[mozilla-tiger.svg][conversion=mp]}                    {1}
    {\externalfigure[mozilla-tiger.svg][conversion=mp,height=1cm]}         {2}
    {\externalfigure[mozilla-tiger.svg][conversion=mp,height=3cm,width=1cm]} {3}
    {\externalfigure[mozilla-tiger.svg][conversion=mp,height=1cm,width=8cm]} {4}
\stopcombination
```

We get:



1 2 3 4

2 Internal figures

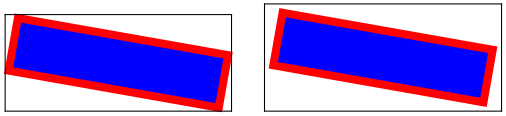
You can put some svg code in a buffer:

```
\startbuffer[svgtest]
    <svg>
        <rect
```

```
            x="0" y="0" width="80" height="20"
            fill="blue" stroke="red" stroke-width="3"
            stroke-linejoin="miter"
            transform="rotate(10)"
        />
    </svg>
\stopbuffer
```

In the future more options might be added but for now there's only an offset possible:

```
\startcombination[2*1]
    {\framed[offset=overlay]{\includesvgbuffer[svgtest]}}             {default}
    {\framed[offset=overlay]{\includesvgbuffer[svgtest][offset=2bp]}} {some offset}
\stopcombination
```



default some offset

There is a companion command `\includesvgfile` which accepts a filename and also supports offsets. Sometimes the offset is needed to prevent unwanted clipping.

3 Mixing in MetaFun

An svg image can be directly included in an MetaFun image. This makes it possible to enhance (or manipulate) such an image, as in:

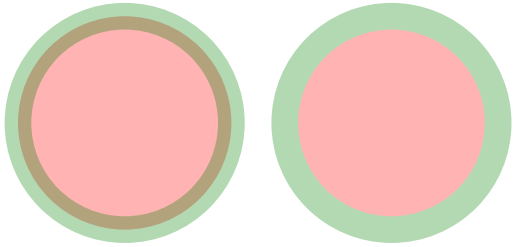
```
\startMPcode
    draw lmt_svg [
        filename = "mozilla-tiger.svg",
        origin   = true,
```

```
    ] rotated 45 slanted .75 ysized 2cm ;

    setbounds currentpicture to
        boundingbox currentpicture
        enlarged 1mm ;

    addbackground
        withcolor "darkgray" ;
\stopMPcode
```

An svg image included this way becomes a regular MetaPost picture, so a collection of paths. Because MetaPost on the average produces rather compact output the svg image normally also is efficiently embedded. You don't need to worry about loosing quality, because MetaPost is quite accurate and we use so called 'double' number mode anyway.

Another trick is to inline the code:

It doesn't really make sense as MetaPost code is just as simple but it looks cool:

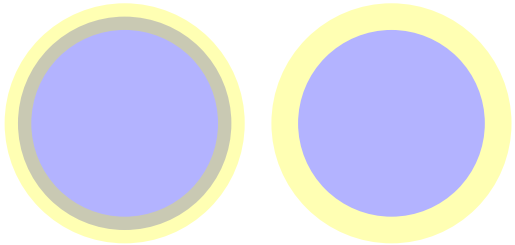


And actually it's less code (which internally of course expands to more):

```
\startMPcode
    pickup pencircle scaled 10;
```

```
    path p ; p := fullcircle scaled 80 ;
    fill p shifted (50,50) withcolor blue
        withtransparency(1,0.3) ;
    draw p shifted (50,50) withcolor yellow
        withtransparency(1,0.3) ;
    draw image (
        fill p shifted (150,50) withcolor blue ;
        draw p shifted (150,50) withcolor yellow ;
        setgroup currentpicture to boundingbox currentpicture
            withtransparency(1,0.3) ;
    ) ;
\stopMPcode
```

It’s all a matter of taste. Watch the grouping trick!



4 Fonts

This is still experimental.

5 Labels

This is still experimental.

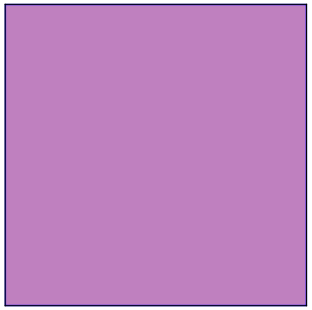
6 Tracing

This is still experimental.

Mozilla test snippets

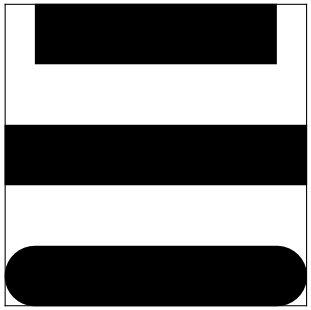
The Mozilla documentation pages for svg are pretty good and contain snippets that can be used for testing. More examples might be added in due time.

1 Snippet 1



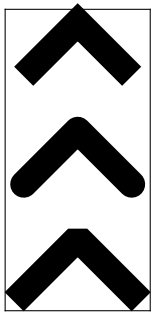
```svg
<svg width="160" height="140" xmlns="http://www.w3.org/2000/svg" version="1.1">
    <rect x="10" y="10" width="100" height="100" stroke="blue" fill="purple" fill-opacity="0.5" stroke-opacity="0.8"/>
</svg>
```

2 Snippet 2



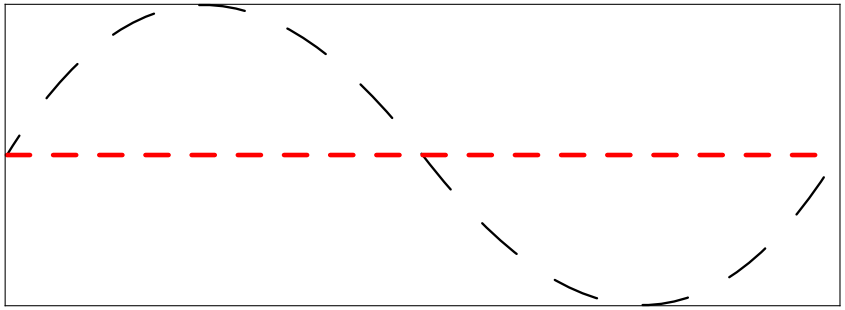
```svg
<svg width="160" height="140" xmlns="http://www.w3.org/2000/svg" version="1.1">
  <line x1="40" x2="120" y1="20"  y2="20"  stroke="black" stroke-width="20" stroke-linecap="butt"/>
  <line x1="40" x2="120" y1="60"  y2="60"  stroke="black" stroke-width="20" stroke-linecap="square"/>
  <line x1="40" x2="120" y1="100" y2="100" stroke="black" stroke-width="20" stroke-linecap="round"/>
</svg>
```

3 Snippet 3



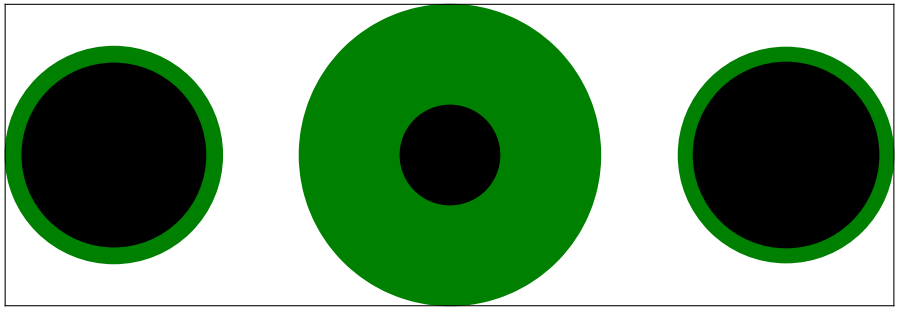
```
<svg width="160" height="280" xmlns="http://www.w3.org/2000/svg" version="1.1">
  <polyline points="40  60 80  20 120  60" stroke="black" stroke-width="20" stroke-linecap="butt"   fill="none" stroke-linejoin
    ="miter"/>
  <polyline points="40 140 80 100 120 140" stroke="black" stroke-width="20" stroke-linecap="round"  fill="none" stroke-linejoin
    ="round"/>
  <polyline points="40 220 80 180 120 220" stroke="black" stroke-width="20" stroke-linecap="square" fill="none" stroke-linejoin
    ="bevel"/>
</svg>
```

4 Snippet 4



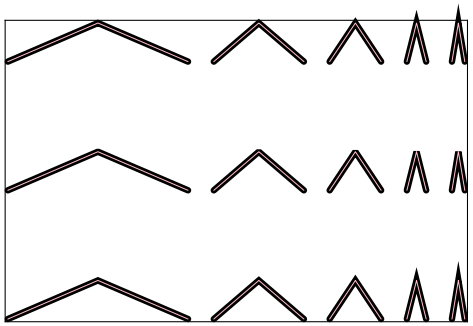
```
<svg width="200" height="150" xmlns="http://www.w3.org/2000/svg" version="1.1">
    <path d="M 10 75 Q 50 10 100 75 T 190 75" stroke="black" stroke-linecap="round"                  stroke-dasharray="5,10,5"
      fill="none"/>
    <path d="M 10 75 L 190 75"               stroke="red"   stroke-linecap="round" stroke-width="1" stroke-dasharray="5,5"
      fill="none"/>
</svg>
```

5 Snippet 5



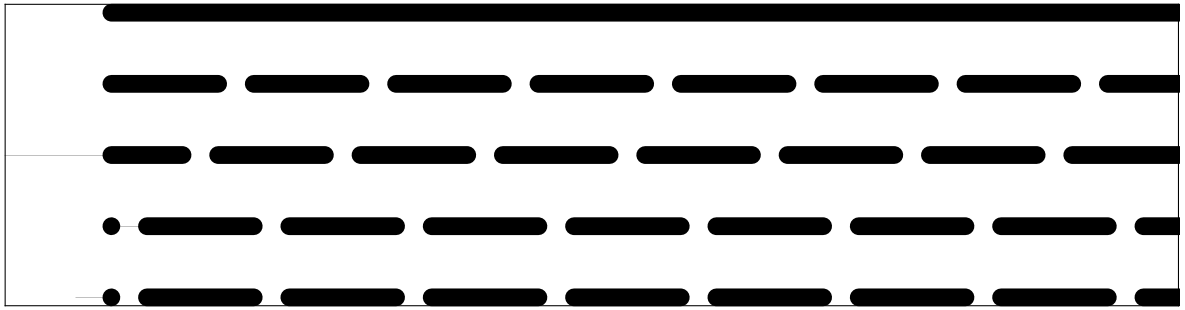
```
<svg viewBox="0 0 30 10" xmlns="http://www.w3.org/2000/svg">
  <circle cx="5"  cy="5" r="3" stroke="green" />
  <circle cx="15" cy="5" r="3" stroke="green" stroke-width="3" />
  <circle cx="25" cy="5" r="3" stroke="green" stroke-width="2%" />
</svg>
```

6 Snippet 6



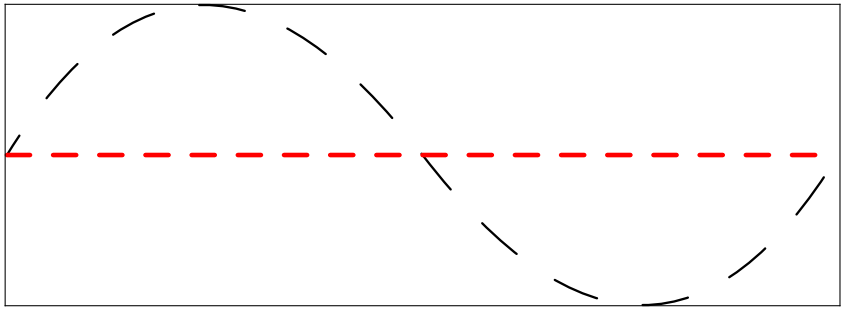
```
<svg viewBox="0 0 38 30" xmlns="http://www.w3.org/2000/svg">
  <path stroke="black" fill="none" stroke-linejoin="miter" id="p1"
        d="M1,9 l7   ,-3 l7   ,3
           m2,0 l3.5 ,-3 l3.5 ,3
           m2,0 l2   ,-3 l2   ,3
           m2,0 l0.75,-3 l0.75,3
           m2,0 l0.5 ,-3 l0.5 ,3" />
  <path stroke="black" fill="none" stroke-linejoin="miter" stroke-miterlimit="1" id="p2"
        d="M1,19 l7   ,-3 l7   ,3
           m2, 0 l3.5 ,-3 l3.5 ,3
           m2, 0 l2   ,-3 l2   ,3
           m2, 0 l0.75,-3 l0.75,3
           m2, 0 l0.5 ,-3 l0.5 ,3" />
  <path stroke="black" fill="none" stroke-linejoin="miter" stroke-miterlimit="8" id="p3"
        d="M1,29 l7   ,-3 l7   ,3
           m2, 0 l3.5 ,-3 l3.5 ,3
           m2, 0 l2   ,-3 l2   ,3
           m2, 0 l0.75,-3 l0.75,3
           m2, 0 l0.5 ,-3 l0.5 ,3" />
  <path stroke="pink" fill="none" stroke-width="0.05"
        d="M1, 9 l7,-3 l7,3 m2,0 l3.5,-3 l3.5,3 m2,0 l2,-3 l2,3 m2,0 l0.75,-3 l0.75,3 m2,0 l0.5,-3 l0.5,3
           M1,19 l7,-3 l7,3 m2,0 l3.5,-3 l3.5,3 m2,0 l2,-3 l2,3 m2,0 l0.75,-3 l0.75,3 m2,0 l0.5,-3 l0.5,3
           M1,29 l7,-3 l7,3 m2,0 l3.5,-3 l3.5,3 m2,0 l2,-3 l2,3 m2,0 l0.75,-3 l0.75,3 m2,0 l0.5,-3 l0.5,3" />
</svg>
```

7 Snippet 7



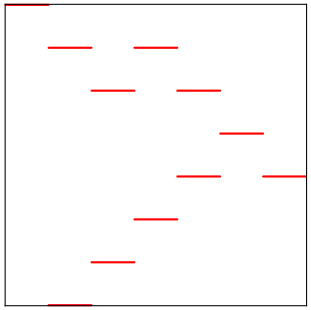
```
<svg viewBox="-3 0 33 10" xmlns="http://www.w3.org/2000/svg">
  <line x1="0" y1="1" x2="30" y2="1" stroke="black" />
  <line x1="0" y1="3" x2="30" y2="3" stroke="black" stroke-dasharray="3 1" />
  <line x1="0" y1="5" x2="30" y2="5" stroke="black" stroke-dasharray="3 1" stroke-dashoffset="3" />
  <line x1="0" y1="7" x2="30" y2="7" stroke="black" stroke-dasharray="3 1" stroke-dashoffset="-3" />
  <line x1="0" y1="9" x2="30" y2="9" stroke="black" stroke-dasharray="3 1" stroke-dashoffset="1" />
  <path d="M0,5 h-3 M0,7 h3 M0,9 h-1" stroke="rgba(255,0,0,.5)" />
</svg>
```

8 Snippet 8



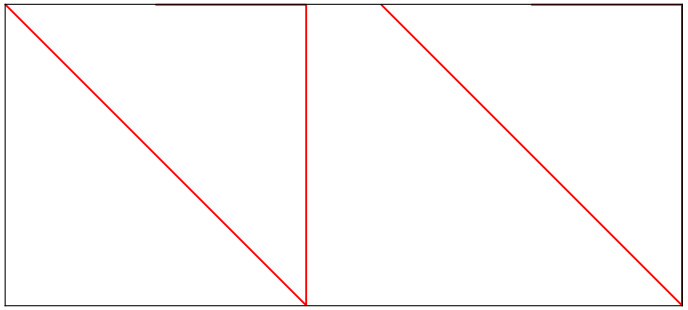
```
<svg width="200" height="150" xmlns="http://www.w3.org/2000/svg" version="1.1">
  <path d="M 10 75 Q 50 10 100 75 T 190 75" stroke="black" stroke-linecap="round" stroke-dasharray="5,10,5" fill="none"/>
  <path d="M 10 75 L 190 75"                stroke="red"   stroke-linecap="round" stroke-dasharray="5,5"    fill="none"
    stroke-width="1"/>
</svg>
```

9 Snippet 9



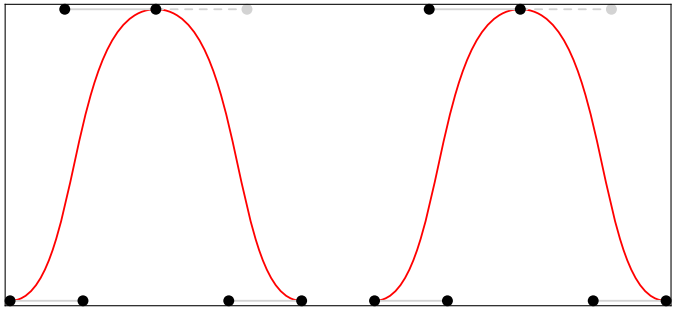
```
<svg viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
  <path fill="none" stroke="red"
    d="M 10,10 h 10
       m  0,10 h 10
       m  0,10 h 10
       M 40,20 h 10
       m  0,10 h 10
       m  0,10 h 10
       m  0,10 h 10
       M 50,50 h 10
       m-20,10 h 10
       m-20,10 h 10
       m-20,10 h 10" />
</svg>
```

10 Snippet 10



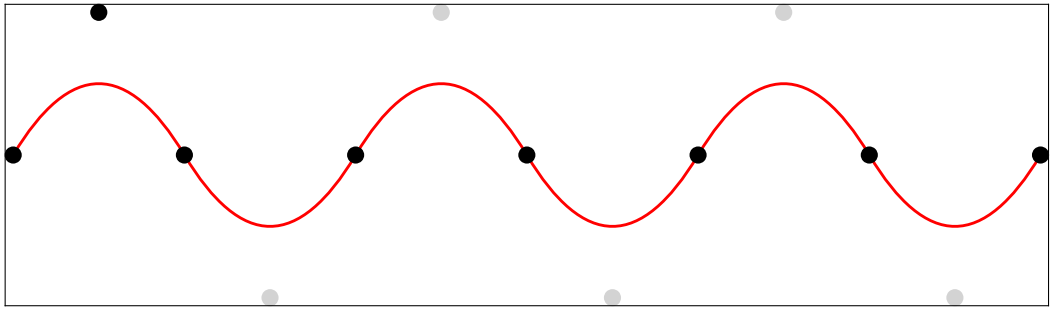
```
<svg viewBox="0 0 200 100" xmlns="http://www.w3.org/2000/svg">
  <path fill="none" stroke="red" d="M 10,10 L 90,90 V 10 H 50" />
  <path fill="none" stroke="red" d="M 110,10 l 80,80 v -80 h -40" />
</svg>
```

11 Snippet 11



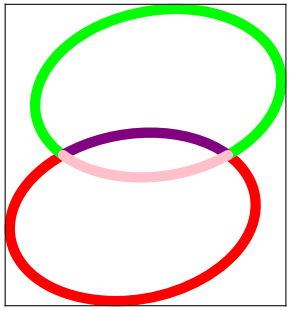
```
<svg viewBox="0 0 200 100" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink">
  <path fill="none" stroke="red" d="M 10,90 C 30,90 25,10 50,10 S 70,90 90,90" />
  <path fill="none" stroke="red" d="M 110,90 c 20,0 15,-80 40,-80 s 20,80 40,80" />
  <g id="ControlPoints">
    <line x1="10" y1="90" x2="30" y2="90" stroke="lightgrey" />
    <circle cx="30" cy="90" r="1.5"/>
    <line x1="50" y1="10" x2="25" y2="10" stroke="lightgrey" />
    <circle cx="25" cy="10" r="1.5"/>
    <line x1="50" y1="10" x2="75" y2="10" stroke="lightgrey" stroke-dasharray="2" />
    <circle cx="75" cy="10" r="1.5" fill="lightgrey"/>
    <line x1="90" y1="90" x2="70" y2="90" stroke="lightgrey" />
    <circle cx="70" cy="90" r="1.5"/>
    <circle cx="10" cy="90" r="1.5"/>
    <circle cx="50" cy="10" r="1.5"/>
    <circle cx="90" cy="90" r="1.5"/>
  </g>
  <use xlink:href="#ControlPoints" x="100" />
</svg>
```

12 Snippet 12



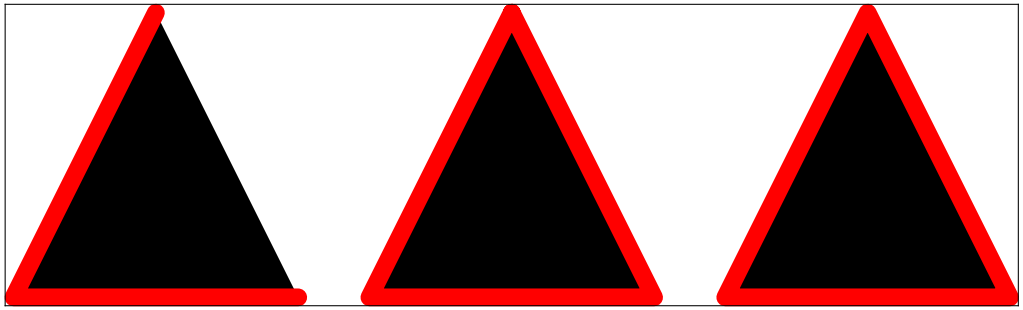
```
<svg viewBox="0 0 200 100" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink">
  <path fill="none" stroke="red" d="M 10,50 Q 25,25 40,50 t 30,0 30,0 30,0 30,0 30,0" />
  <g>
    <polyline points="10,50 25,25 40,50" stroke="rgba(0,0,0,.2)" fill="none" />
    <circle cx="25" cy="25" r="1.5" />
    <circle cx="10" cy="50" r="1.5"/>
    <circle cx="40" cy="50" r="1.5"/>
    <g id="SmoothQuadraticDown">
      <polyline points="40,50 55,75 70,50" stroke="rgba(0,0,0,.2)" stroke-dasharray="2" fill="none" />
      <circle cx="55" cy="75" r="1.5" fill="lightgrey" />
      <circle cx="70" cy="50" r="1.5" />
    </g>
    <g id="SmoothQuadraticUp">
      <polyline points="70,50 85,25 100,50" stroke="rgba(0,0,0,.2)" stroke-dasharray="2" fill="none" />
      <circle cx="85" cy="25" r="1.5" fill="lightgrey" />
      <circle cx="100" cy="50" r="1.5" />
    </g>
    <use xlink:href="#SmoothQuadraticDown" x="60" />
    <use xlink:href="#SmoothQuadraticUp"   x="60" />
    <use xlink:href="#SmoothQuadraticDown" x="120" />
  </g>
</svg>
```

13 Snippet 13



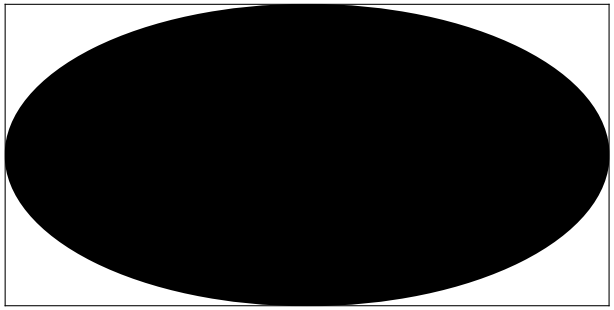
```
<svg viewBox="0 0 20 20" xmlns="http://www.w3.org/2000/svg">
  <path fill="none" stroke="red"    d="M 6,10 A 6 4 10 1 0 14,10" />
  <path fill="none" stroke="lime"   d="M 6,10 A 6 4 10 1 1 14,10" />
  <path fill="none" stroke="purple" d="M 6,10 A 6 4 10 0 1 14,10" />
  <path fill="none" stroke="pink"   d="M 6,10 A 6 4 10 0 0 14,10" />
</svg>
```

14 Snippet 14



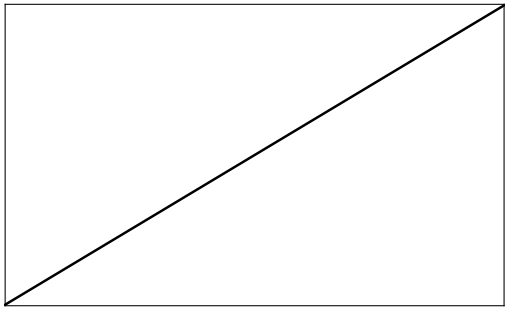
```
<svg viewBox="0 -1 30 11" xmlns="http://www.w3.org/2000/svg">
  <path stroke="red" d="M  5,1 l -4,8 8,0" />
  <path stroke="red" d="M 15,1 l -4,8 8,0 -4,-8" />
  <path stroke="red" d="M 25,1 l -4,8 8,0 z" />
</svg>
```

15 Snippet 15



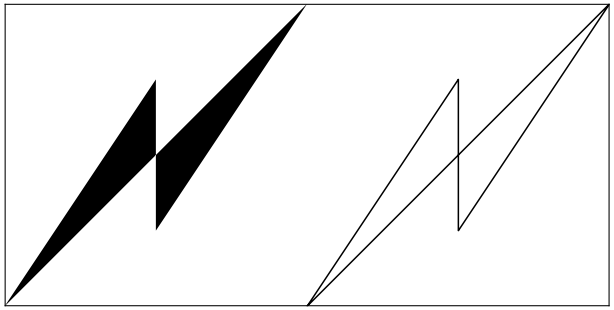
```svg
<svg viewBox="0 0 200 100" xmlns="http://www.w3.org/2000/svg">
  <ellipse cx="100" cy="50" rx="100" ry="50" />
</svg>
```

16 Snippet 16



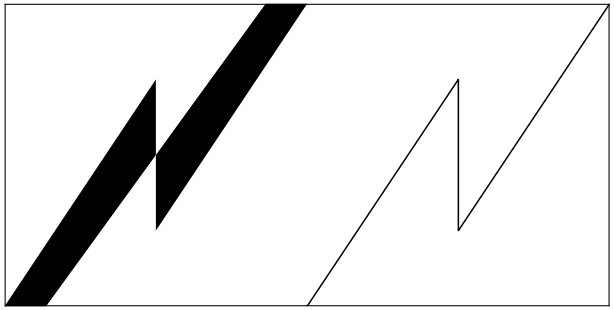
```svg
<svg viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
    <line x1="0" y1="80" x2="100" y2="20" stroke="black" />
</svg>
```

17 Snippet 17



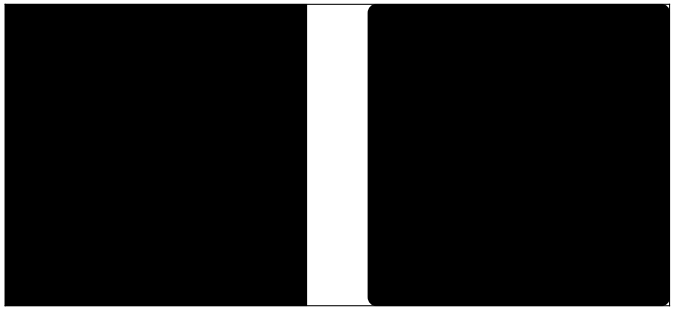
```
<svg viewBox="0 0 200 100" xmlns="http://www.w3.org/2000/svg">
    <polygon points="0,100 50,25 50,75 100,0" />
    <polygon points="100,100 150,25 150,75 200,0" fill="none" stroke="black" />
</svg>
```

18 Snippet 18



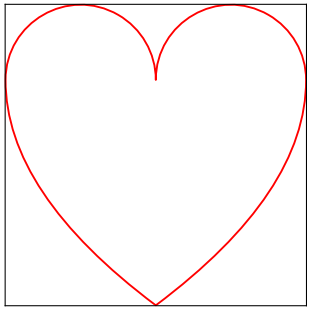
```svg
<svg viewBox="0 0 200 100" xmlns="http://www.w3.org/2000/svg">
    <polyline points="0,100 50,25 50,75 100,0" />
    <polyline points="100,100 150,25 150,75 200,0" fill="none" stroke="black" />
</svg>
```

19 Snippet 19



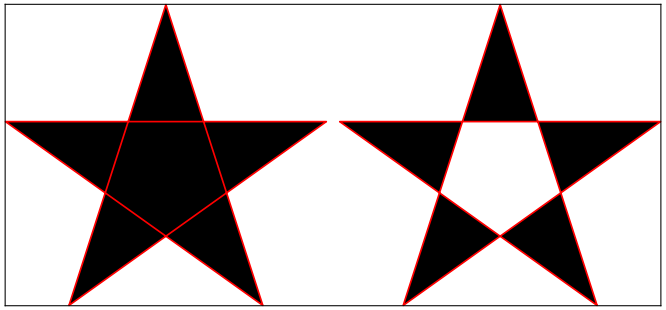
```svg
<svg viewBox="0 0 220 100" xmlns="http://www.w3.org/2000/svg">
  <rect width="100" height="100" />
  <rect x="120" width="100" height="100" rx="15" />
</svg>
```

20 Snippet 20



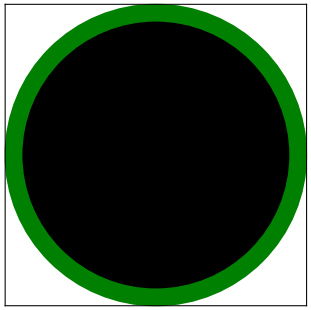
```
<svg viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
  <path fill="none" stroke="red"
    d="M 10,30
       A 20,20 0,0,1 50,30
       A 20,20 0,0,1 90,30
       Q 90,60 50,90
       Q 10,60 10,30 z" />
</svg>
```

21 Snippet 21



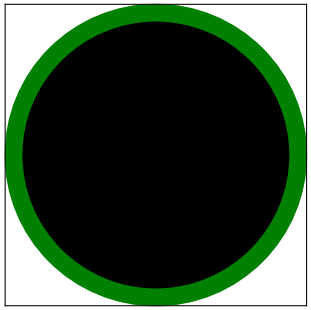
```
<svg viewBox="-10 -10 220 120" xmlns="http://www.w3.org/2000/svg">
  <polygon fill-rule="nonzero" stroke="red" points="50,0 21,90 98,35 2,35 79,90"/>
  <polygon fill-rule="evenodd" stroke="red" points="150,0 121,90 198,35 102,35 179,90"/>
</svg>
```

22 Snippet 22



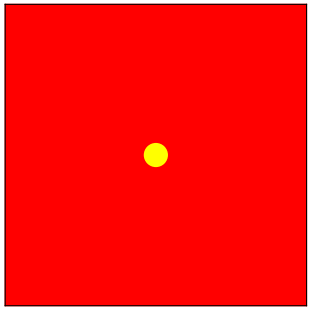
```
<svg x="0" width="10" height="10" clip="auto">
    <circle cx="5" cy="5" r="4" stroke="green" />
</svg>
```

23 Snippet 23



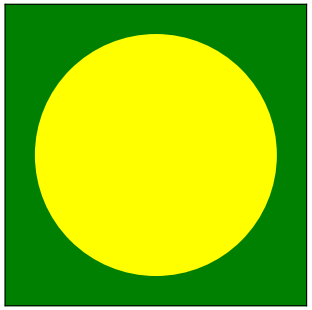
```
<svg x="10" width="10" height="10" clip="rect(1, 9, 8, 2)">
    <circle cx="5" cy="5" r="4" stroke="green" />
  </svg>
</svg>
```

24 Snippet 24



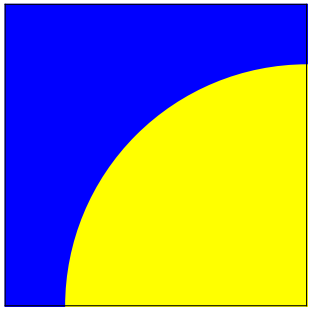
```
<svg viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
  <rect x="0" y="0" width="100%" height="100%" fill="red"/>
  <circle cx="50%" cy="50%" r="4" fill="yellow"/>
</svg>
```

25 Snippet 25



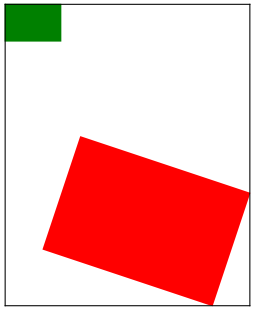
```
<svg viewBox="0 0 10 10" xmlns="http://www.w3.org/2000/svg">
  <rect x="0" y="0" width="100%" height="100%" fill="green"/>
  <circle cx="50%" cy="50%" r="4" fill="yellow"/>
</svg>
```

26 Snippet 26



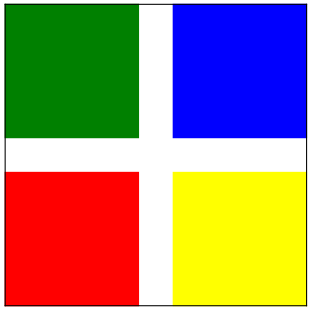
```
<svg viewBox="-5 -5 10 10" xmlns="http://www.w3.org/2000/svg">
  <rect x="0" y="0" width="100%" height="100%" fill="blue"/>
  <circle cx="50%" cy="50%" r="4" fill="yellow"/>
</svg>
```

27 Snippet 27



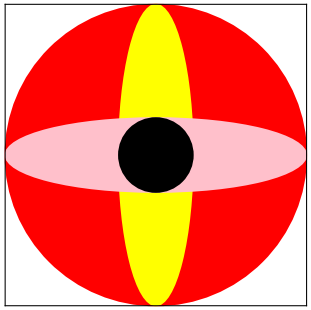
```svg
<svg viewBox="0 0 200 200" xmlns="http://www.w3.org/2000/svg">
  <rect x="10" y="10" width="30" height="20" fill="green" />
  <rect x="10" y="10" width="30" height="20" fill="red" transform="matrix(3 1 -1 3 30 40)" />
</svg>
```

28 Snippet 28



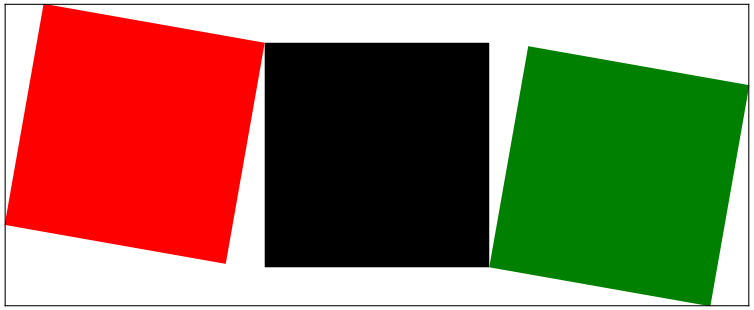
```svg
<svg viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
  <rect x="5" y="5" width="40" height="40" fill="green" />
  <rect x="5" y="5" width="40" height="40" fill="blue" transform="translate(50)" />
  <rect x="5" y="5" width="40" height="40" fill="red" transform="translate(0 50)" />
  <rect x="5" y="5" width="40" height="40" fill="yellow" transform="translate(50,50)" />
</svg>
```

29 Snippet 29



```svg
<svg viewBox="-50 -50 100 100" xmlns="http://www.w3.org/2000/svg">
  <circle cx="0" cy="0" r="10" fill="red" transform="scale(4)" />
  <circle cx="0" cy="0" r="10" fill="yellow" transform="scale(1,4)" />
  <circle cx="0" cy="0" r="10" fill="pink" transform="scale(4,1)" />
  <circle cx="0" cy="0" r="10" fill="black" />
</svg>
```

30 Snippet 30



```
<svg viewBox="-12 -2 34 14" xmlns="http://www.w3.org/2000/svg">
  <rect x="0" y="0" width="10" height="10" />
  <rect x="0" y="0" width="10" height="10" fill="red" transform="rotate(100)" />
  <rect x="0" y="0" width="10" height="10" fill="green" transform="rotate(100,10,10)" />
</svg>
```

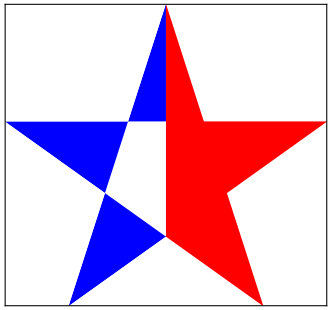
31 Snippet 31

```
<svg viewBox="-5 -5 10 10" xmlns="http://www.w3.org/2000/svg">
  <rect x="-3" y="-3" width="6" height="6" />
  <rect x="-3" y="-3" width="6" height="6" fill="red" transform="skewX(30)" />
</svg>
```

32 Snippet 32

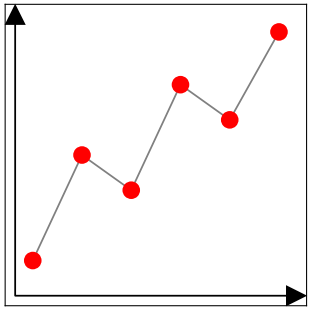
```svg
<svg viewBox="-5 -5 10 10" xmlns="http://www.w3.org/2000/svg">
  <rect x="-3" y="-3" width="6" height="6" />
  <rect x="-3" y="-3" width="6" height="6" fill="red" transform="skewY(30)" />
</svg>
```

33 Snippet 33



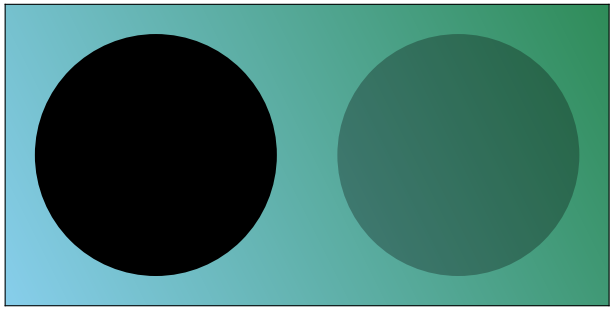
```
<svg width="100" viewBox="0 0 100 90" xmlns="http://www.w3.org/2000/svg" version="1.1">
  <defs>
    <path d="M50,0 21,90 98,35 2,35 79,90z" id="star" />
  </defs>
  <clipPath id="emptyStar">
    <use xlink:href="#star" clip-rule="evenodd" />
  </clipPath>
  <rect clip-path="url(#emptyStar)" width="50" height="90" fill="blue" />
  <clipPath id="filledStar">
    <use xlink:href="#star" clip-rule="nonzero" />
  </clipPath>
  <rect clip-path="url(#filledStar)" width="50" height="90" x="50" fill="red" />
</svg>
```

34 Snippet 34



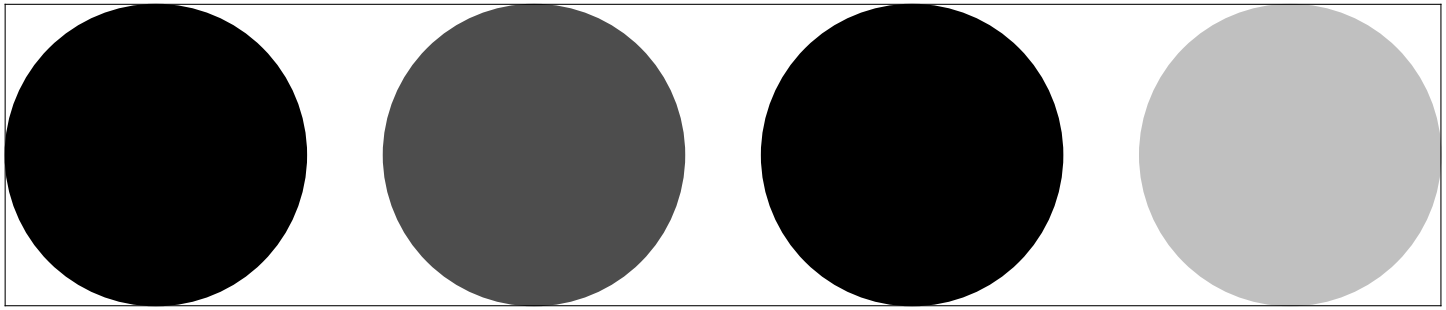
```
<svg viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
  <defs>
    <marker id="arrow" viewBox="0 0 10 10" refX="5" refY="5" markerWidth="6" markerHeight="6" orient="auto-start-reverse">
      <path d="M 0 0 L 10 5 L 0 10 z" />
    </marker>
    <marker id="dot" viewBox="0 0 10 10" refX="5" refY="5" markerWidth="5" markerHeight="5">
      <circle cx="5" cy="5" r="5" fill="red" />
    </marker>
  </defs>
  <polyline points="10,10 10,90 90,90" fill="none" stroke="black" marker-start="url(#arrow)" marker-end="url(#arrow)"  />
  <polyline points="15,80 29,50 43,60 57,30 71,40 85,15" fill="none" stroke="grey" marker-start="url(#dot)" marker-mid="
    url(#dot)"  marker-end="url(#dot)" />
</svg>
```

35 Snippet 35



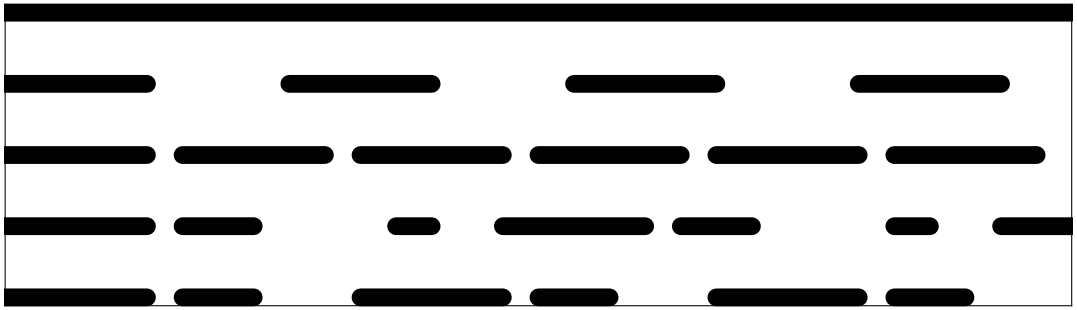
```svg
<svg viewBox="0 0 200 100" xmlns="http://www.w3.org/2000/svg">
  <defs>
    <linearGradient id="gradient" x1="0%" y1="0%" x2="0" y2="100%">
      <stop offset="0%" style="stop-color:skyblue;" />
      <stop offset="100%" style="stop-color:seagreen;" />
    </linearGradient>
  </defs>
  <rect x="0" y="0" width="100%" height="100%" fill="url(#gradient)" />
  <circle cx="50" cy="50" r="40" fill="black" />
  <circle cx="150" cy="50" r="40" fill="black" opacity="0.3" />
</svg>
```

36 Snippet 36



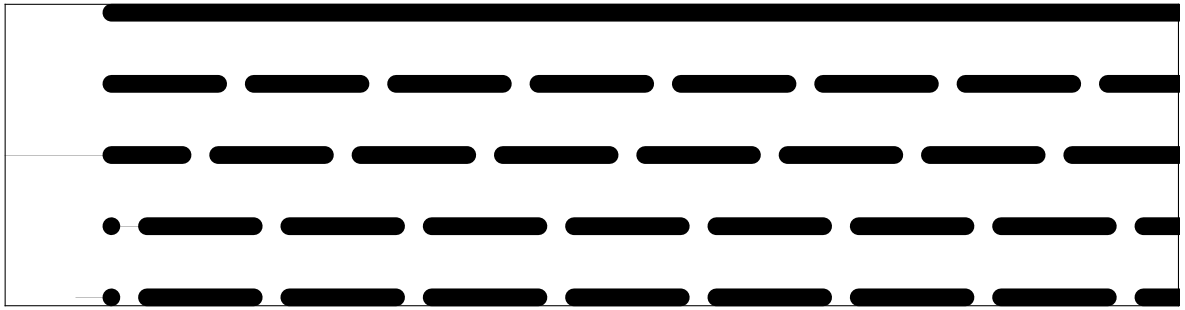
```
<svg viewBox="0 0 400 100" xmlns="http://www.w3.org/2000/svg">
  <circle cx="50" cy="50" r="40" />
  <circle cx="150" cy="50" r="40" fill-opacity="0.7" />
  <circle cx="250" cy="50" r="40" fill-opacity="50%" />
  <circle cx="350" cy="50" r="40" style="fill-opacity: .25;" />
</svg>
```

37 Snippet 37



```
<svg viewBox="0 0 30 10" xmlns="http://www.w3.org/2000/svg">
  <line x1="0" y1="1" x2="30" y2="1" stroke="black" />
  <line x1="0" y1="3" x2="30" y2="3" stroke="black" stroke-dasharray="4" />
  <line x1="0" y1="5" x2="30" y2="5" stroke="black" stroke-dasharray="4 1" />
  <line x1="0" y1="7" x2="30" y2="7" stroke="black" stroke-dasharray="4 1 2" />
  <line x1="0" y1="9" x2="30" y2="9" stroke="black" stroke-dasharray="4 1 2 3" />
</svg>
```

38 Snippet 38



```
<svg viewBox="-3 0 33 10" xmlns="http://www.w3.org/2000/svg">
  <line x1="0" y1="1" x2="30" y2="1" stroke="black" />
  <line x1="0" y1="3" x2="30" y2="3" stroke="black" stroke-dasharray="3 1" />
  <line x1="0" y1="5" x2="30" y2="5" stroke="black" stroke-dasharray="3 1" stroke-dashoffset="3" />
  <line x1="0" y1="7" x2="30" y2="7" stroke="black" stroke-dasharray="3 1" stroke-dashoffset="-3" />
  <line x1="0" y1="9" x2="30" y2="9" stroke="black" stroke-dasharray="3 1" stroke-dashoffset="1" />
  <path d="M0,5 h-3 M0,7 h3 M0,9 h-1" stroke="rgba(255,0,0,.5)" />
</svg>
```

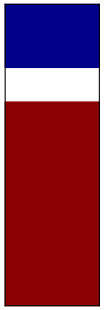
Microsoft test snippets

These snippets come from the Microsoft typography pages that discuss OpenType and svg. Because these are actually examples of glyphs, we need to set some defaults:

```
   x       0
   y    1000
 width  1000
height  1000
```

in order to get the right placement. This has to do with the fact that the vertical svg coordinates go in the other direction compared to MetaPost and pdf.

1 Snippet 1

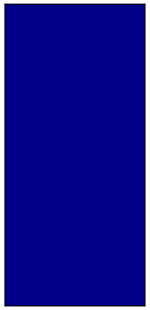


```svg
<svg id="glyph7" version="1.1" xmlns="http://www.w3.org/2000/svg">
    <rect x="100" y="-430" width="200" height="430" fill="darkred" />
    <rect x="100" y="-635" width="200" height="135" fill="darkblue" />
</svg>
```

2 Snippet 2

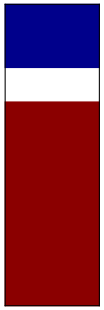
```svg
<svg id="glyph7" version="1.1" xmlns="http://www.w3.org/2000/svg" viewBox="0 1000 1000 1000">
    <rect x="100" y="570" width="200" height="430" fill="darkgreen" />
    <rect x="100" y="365" width="200" height="135" fill="darkblue" />
</svg>
```

3 Snippet 3



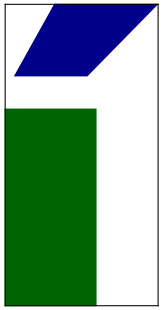
```svg
<svg version="1.1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink">
    <defs>
        <g id="i-base">
            <rect x="100" y="570" width="200" height="430" fill="darkblue" />
        </g>
    </defs>
    <g id="glyph2" transform="translate(0,-1000)">
        <use xlink:href="#i-base" />
    </g>
</svg>
```

4 Snippet 4



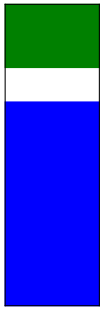
```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink">
    <defs>
        <g id="i-base">
            <rect x="100" y="570" width="200" height="430" fill="darkred" />
        </g>
    </defs>
    <g id="glyph13" transform="translate(0,-1000)">
        <use xlink:href="#i-base" />
        <rect x="100" y="365" width="200" height="135" fill="darkblue" />
    </g>
</svg>
```

5 Snippet 5



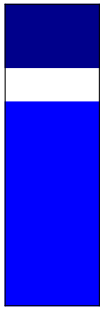
```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink">
    <defs>
        <g id="i-base">
            <rect x="100" y="570" width="200" height="430" fill="darkgreen" />
        </g>
    </defs>
    <g id="glyph14" transform="translate(0,-1000)">
        <use xlink:href="#i-base" />
        <polygon fill="darkblue" points="120,500 280,500 435,342 208,342"/>
    </g>
</svg>
```

6 Snippet 6



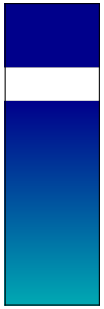
```
<svg id="glyph7" version="1.1" xmlns="http://www.w3.org/2000/svg" viewBox="0 1000 1000 1000">
    <defs>
        <linearGradient id="grad" x1="0%" y1="0%" x2="0%" y2="100%">
            <stop offset="0%" stop-color="darkblue" stop-opacity="1" />
            <stop offset="100%" stop-color="#00aab3" stop-opacity="1" />
        </linearGradient>
    </defs>
    <rect x="100" y="570" width="200" height="430" fill="blue" xfill="url(#grad)" />
    <rect x="100" y="365" width="200" height="135" fill="green" xfill="currentColor" />
</svg>
```

7 Snippet 7



```
<svg id="glyph7" version="1.1" xmlns="http://www.w3.org/2000/svg" viewBox="0 1000 1000 1000">
    <defs>
        <linearGradient id="grad" x1="0%" y1="0%" x2="0%" y2="100%">
            <stop offset="0%" stop-color="var(--color0,darkblue)" stop-opacity="1" />
            <stop offset="100%" stop-color="var(--color1,#00aab3)" stop-opacity="1" />
        </linearGradient>
    </defs>
    <rect x="100" y="570" width="200" height="430" fill="blue" xfill="url(#grad)" />
    <rect x="100" y="365" width="200" height="135" fill="darkblue" />
</svg>
```

8 Snippet 8



```
<svg id="glyph2" version="1.1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" viewBox="0 1000
  1000 1000">
    <image x="100" y="365" width="200" height="635"
    xlink:href="
        iVBORw0KGgoAAAANSUhEUgAAAMgAAAJ7CAYAAACmmd5sAAAFZklEQVR42u3XsQ3D
        MBAEQUpw9ypahrMPGGwiwcFMCQQW9zzWuu4FbJ2eAAQCAgGBgEBAICAQEAgIBAQC
        CAQEAgIBgYBAQCAgEBAICAQEAggEBAICAYGAQEAgIBAQCAgEEAgIBAQCAgGBgEBA
        ICAQEAgIBBAICAQEAgIBgYBAQCAgEBAIIBAQCAgEBAICAYGAQEAgIBAQCCAQEAgI
        BAQCAgGBgEBAICAQQCAgEBAICAQEAgIBgYBAQCAgEEAgIBAQCAgEBAICAYGAQEAg
        IBBPAAIBgYBAQCAgEBAICAQEAgIBBAICAYGAQEAgIBAQCAgEBAICAQQCAgGBgEBA
        ICAQEAgIBAQCCAQEAgIBgYBAQCAgEBAICAQEAggEBAICAYGAQEAgIBAQCAgEAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAA4DHHWtftGWDv80sE2Ds9AQgEBAL+IPBuIAoBJxYIBAQCPukgEHBigUBAIOAP
        AlgQiAtiQsCCgEDAJx0sCFgQsCAgEHBigQUB5oKYELAgIBDwSQcLAhYELAgIBJxY
        YEEACwItEIWAEwucWGBBwIKABQGBgBMLLAhYEMCCQFwQEwJOLHBigQUBCwICAScW
        WBCwIGBBAIFAPbHcWGBBwCcdLAgIBJxYYEHAgoAFAYEA88RyY4EFAZ90sCAgEBAI
        +IOAQMCJBQIBBALxD+ITAj7p4MQCgYBAwB8EBAJOLBAICATwB4EYiELAiQUCAYGA
        TzoIBJxYIBAQCPiDABYE4oKYELAgIBDwSQcLAhYELAgIBJxYYEGAuSAmBCwICAR8
        0sGCgAUBCwICAScWWBDAgkALRCHgxAInFlgQsCBgQUAg4MQCCwIWBLAgEBfEhIAT
        C5xYYEHAgoBAwIkFFgQsCFgQQCBQTyw3FlgQ8EkHCwICAScWWBCwIGBBQCDAPLHc
        WGBBwCcdLAgIBAQC/iAgEHBigUAAgUD8g/iEgE86OLFAICAQ8AcBgYATCwQCAgH8
        QSAGohBwYoFAQCDgkw4CAScWCAQEAv4ggAWBuCAmBCwICAR80sGCgAUBCwICAScW
        WBBgLogJAQsCAgGfdLAgYEHAgoBAwIkFFgSwINACUQg4scCJBRYELAhYEBAIOLHA
        goAFASwIxAUxIeDEAicWWBCwICAQcGKBBQELAhYEEAjUE8uNBRYEfNLBgoBAwIkF
        FgQsCFgQEAgwTyw3FlgQ8EkHCwICAYGAPwgIBJxYIBBAIBD/ID4h4JMOTiwQCAgE
        /EFAIODEAoGAQAB/EIiBKAScWCAQEAj4pINAwIkFAgGBgD8IYEEgLogJAQsCAgGf
```

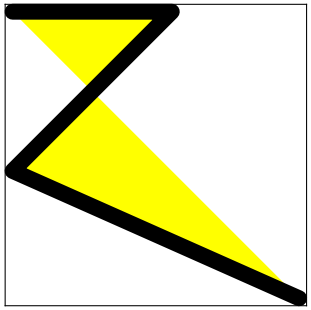
```
        dLAgYEHAgoBAwIkFFgSYC2JCwIKAQMAnHSwIWBCwICAQcGKBBQEsCLRAFAJOLHBi
        gQUBCwIWBAQCTiywIGBBAAsCcUFMCDixwIkFFgQsCAgEnFhgQcCCgAUBBAL1xHJj
        gQUBn3SwICAQcGKBBQELAhYEBALME8uNBRYEfNLBgoBAQCDgDwICAScWCAQQCMQ/
        iE8I+KSDEwsEAgIBfxAQCDixQCAgEMAfBGIgCgEnFggEBAI+6SAQcGKBQEAg4A8C
        WBCIC2JCwIKAQMAnHSwIWBCwICAQcGKBBQHmgpgQsCAgEPBJBwsCFgQsCAgEnFhg
        QQALAi0QhYATC5xYYEHAgoAFAYGAEwssCFgQwIJAXBATAk4scGKBBQELAgIBJxZY
        ELAgYEEAgUA9sdxYYEHAJx0sCAgEnFhgQcCCgAUBgQDzxHJjgQUBn3SwICAQEAj4
        g4BAwIkFAgEEAvEP4hMCPungxAKBgEDgH3wBrUwJtCBGuc0AAAAASUVORK5CYII=
```


```
    "/>
</svg>
```

Xah Lee test snippets

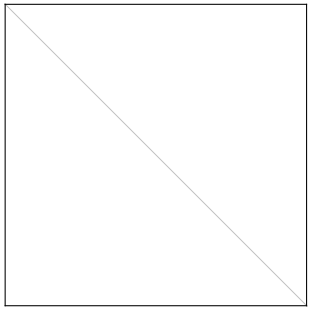
These snippets come from the `http://xahlee.info/js/svg_path_spec.html`, which gives a nice overview of svg. Not all examples are here. There are some nice interactive examples there plus info about using fonts.

1 Snippet 1



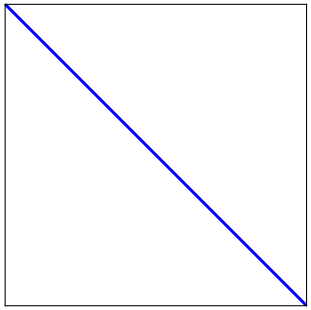
```svg
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow; stroke-width:5"
        d="M 0 0 L 50 0 L 0 50 L 90 90"
    />
</svg>
```

2 Snippet 2



```svg
<svg width="100" height="100">
    <path
        d="M 0 0 L 50 50"
    />
</svg>
```

3 Snippet 3

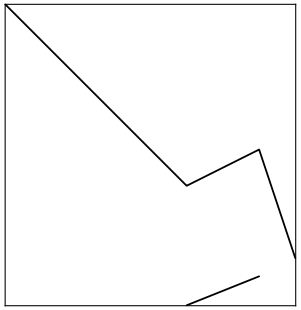


```
<svg width="100" height="100">
    <path
        style="stroke:blue"
        d="M 0 0 L 50 50"
    />
</svg>
```

4 Snippet 4

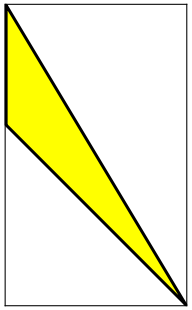
```
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow"
        d="m 50 0 l 8 5 l -8 5 l 8 5 l -8 5 l 8 5 l -8 5 l 8 5 l -8 5"
    />
</svg>
```

5 Snippet 5



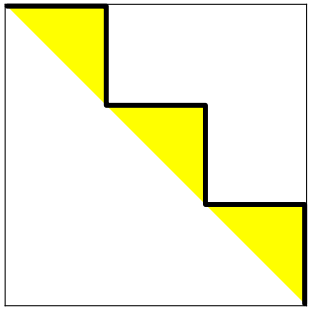
```svg
<svg width="100" height="100">
    <path
        style="stroke:black; fill:none;"
        d="M 0 0 L 50 50 l 20 -10 l 10 30 m -10 5 l -20 8"
    />
</svg>
```

6 Snippet 6



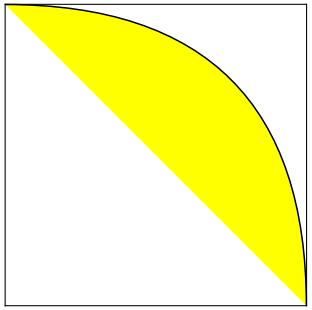
```
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow"
        d="m 10 10 l 0 20 l 30 30 z"
    />
</svg>
```

7 Snippet 7



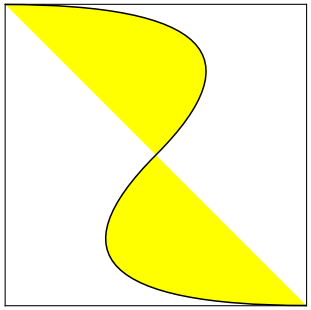
```
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow"
        d="M 10 10 h 10 v 10 h 10 v 10 h 10 v 10"
    />
</svg>
```

8 Snippet 8



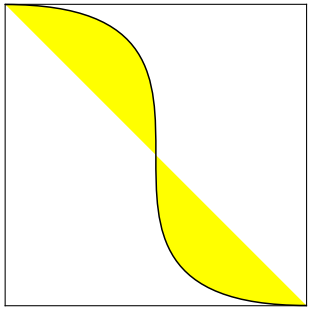
```
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow"
        d="M 0 0 Q 100 0 , 100 100"
    />
</svg>
```

9 Snippet 9



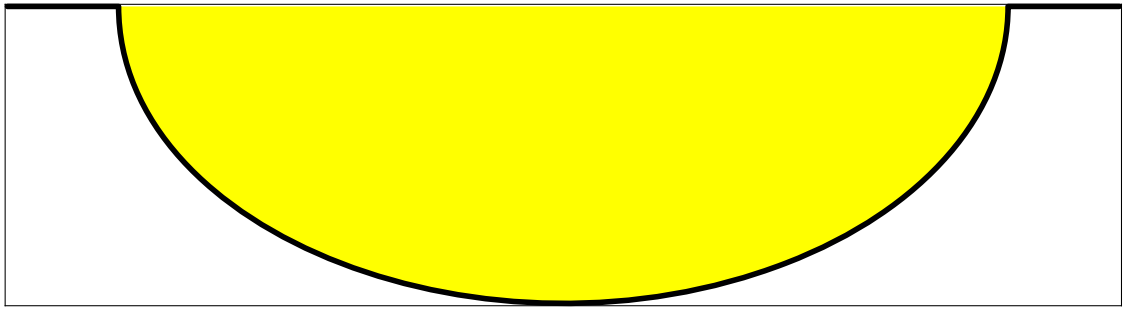
```
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow"
        d="M 0 0 Q 100 0 , 50 50 T 100 100"
    />
</svg>
```

10 Snippet 10



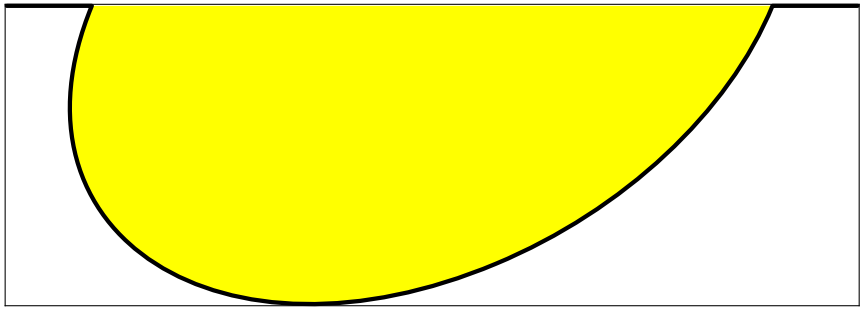
```
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow"
        d="M 0 0 C 100 0, 0 100, 100 100"
    />
</svg>
```

11 Snippet 11



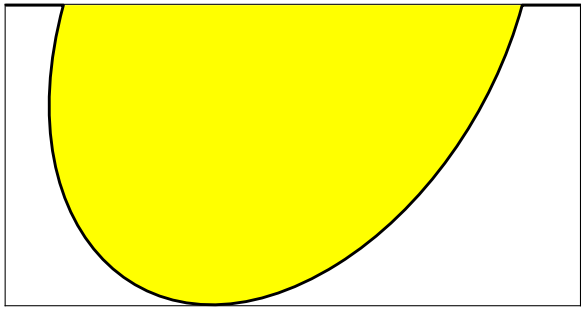
```
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow"
        d="M 0 50 L 10 50 A 30 20, 0, 0 0, 90 50 L 100 50"
    />
</svg>
```

12 Snippet 12



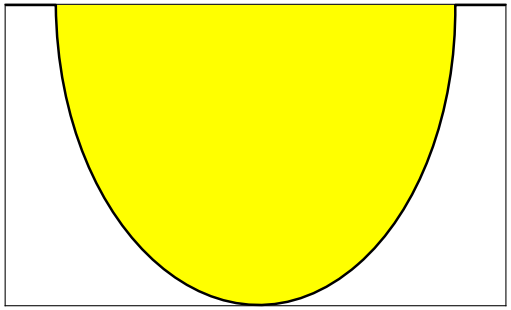
```
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow"
        d="M 0 50 L 10 50 A 3 2, 30, 0 0, 90 50 L 100 50"
    />
</svg>
```

13 Snippet 13



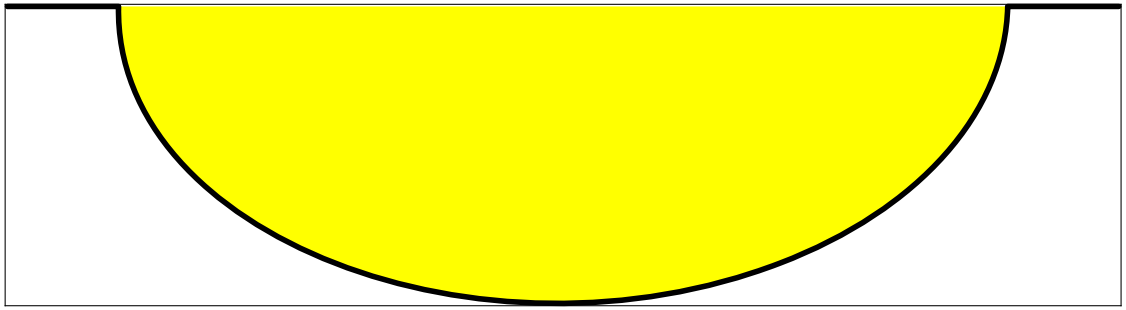
```
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow"
        d="M 0 50 L 10 50 A 3 2, 61, 0 0, 90 50 L 100 50"
    />
</svg>
```

14 Snippet 14



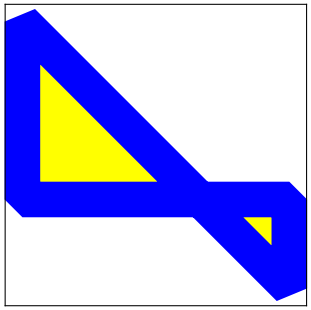
```
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow"
        d="M 0 50 L 10 50 A 3 2, 91, 0 0, 90 50 L 100 50"
    />
</svg>
```

15 Snippet 15



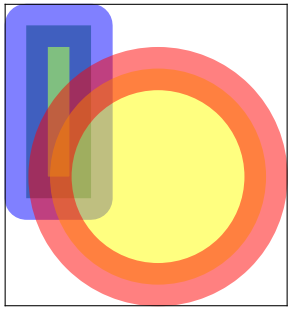
```
<svg width="100" height="100">
    <path
        style="stroke:black; fill:yellow"
        d="M 0 50 L 10 50 A 3 2, 181, 0 0, 90 50 L 100 50"
    />
</svg>
```

16 Snippet 16



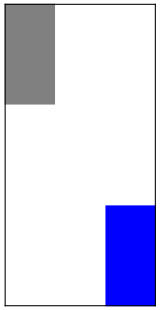
```
<svg width="200" height="200">
<polygon
    style="stroke: blue; fill: yellow; stroke-width:20; stroke-linejoin:bevel"
    points="0 0 150 150 150 100 0 100"
/>
</svg>
```

17 Snippet 17



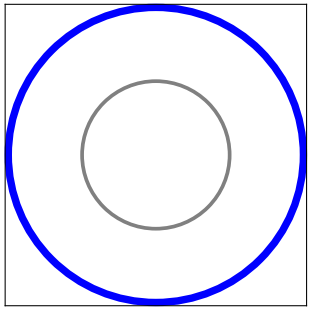
```
<svg width="100" height="100">
    <rect
        style="fill:green; stroke:blue; stroke-width:20; stroke-opacity:.5; fill-opacity:.5"
        x="9" y="0" width="30" height="80"
    />
    <circle
        style="fill:yellow; stroke:red; stroke-width:20; stroke-opacity:.5; fill-opacity:.5"
        cx="70" cy="70" r="50"
    />
</svg>
```

18 Snippet 18



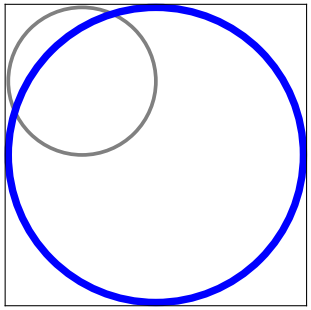
```
<svg width="100" height="100">
    <rect
        style="fill:gray"
        x="0" y="0" width="10" height="20"
    />
    <rect
        style="fill:blue;"
        x="0" y="0" width="10" height="20"
        transform="translate(20 40)"
    />
</svg>
```

19 Snippet 19



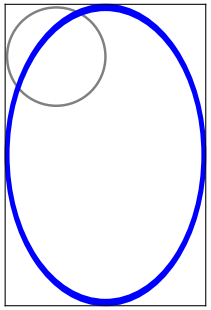
```
<svg width="100" height="100">
<circle
    cx="0" cy="0" r="10"
    style="fill:none; stroke:gray"
/>
<circle
    cx="0" cy="0" r="10"
    style="fill:none; stroke:blue" transform="scale(2)"
/>
</svg>
```

20 Snippet 20



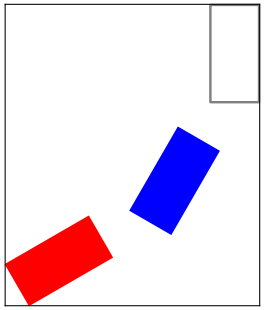
```
<svg width="100" height="100">
    <circle
        cx="10" cy="10" r="10"
        style="fill:none; stroke:gray"
    />
    <circle
        cx="10" cy="10" r="10"
        style="fill:none; stroke:blue" transform="scale(2)"
    />
</svg>
```

21 Snippet 21



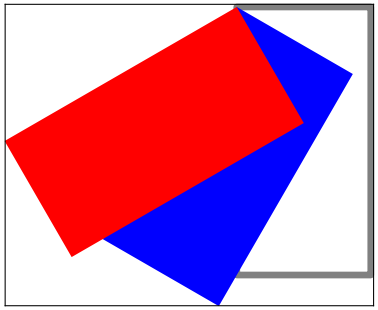
```
<svg width="100" height="100">
    <circle
        cx="10" cy="10" r="10"
        style="fill:none; stroke:gray"
    />
    <circle
        cx="10" cy="10" r="10"
            style="fill:none; stroke:blue"
            transform="scale(2, 3)"
    />
</svg>
```

22 Snippet 22



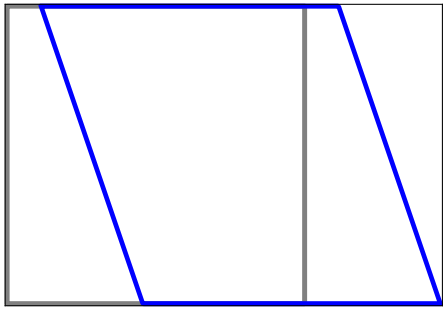
```
<svg width="100" height="100">
    <rect
        x="50" y="0" width="10" height="20"
        style="fill:none; stroke:gray;"
    />
    <rect
        x="50" y="0" width="10" height="20"
        style="fill:blue;"
        transform="rotate(30)"
    />
    <rect
        x="50" y="0" width="10" height="20"
        style="fill:red;"
        transform="rotate(60)"
    />
</svg>
```

23 Snippet 23



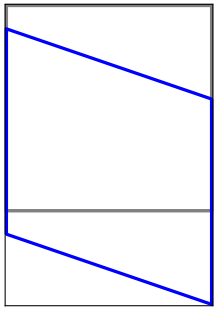
```
<svg width="100" height="100">
    <rect
        x="50" y="0" width="10" height="20"
        style="fill:none; stroke:gray;"
    />
    <rect
        x="50" y="0" width="10" height="20"
        style="fill:blue;"
        transform="rotate(30 50 0)"
    />
    <rect
        x="50" y="0" width="10" height="20"
        style="fill:red;"
        transform="rotate(60 50 0)"
    />
</svg>
```

24 Snippet 24



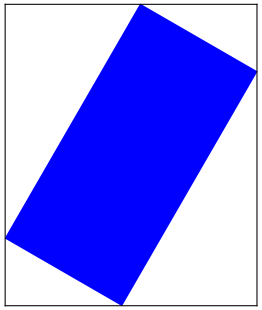
```
<svg width="100" height="100">
    <rect
        x="10" y="10" width="30" height="30"
        style="fill:none; stroke:gray;"
    />
    <rect
        x="10" y="10" width="30" height="30"
        style="fill:none; stroke:blue;"
        transform="skewX(20)"
    />
</svg>
```

25 Snippet 25



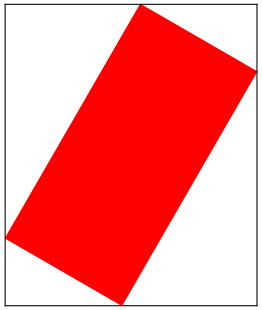
```
<svg width="100" height="100">
    <rect
        x="10" y="10" width="30" height="30"
        style="fill:none; stroke:gray;"
    />
    <rect
        x="10" y="10" width="30" height="30"
        style="fill:none; stroke:blue;"
        transform="skewY(20)"
    />
</svg>
```

26 Snippet 26



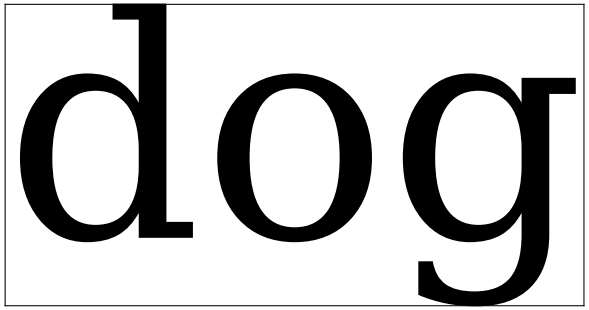
```
<svg width="100" height="100">
    <rect
        x="0" y="0" width="10" height="20"
        style="fill:blue;"
        transform="rotate(30) translate(40 0)"
    />
</svg>
```

27 Snippet 27



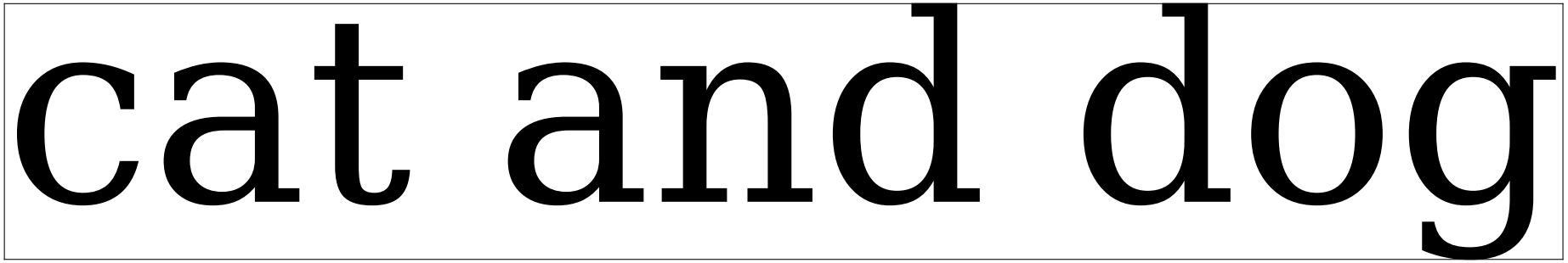
```
<svg width="100" height="100">
    <rect
        x="0" y="0" width="10" height="20"
        style="fill:red;"
        transform="translate(40 0) rotate(30)"
    />
</svg>
```

28 Snippet 28



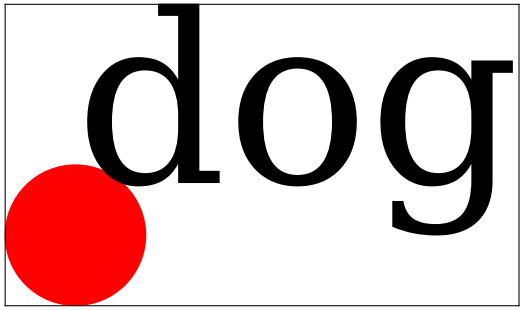

```
<svg width="100" height="100">
    <text x="50" y="50">dog</text>
</svg>
```

29 Snippet 29



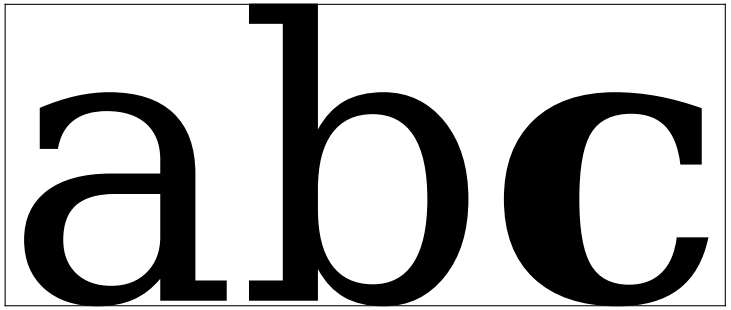

```svg
<svg width="100" height="100">
    <text x="0" y="50">cat and dog</text>
</svg>
```

30 Snippet 30



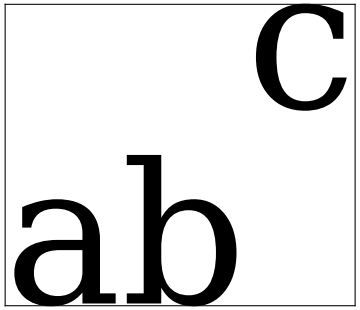

```
<svg width="100" height="100">
    <circle
        cx="50" cy="50" r="3"
        style="fill:red"
    />
    <text x="50" y="50">dog</text>
</svg>
```

31 Snippet 31



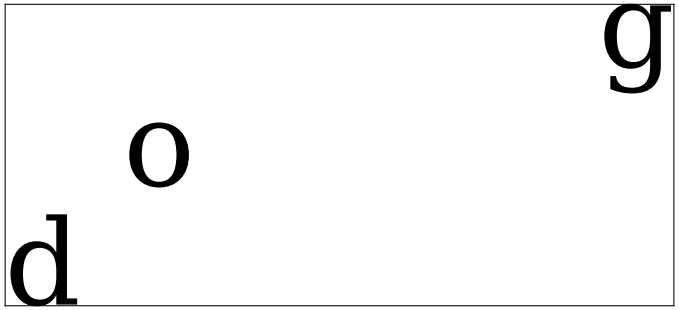

```
<svg width="100" height="100">
    <text x="0" y="50">ab<tspan style="font-weight:bold">c</tspan></text>
</svg>
```

32 Snippet 32




```
<svg width="100" height="100">
    <text x="0" y="50">ab<tspan dy="-10">c</tspan></text>
</svg>
```

33 Snippet 33

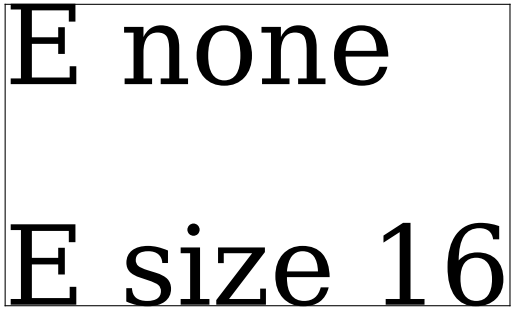



```
<svg width="100" height="100">
    <text x="0 10 50" y="100 90 80">dog</text>
</svg>
```

34 Snippet 34

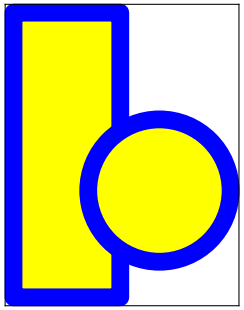

```
<svg width="100" height="100">
    <text x="50" y="50" transform="rotate(-90 50 50)">mouse</text>
</svg>
```

35 Snippet 35



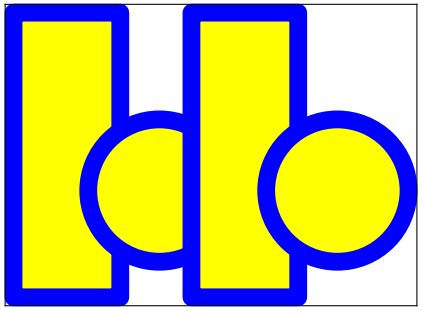

```
<svg width="100" height="100">
    <text x="0" y="50">E none</text>
    <text x="0" y="70" font-size="16">E size 16</text>
</svg>
```

36 Snippet 36



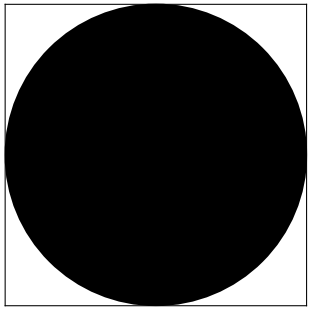
```
<svg width="100" height="100">
    <g id="abc46" style="fill:yellow; stroke:blue; stroke-width:5" >
        <rect x="9" y="0" width="30" height="80"/>
        <circle cx="50" cy="50" r="20"/>
    </g>
</svg>
```

37 Snippet 37



```
<svg width="100" height="100">
    <g id="abc46" style="fill:yellow; stroke:blue; stroke-width:5" >
        <rect x="9" y="0" width="30" height="80"/>
        <circle cx="50" cy="50" r="20"/>
    </g>
    <use xlink:href="#abc46" x="50" y="0" />
</svg>
```

38 Snippet 38

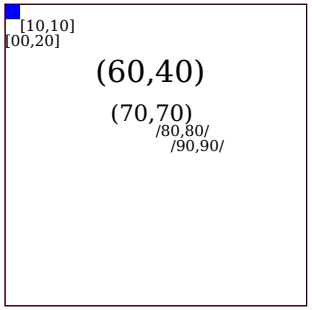


```
<svg width="100" height="100">
    <defs id="c">
        <circle cx="10" cy="10" r="10"/>
    </defs>
    <use xlink:href="#c" x="50" y="0" />
</svg>
```

Our own snippets

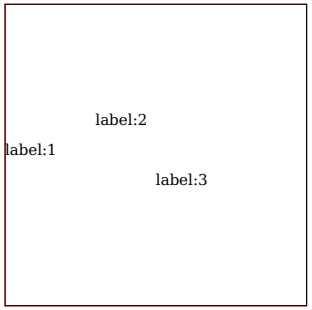
These snippets were made as part if testing. I had some 1500 svg graphics that internally were quite messy (it’s surprising what some applications export) so I sometimes had to extract bits and pieces and make my own tests to figure out how to deal with it.

1 Snippet 1




```
<svg>
    <rect x="0" y="0" width="010" height="010" fill="blue" />
    <rect x="0" y="0" width="200" height="200" fill="none" stroke="red" />
    <text x="000" y="100"
        ><tspan x="000" y="20">[00,20]</tspan
        ><tspan x="010" y="10">[10,10]</tspan
    ></text>
    <text x="060" y="080" font-size="10px"
        ><tspan x="060" y="40" font-size="20px">(60,40)</tspan
        ><tspan x="070" y="70" font-size="15px">(70,70)</tspan
    ></text>
    <text x="100" y="100"><tspan x="080" y="80">/80,80/</tspan
        ><tspan x="090" y="90">/90,90/</tspan
    ></text>
</svg>
```

2 Snippet 2




```
<svg>
    <rect x="0" y="0" width="200" height="200" fill="none" stroke="red" />
    <text x="000" y="100">label:1</text>
    <text x="060" y="080">label:2</text>
    <text x="100" y="120">label:3</text>
</svg>
```