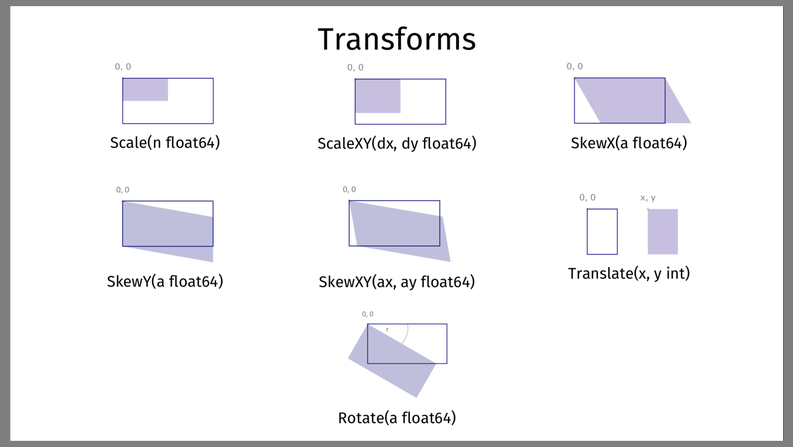
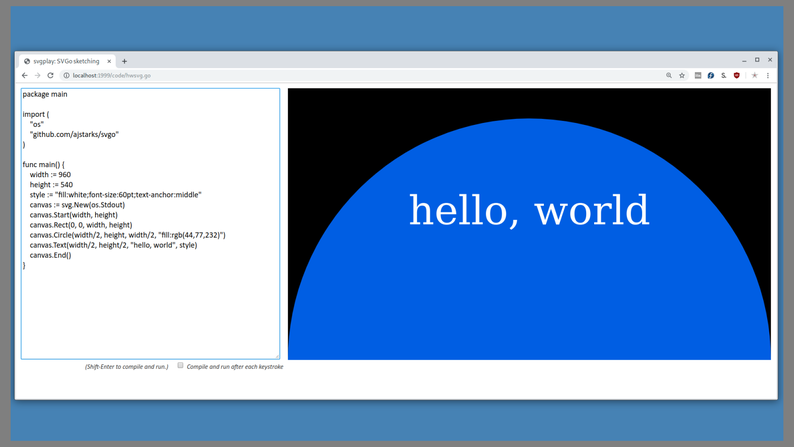
Workshop

Anthony Starks
@ajstarks

You will learn:



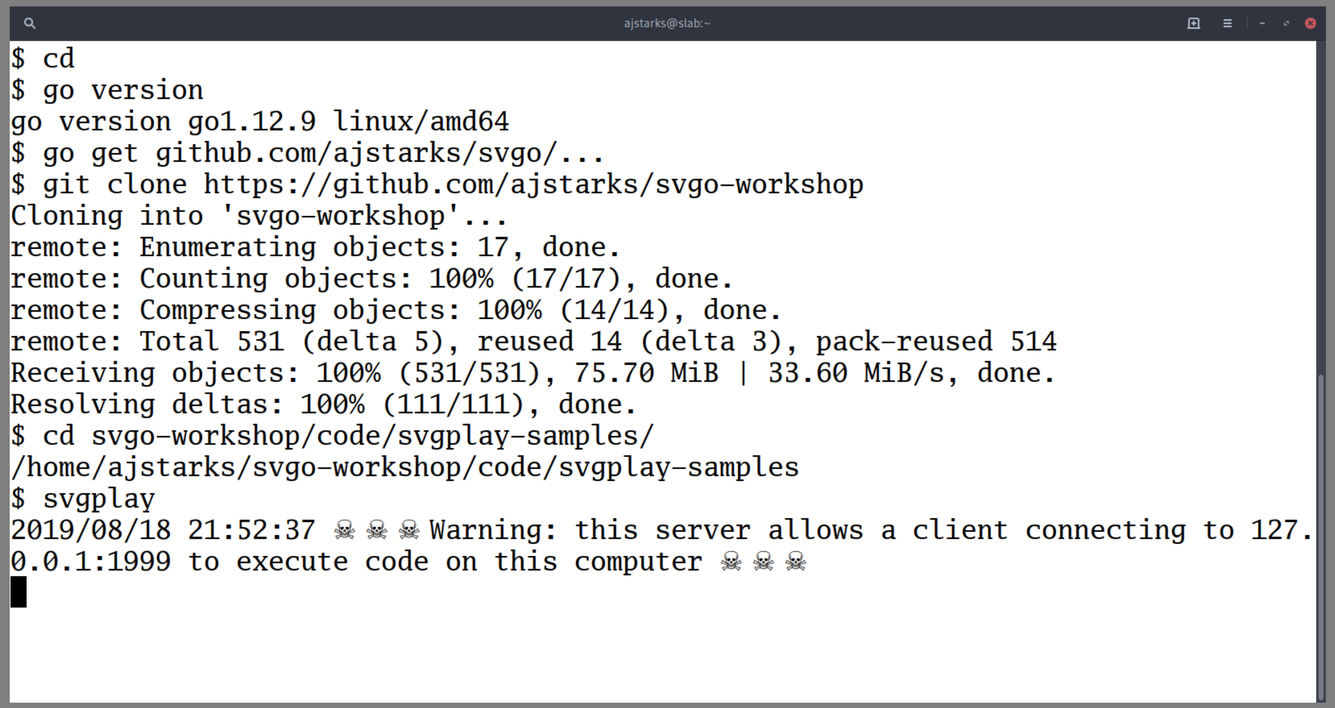


SVGo API

Sketching with code

SVGo with other APIs

You will need:

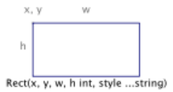


Go installation and command line

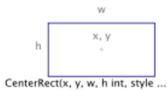
Modern Browser

SVG Go Library

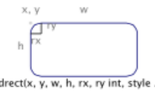
github.com/ajstarks/svgo



Rect(x, y, w, h int, style ...string)



CenterRect(x, y, w, h int, style ...string)

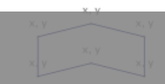


Roundrect(x, y, w, h, rx, ry int, style ...string)

Square(x, y, w int, style ...string)

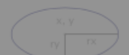
Line(x1, y1, x2, y2 int, style ...string)

Polyline(x, y []int, style ...string)



Polygon(x, y []int, style ...string)

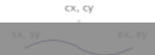
Circle(x, y, r int, style ...string)



Ellipse(x, y, rx, ry int, style ...string)

Arc(sx, sy, ax, ay, r int, lflag, sflag bool, ex, ey int, style ...string)

Qbez(sx, sy, cx, cy, ex, ey int, style ...string)



Bezier(sx, sy, cx, cy, px, py, ex, ey int, style ...string)

Translate(x, y int)

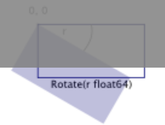
Scale(n float64)

ScaleXY(x, y float64)

SkewX(a float64)

SkewY(a float64)

SkewXY(x, y float64)



Rotate(r float64)

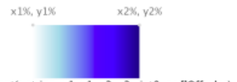
Text(x, y int, s string, style ...string)

Textpath(s, pathid string, style ...string)

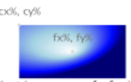
Path(s string, style ...string)

Image(x, y, w, h, int path string, style ...string)

Grid(x, y, w, h, n int, style ...string)



LinearGradient(s string, x1, y1, x2, y2 uint8, oc []Offcolor)



RadialGradient(s string, cx, cy, r, fx, fy uint8, oc []Offcolor)

RGB(r, g, b int)

RGBA(r, g, b int, opacity float64)

New(w io Writer)	specify destination
Start(w, h int, options ...string)/End()	begin/end the document
Startview(w, h, minx, miny, vw, vh int)	begin/end the document with viewport
Group(s ...string)/Gend()	begin/end group with attributes
Gstyle(s string)/Gend()	begin/end group style
Gtransform(s string)/Gend()	begin/end group transform
Gid(id string)/Gend()	begin/end group id
ClipPath(s ...string)/ClipEnd()	begin/end clip path
Def()/DefEnd()	begin/end a defintion block
Marker()/MarkerEnd()	begin/end markers
Pattern()/PatternEnd()	begin/end pattern
Desc(s string)	set the description element
Title(s string)	set the title element
Script(type, data ...string)	define a script
Mask(id string, x,y,w,h int, style ...string)/MaskEnd()	begin/end mask element
Link(href string, title string)/LinkEnd()	begin/end link to href, with a title
Use(x int, y int, link string, style ...string)	use defined objects

Textlines(x, y int, s []string, size, spacing int, fill, align string)

```
$ go get github.com/ajstarks/svgo/...
$ git clone https://github.com/ajstarks/svgo-workshop
$ cd svgo-workshop/code/svgplay-samples
```

Scalable Vector Graphics

`io.Writer`

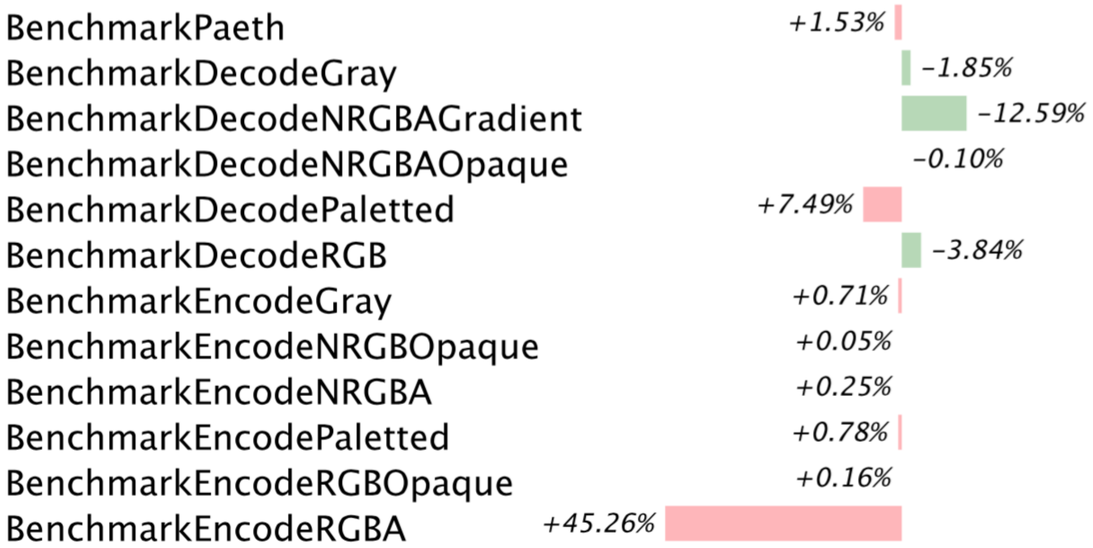
SVGo Clients in the Repo

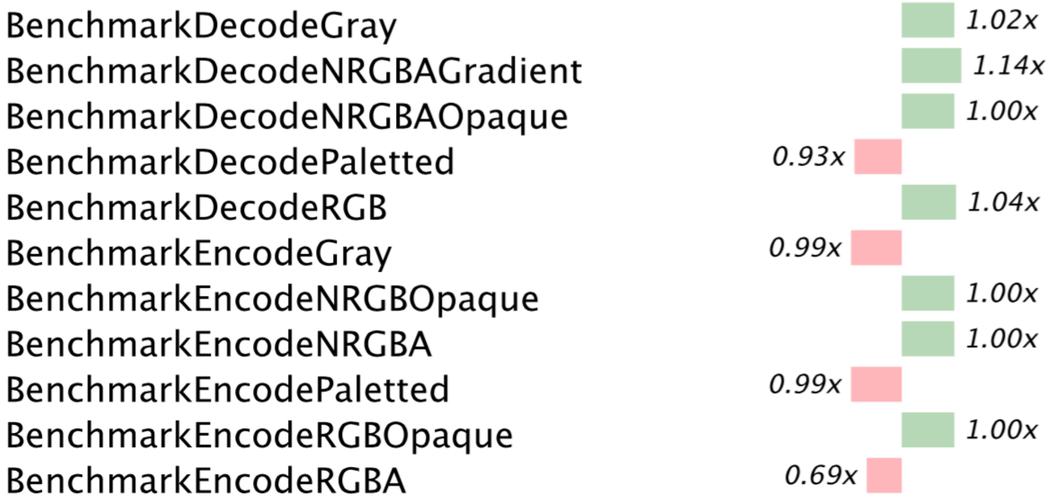
`svgdef`	Creates a SVG representation of the API
`android`	The Android logo
`bubtrail`	Bubble trails
`bulletgraph`	Bullet Graphs (via Stephen Few)
`colortab`	Display SVG named colors with RGB values
`compx`	Component diagrams
`flower`	Random flowers
`fontcompare`	Compare two fonts
`f50`	Get 50 photos from Flickr from a query
`fe`	Filter effects
`funnel`	Funnel from transparent circles
`gradient`	Linear and radial gradients
`html5logo`	HTML5 logo with draggable elements
`imfade`	Show image fading
`lewitt`	Version of Sol Lewitt's Wall Drawing 91
`ltr`	Layer Tennis Remixes
`marker`	Test markers
`paths`	Demonstrate SVG paths
`pattern`	Test patterns
`planets`	Show the scale of the Solar system
`pmap`	Proportion maps
`randcomp`	Compare random number generators
`richter`	Gerhard Richter's 256 colors
`rl`	Random lines
`skewabc`	Skew ABC
`stockproduct`	Visualize product and stock prices
`svgopher`	SVGo Mascot
`svgplay`	SVGo sketching server
`svgplot`	Plot data
`svgrid`	Compose SVG files in a grid
`tsg`	Twitter Search Grid
`tumblrgrid`	Tumblr picture grid
`turbulence`	Turbulence filter effect
`vismem`	Visualize data from files
`webfonts`	"Hello, World" with Google Web Fonts
`websvg`	Generate SVG as a web server

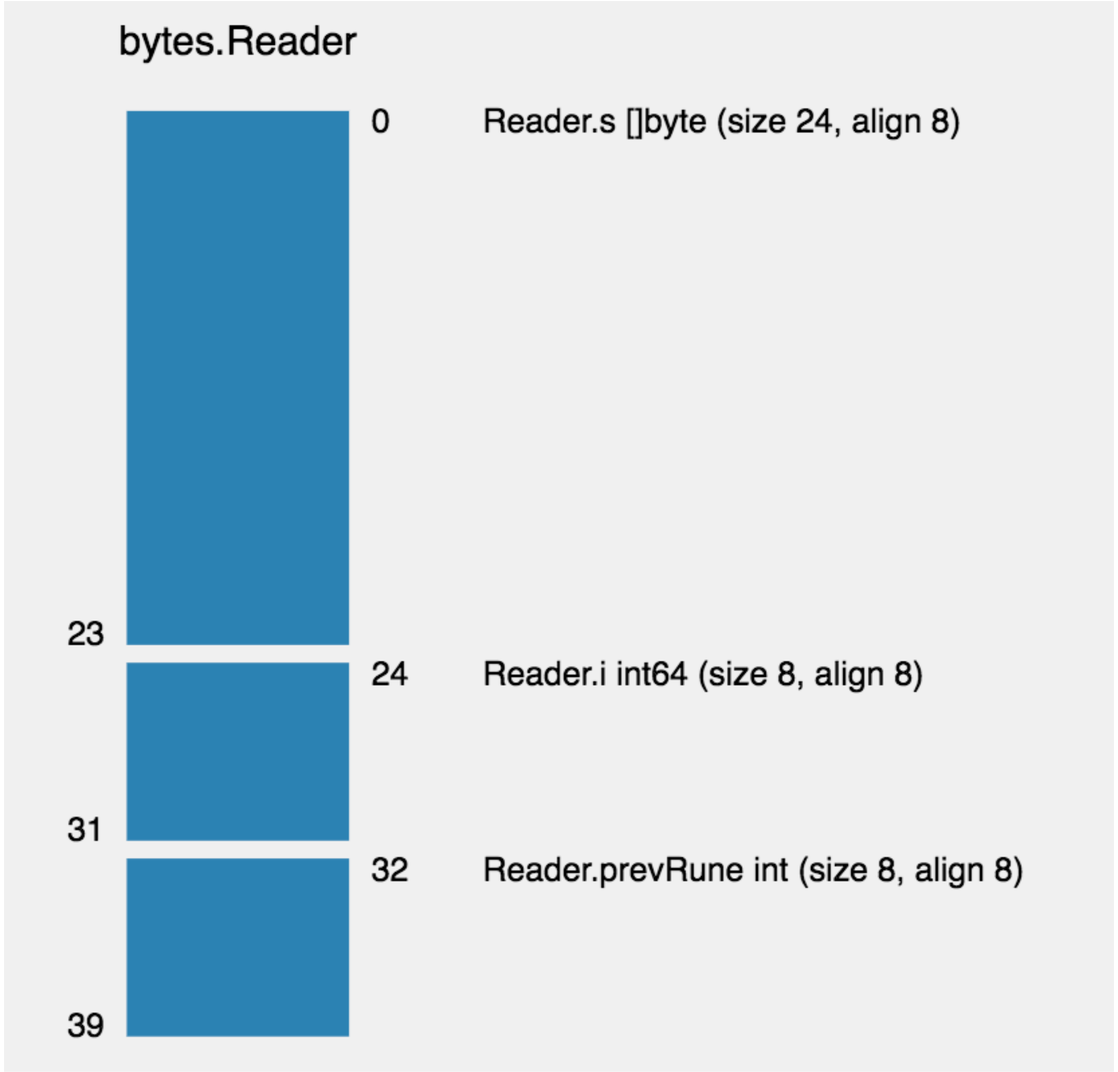
SVGo Clients

benchpng.txt

image/png package: Go 1.3 vs Go 1.4

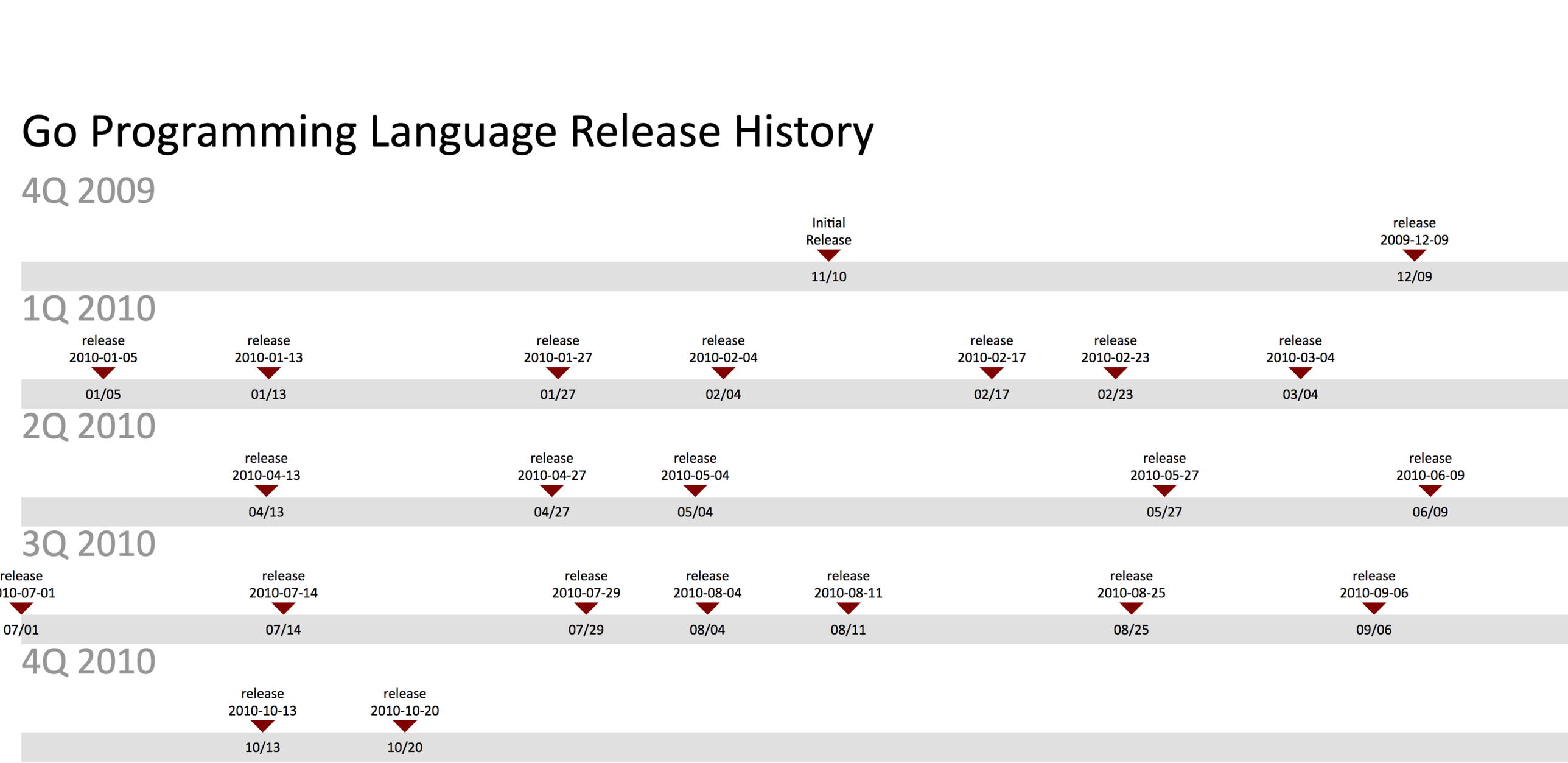


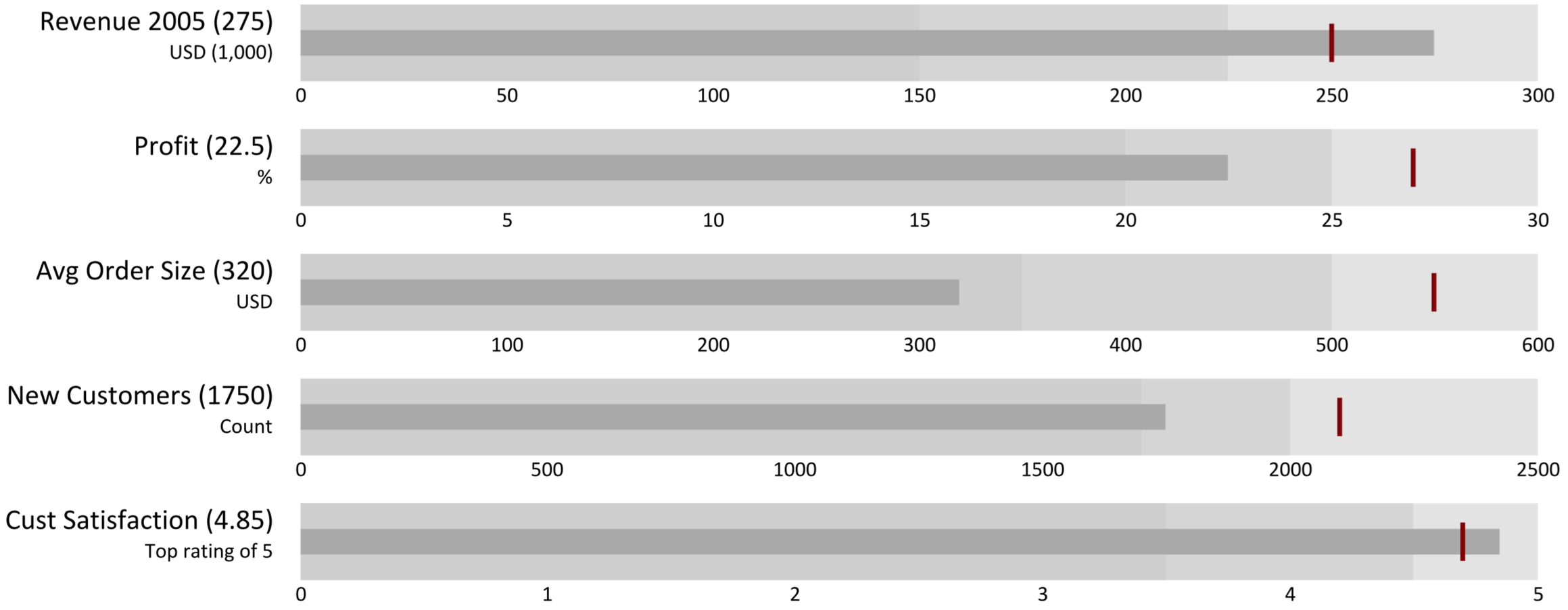




structlayout -json bytes Reader | structlayout-svg -t bytes.Reader > reader.svg

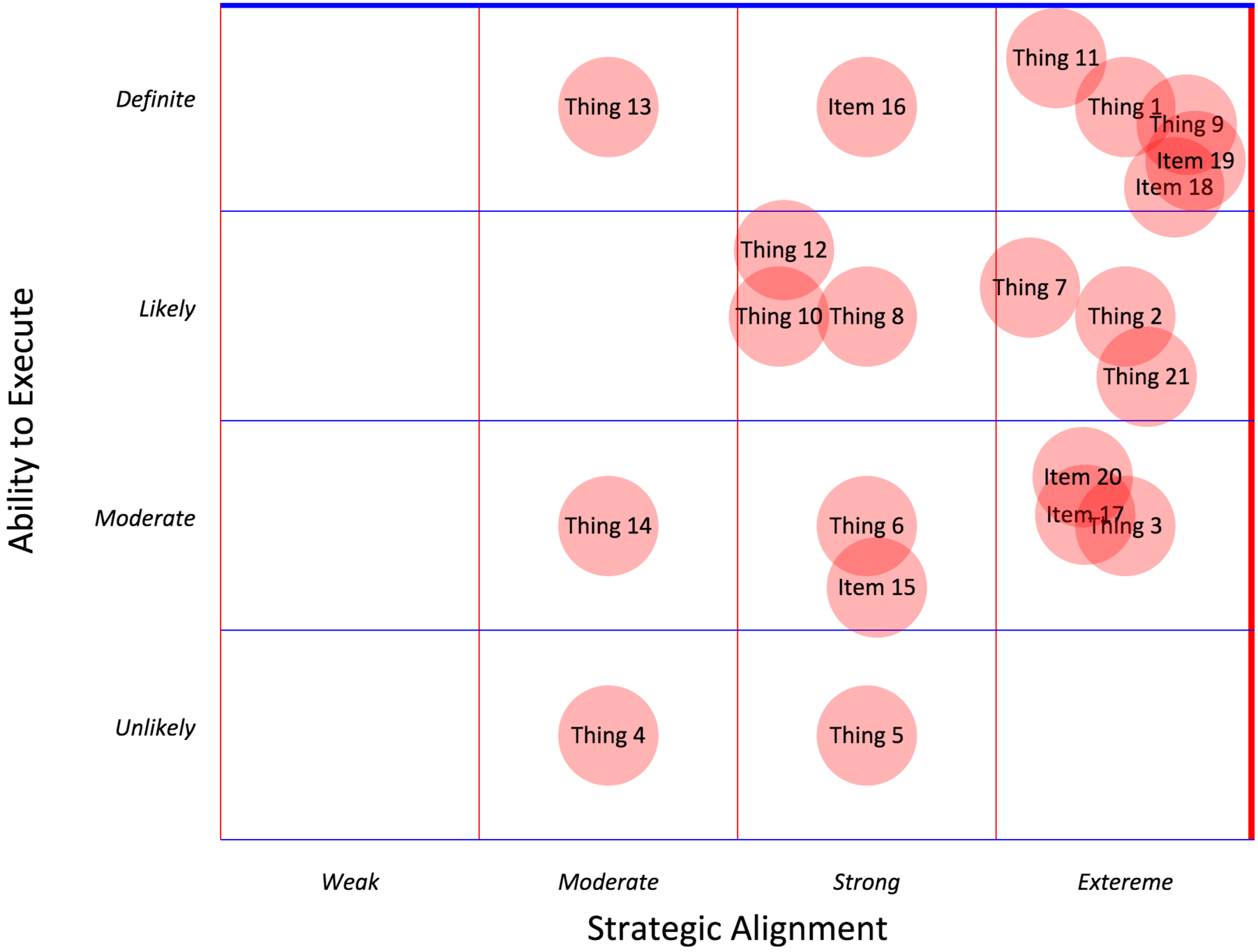
Go Programming Language Release History

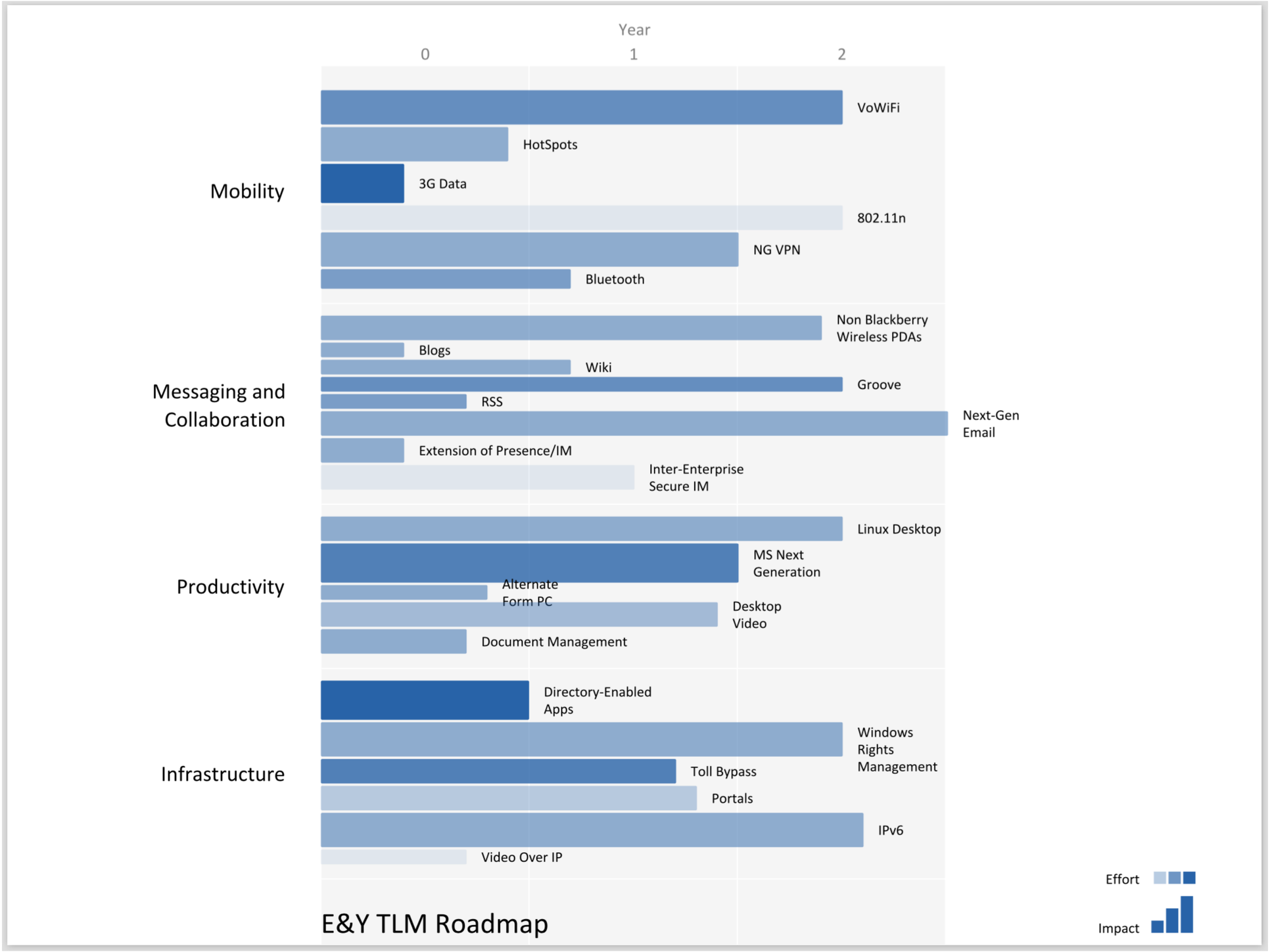


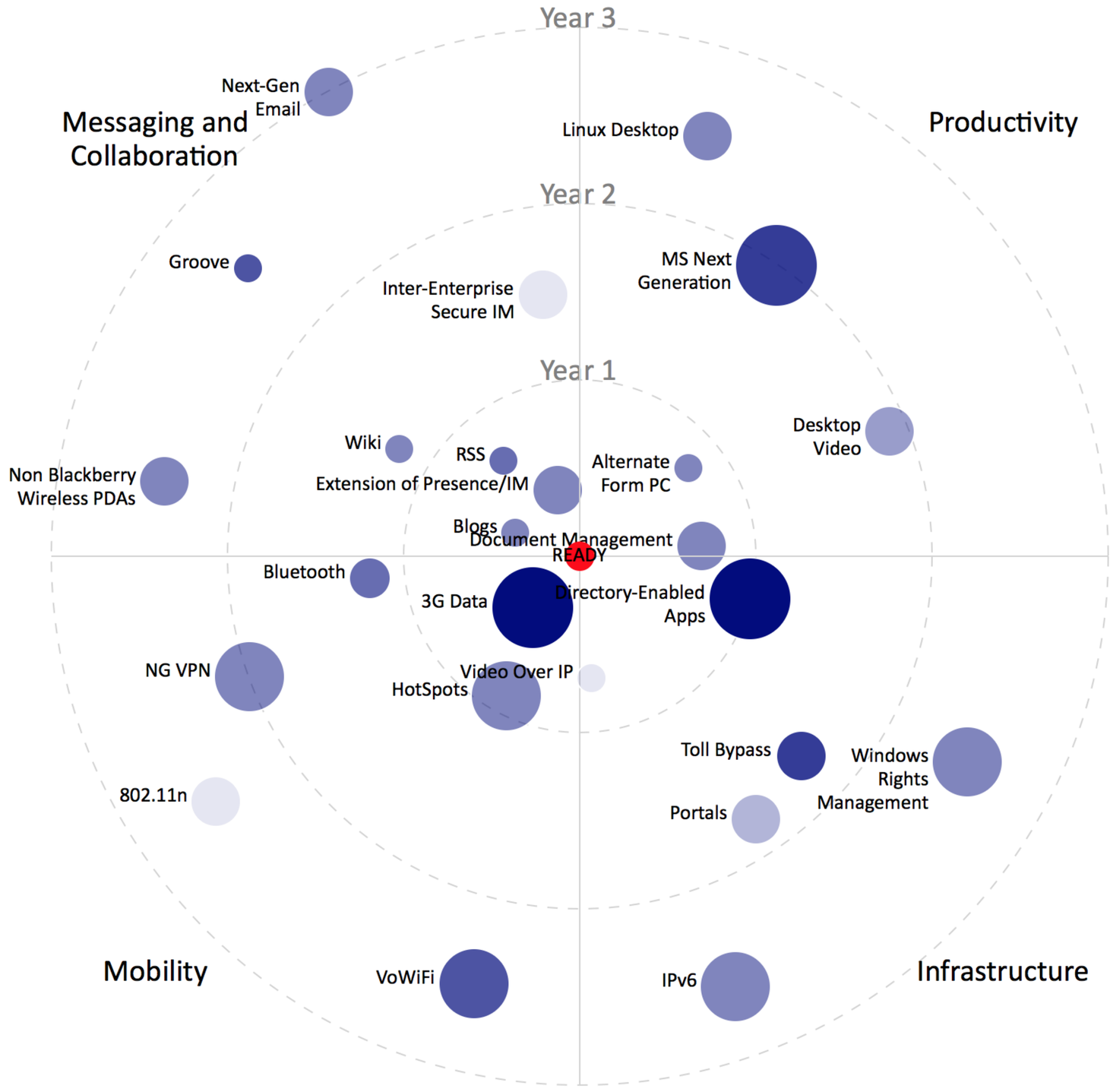


Sample Bullet Graph

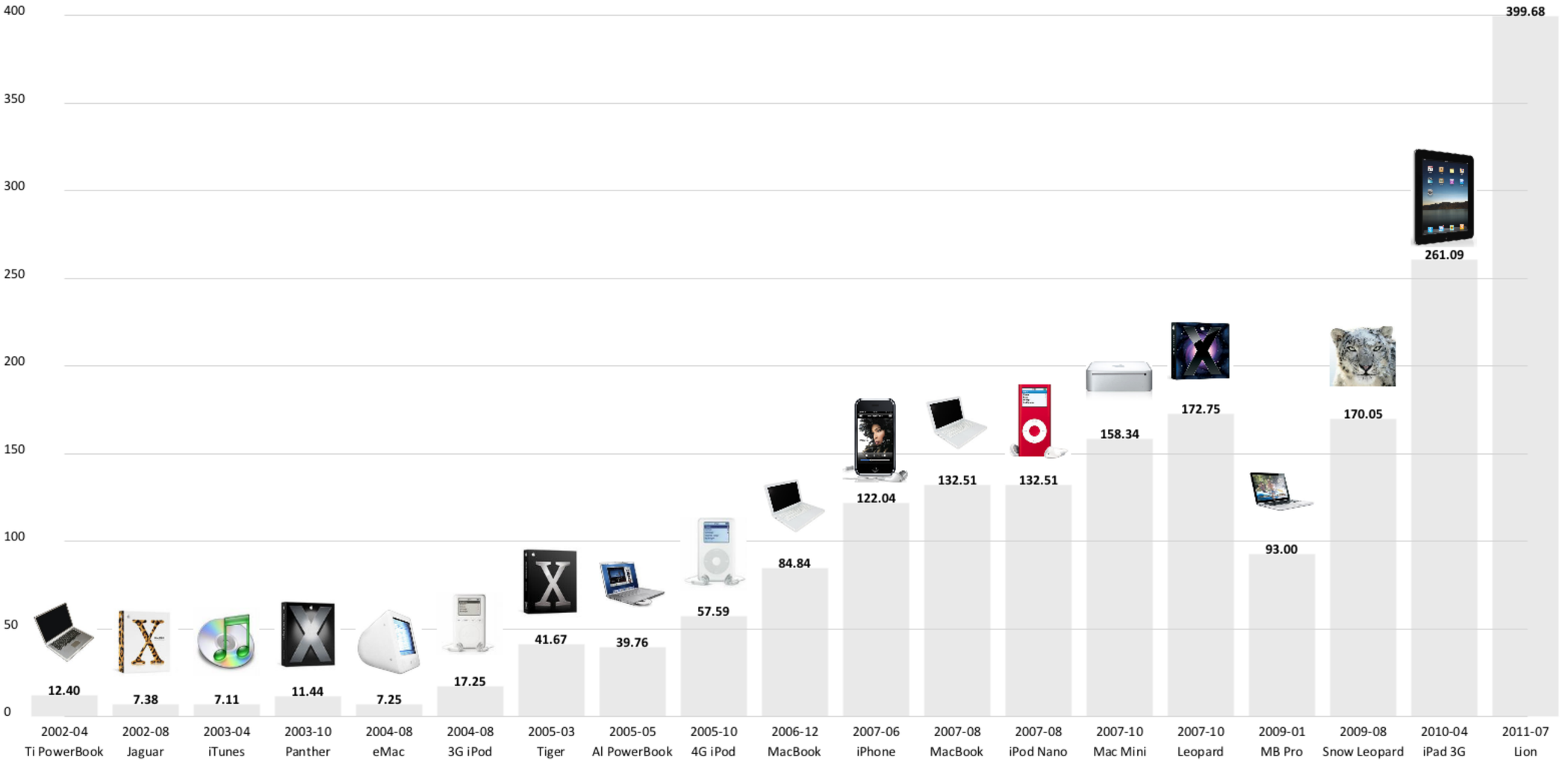
The bullet graph features a single, primary measure (for example, current year-to-date revenue) compares that measure to one or more other measures to enrich its meaning, for example, compared to a target), and displays it in the context of qualitative ranges of performance, such as poor, satisfactory, and good.


Ability to Execute
Definite
Likely
Moderate
Unlikely
Thing 13
Item 16
Thing 11
Thing 1
Thing 9
Item 19
Item 18
Thing 12
Thing 10
Thing 8
Thing 7
Thing 2
Thing 21
Thing 14
Thing 6
Item 15
Item 20
Item 17
Thing 3
Thing 4
Thing 5
Weak
Moderate
Strong
Extereme
Strategic Alignment

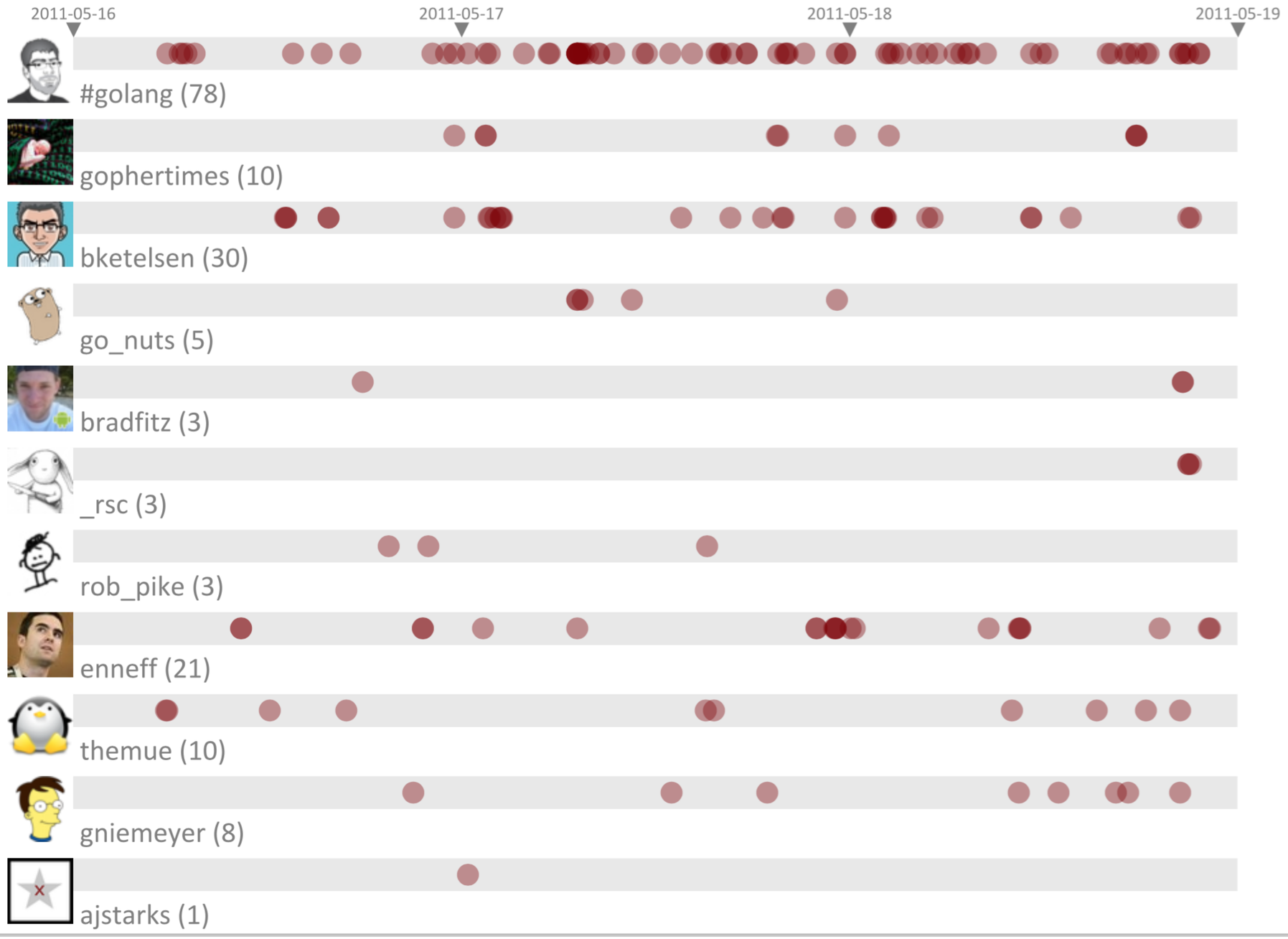

Year
0
1
2
Mobility
VoWiFi
HotSpots
3G Data
802.11n
NG VPN
Bluetooth
Messaging and Collaboration
Non Blackberry Wireless PDAs
Blogs
Wiki
Groove
RSS
Next-Gen Email
Extension of Presence/IM
Inter-Enterprise Secure IM
Productivity
Linux Desktop
MS Next Generation
Alternate Form PC
Desktop Video
Document Management
Infrastructure
Directory-Enabled Apps
Windows Rights Management
Toll Bypass
Portals
IPv6
Video Over IP
E&Y TLM Roadmap
Effort
Impact


Year 3
Year 2
Year 1
Messaging and Collaboration
Productivity
Mobility
Infrastructure
Next-Gen Email
Linux Desktop
Groove
Inter-Enterprise Secure IM
MS Next Generation
Desktop Video
Wiki
RSS
Alternate Form PC
Non Blackberry Wireless PDAs
Extension of Presence/IM
Blogs
Document Management
READY
Bluetooth
3G Data
Directory-Enabled Apps
NG VPN
Video Over IP
HotSpots
Toll Bypass
Windows Rights Management
802.11n
Portals
VoWiFi
IPv6

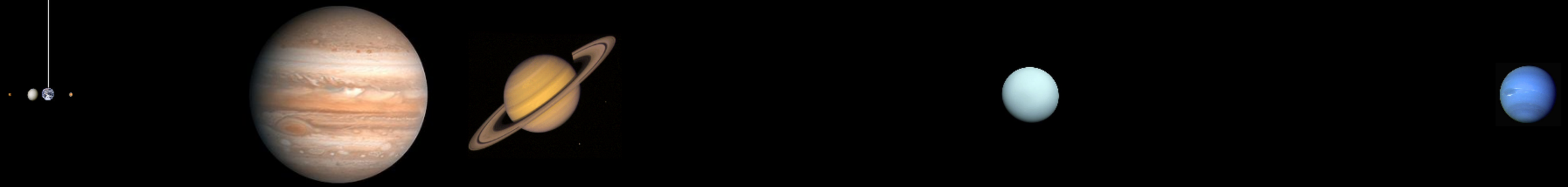
Apple Purchases and Stock Price



Twitter Update Frequency



You are here

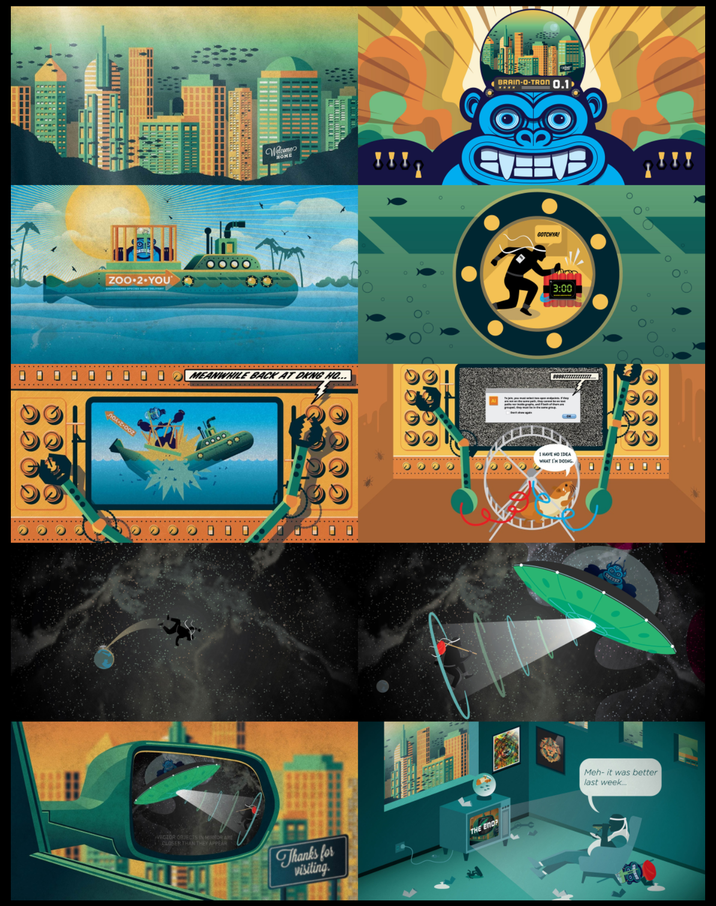


JONES
MONTIEL

- LAYER TENNIS SEASON 4 WEEK 6 MATCH COMMENTARY BY MIKE MONTEIRO -

DKNG
DDL

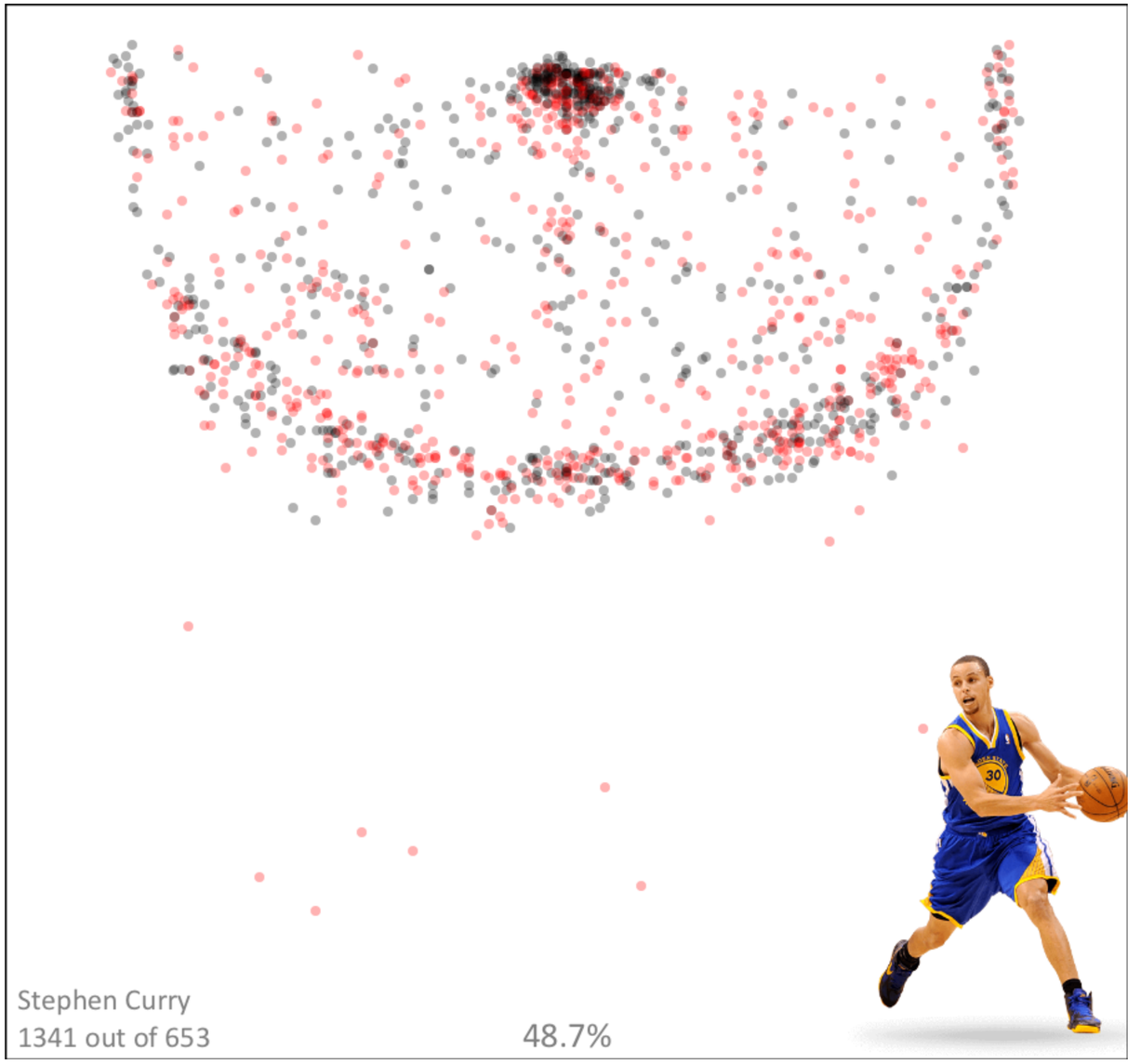
- LAYER TENNIS SEASON 4 WEEK 7 MATCH COMMENTARY BY JOHN GRUBER -

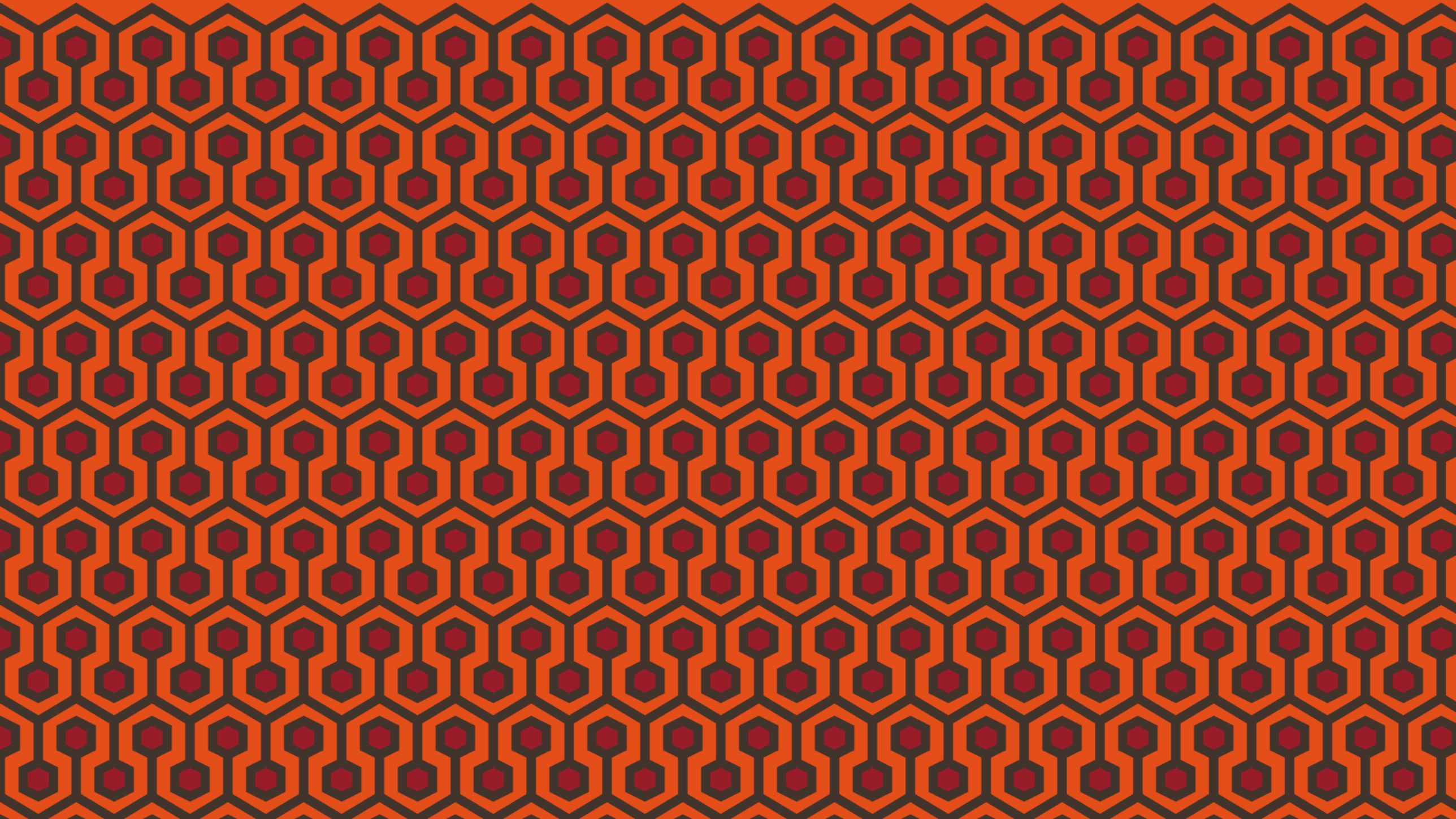


ANDERSON
DKNG

- LAYER TENNIS SEASON 4 PLAYOFF QUARTERFINAL COMMENTARY BY BRYAN BEDELL -


CONTINO CASSARO
PUTNAM STOCKS
WHITE TAYLOR
HALL WARREN
STRAWBERRYLUNA DOUBLENAUT
JONES MONTIEL
DKNG DDL
SHAWNA X STEVENS
TAYLOR JONES
PUTNAM RAJKUMAR
REYES WHITE
ANDERSON DKNG
TAYLOR WHITE
RAJKUMAR ANDERSON
ANDERSON WHITE
A FIGHT TO THE DEATH!
THE BATTLE OF LONG ISLAND
DIVIDE AND CONQUER
DIVIDE AND CONQUER?
JON CONTINO SOMEHOW TURNS INTO AN EAGLE!!
CUT OFFS
2022: A SPACE ODDITY
LANDING... 5 4 3 2 1
The Great British Autumn
Autumn HARVEST
OR
THEY MAY EAT YOUR PENGUIN
WHAT THE PUCK?
DECK THE HALL THIS CHRISTMAS EVE
HALLO ON A SEC AREN'T WE GETTING A BIT HASTEVE?! IT'S ONLY OCTOBER
SORRY AM I THE ONLY ONE WITH A TIME MACHINE?
WE SAY ABOUT THEY SAY A
REGULAR
SUPER SIZE ME!
VISIT THE "GREAT" WHITE NORTH
THE GREAT WHITE NORTH
CANADIANS HAVE BIGGER BALLS
"OUR BEER IS BETTER" QUOTE GOES HERE.
EVERYONE KNOWS THAT'S NOT TRUE
bronze
bro hoe
whamf
PENGUIN BURGER
It wasn't me
She won't be missed for long.
CASSANDRA THUNDER
07:30
OINK
OOPS!
SNOOZE
200M
LET'S GO CRAZY


Stephen Curry
1341 out of 653
48.7%

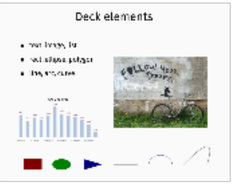


```
canvas.Def()
canvas.Gid("unit")
canvas.Polyline(xl, yl, "fill:none")
canvas.Polygon(xp, yp)
canvas.Gend()

canvas.Gid("runit")
canvas.TranslateRotate(150, 180, 180)
canvas.Use(0, 0, "#unit")
canvas.Gend()
canvas.Gend()
canvas.DefEnd()

for y := 0; < height; y += 130 {
    for x := 100; x < width; x+=100 {
        canvas.Use(x, y, "#unit")
        canvas.Use(x, y, "#runit")
    }
}
```

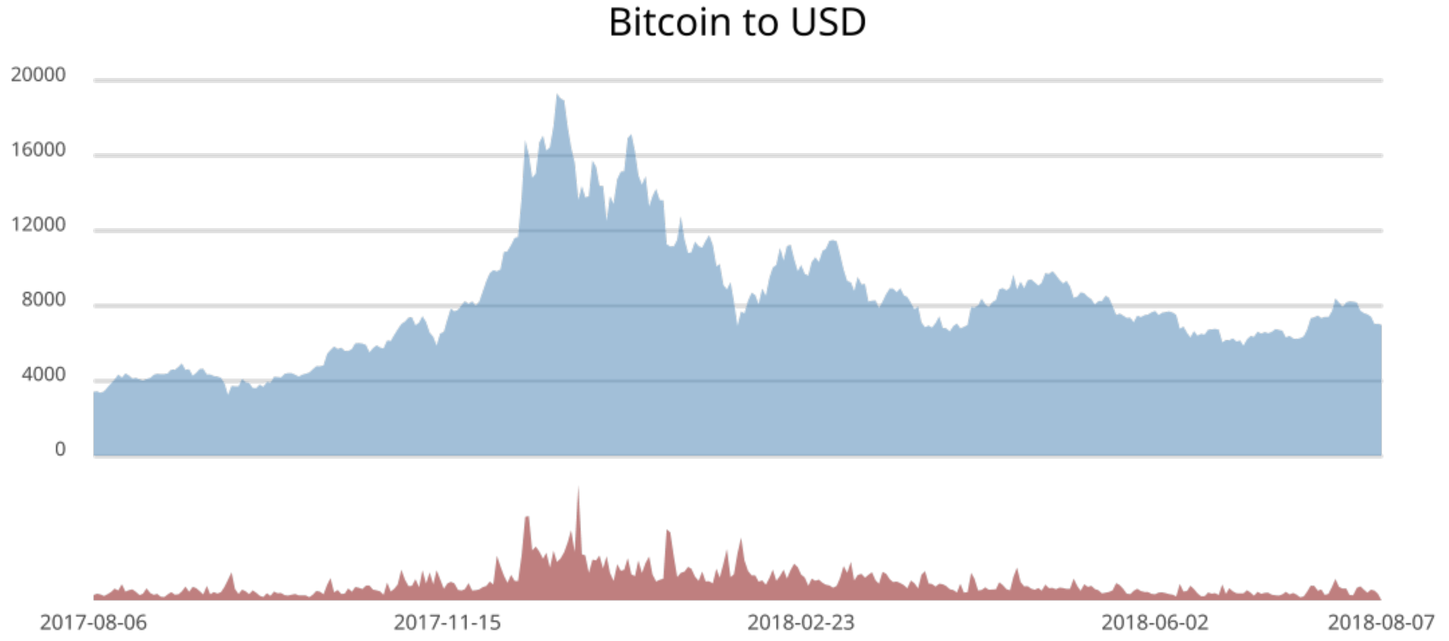
decksh → deck markup → SVG PDF PNG

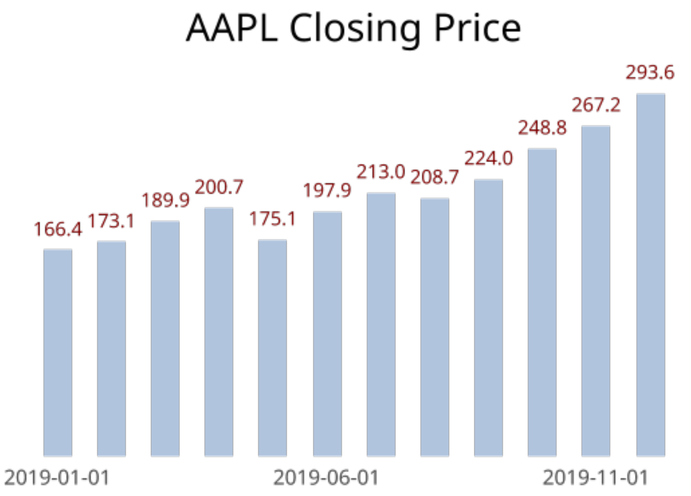
```
deck
    slide "rgb(250,250,250)" "black"
        ctext   "Deck elements" 50 90 5
        image   "follow.jpg"    70 50 640 480 50
        blist   10 75 3
            li "text, image, list"
            li "rect, ellipse, polygon"
            li "line, arc, curve"
        elist

        gy=10
        rect    15 gy 8 6             "rgb(127,0,0)"
        ellipse 27.5 gy 8 6           "rgb(0,127,0)"
        line    50 gy 60 gy
        curve   80 gy 95 30 90 gy
        arc     70 gy 10 8 0 180 0.1  "rgb(0,0,127)"
        polygon "37 37 45" "13 7 10"  "rgb(0,0,127)"

        opts="-fulldeck=f -textsize 1  -xlabel=2  -barwidth 1.5"
        dchart -left 10 -right 42 -top 42 -bottom 25 opts AAPL.d
    eslide
edeck
```

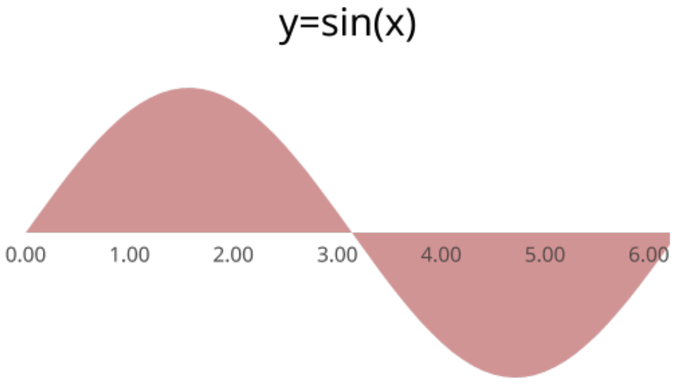
```
<deck>
<slide bg="rgb(250,250,250)" fg="black">
<text align="c" xp="50" yp="90" sp="5">Deck elements</text>
<image name="follow.jpg" xp="70" yp="50" width="640" height="480" scale="50" />
<list type="bullet" xp="10" yp="75" sp="3">
<li>text, image, list</li>
<li>rect, ellipse, polygon</li>
<li>line, arc, curve</li>
</list>
<rect xp="15" yp="10" wp="8" hp="6" color="rgb(127,0,0)" />
<ellipse xp="27.5" yp="10" wp="8" hp="6" color="rgb(0,127,0)" />
<line xp1="50" yp1="10" xp2="60" yp2="10" />
<curve xp1="80" yp1="10" xp2="95" yp2="30" xp3="90" yp3="10" />
<arc xp="70" yp="10" wp="10" hp="8" a1="0" a2="180" sp="0.1" color="rgb(0,0,127)" />
<polygon xc="37 37 45" yc="13 7 10" color="rgb(0,0,127)" />
<text xp="26.00" yp="45.60" sp="1.50" align="center" wp="0.00" font="sans" opacity="100.00"
color="black" type="">AAPL Volume</text>
<line xp1="10.00" yp1="25.00" xp2="10.00" yp2="37.46" sp="1.50" opacity="100.00"
color="lightsteelblue" />
<text xp="10.00" yp="38.46" sp="0.75" align="center" wp="0.00" font="sans" opacity="100.00"
color="rgb(127,0,0)" type="">679.9</text>
<text xp="10.00" yp="23.00" sp="0.80" align="center" wp="0.00" font="sans" opacity="100.00"
color="rgb(75,75,75)" type="">2017-09-01</text>
<line xp1="12.91" yp1="25.00" xp2="12.91" yp2="34.24" sp="1.50" opacity="100.00"
color="lightsteelblue" />
<text xp="12.91" yp="35.24" sp="0.75" align="center" wp="0.00" font="sans" opacity="100.00"
color="rgb(127,0,0)" type="">504.3</text>
...
</slide>
</deck>
```

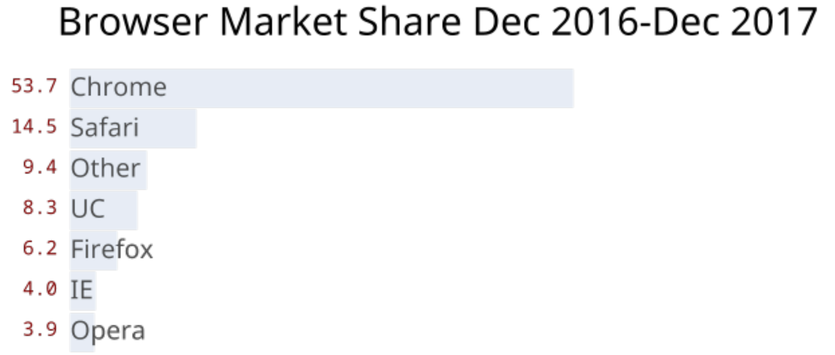


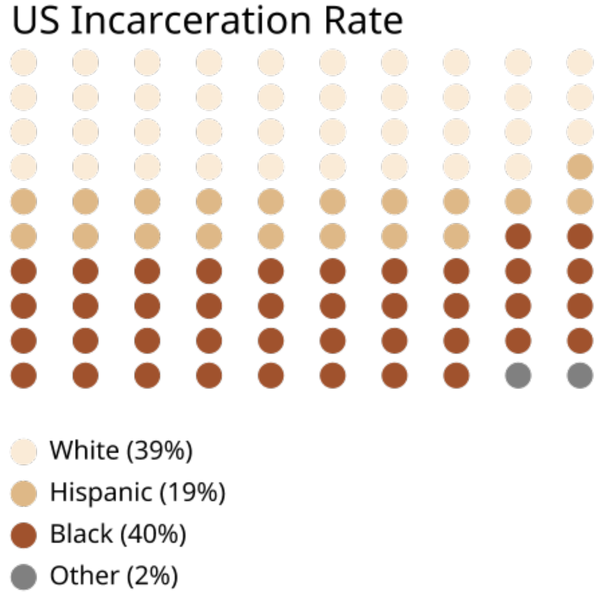
dchart: charts for deck/decksh



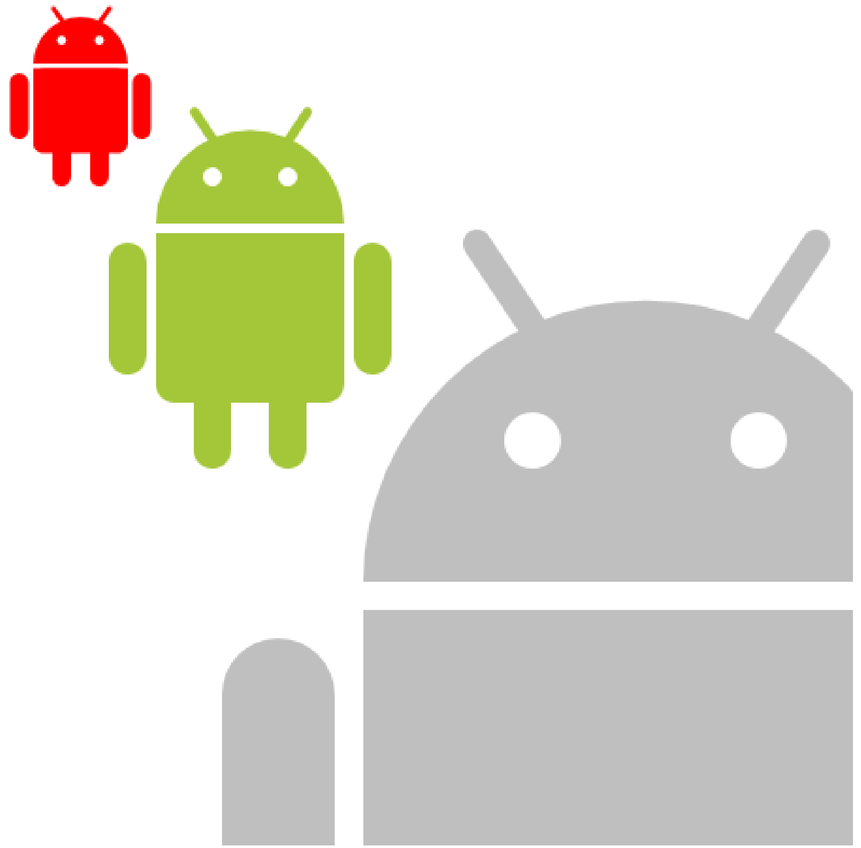


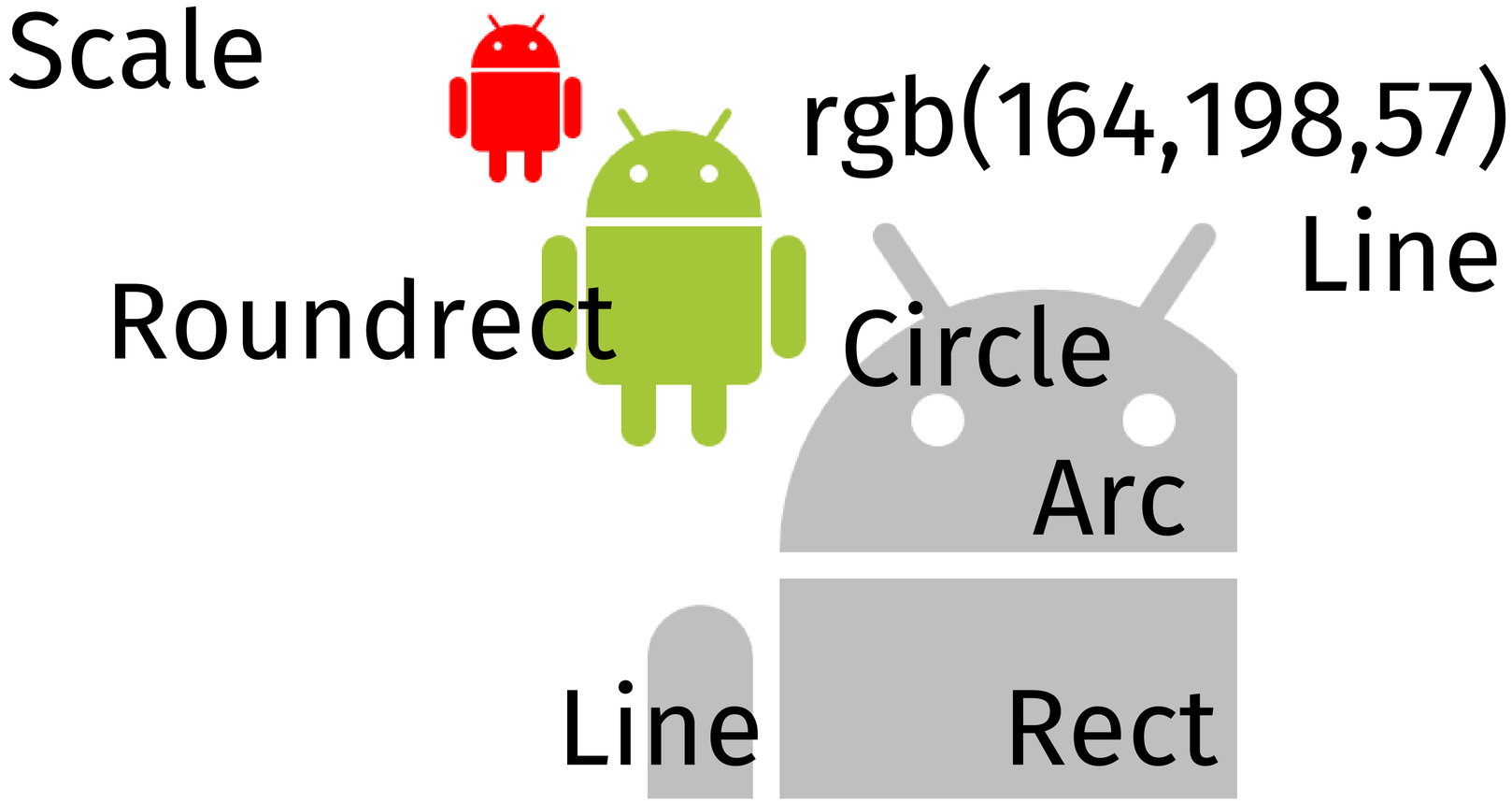






SVGo API Design

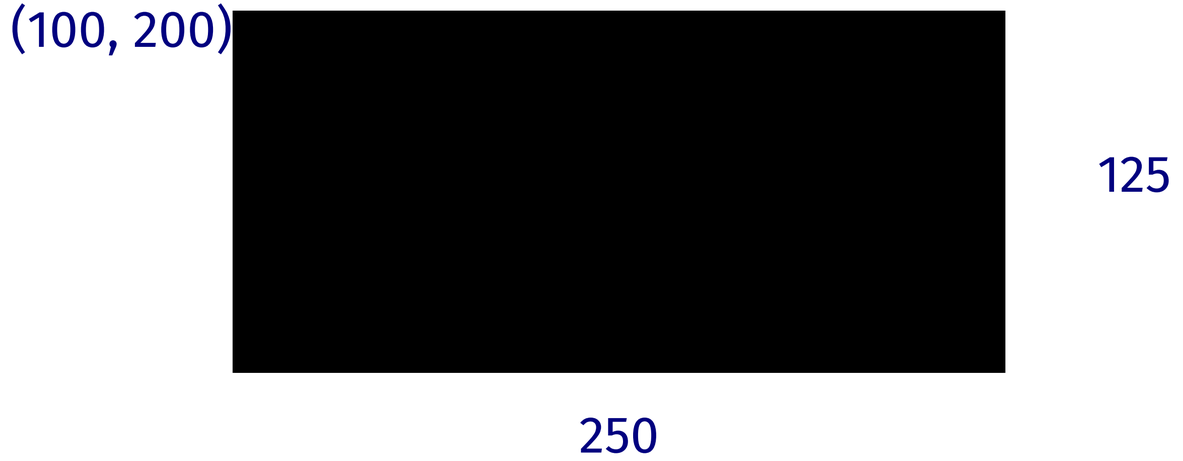



Scale
rgb(164,198,57)
Line
Roundrect
Circle
Arc
Line
Rect

Element | Arguments

Rect (100,200,250,125)

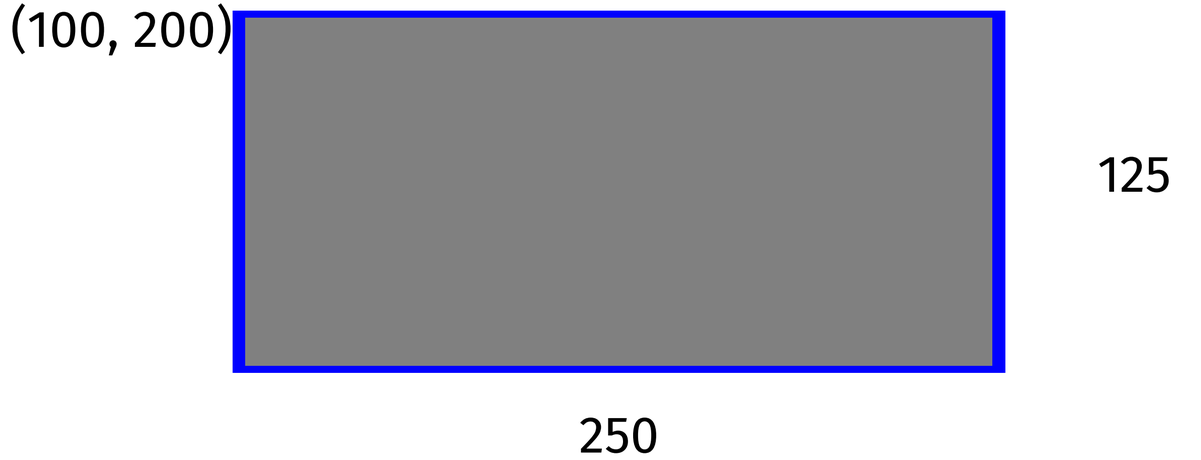
```
<rect x="100" y="200" width="250" height="125"/>
```



Element Arguments CSS Style

```
Rect (100,200,250,125,
      "fill:gray;stroke:blue")
```

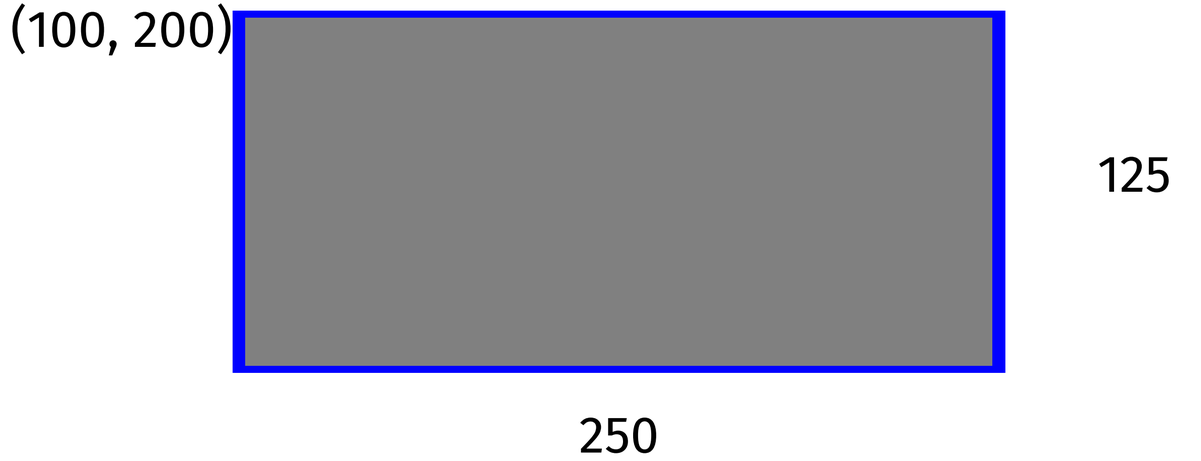
```
<rect x="100" y="200" width="250" height="125"
      style="fill:gray;stroke:blue"/>
```

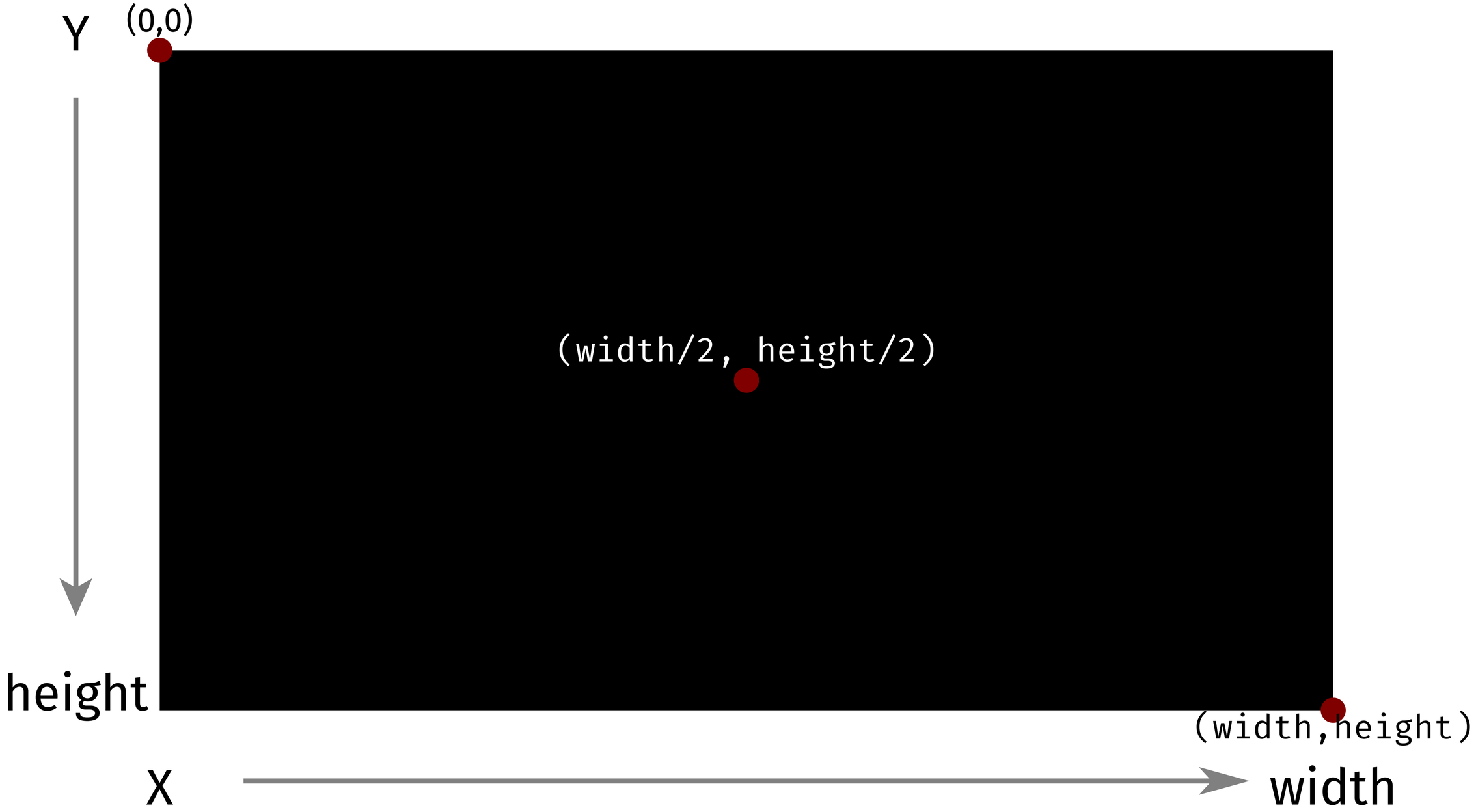


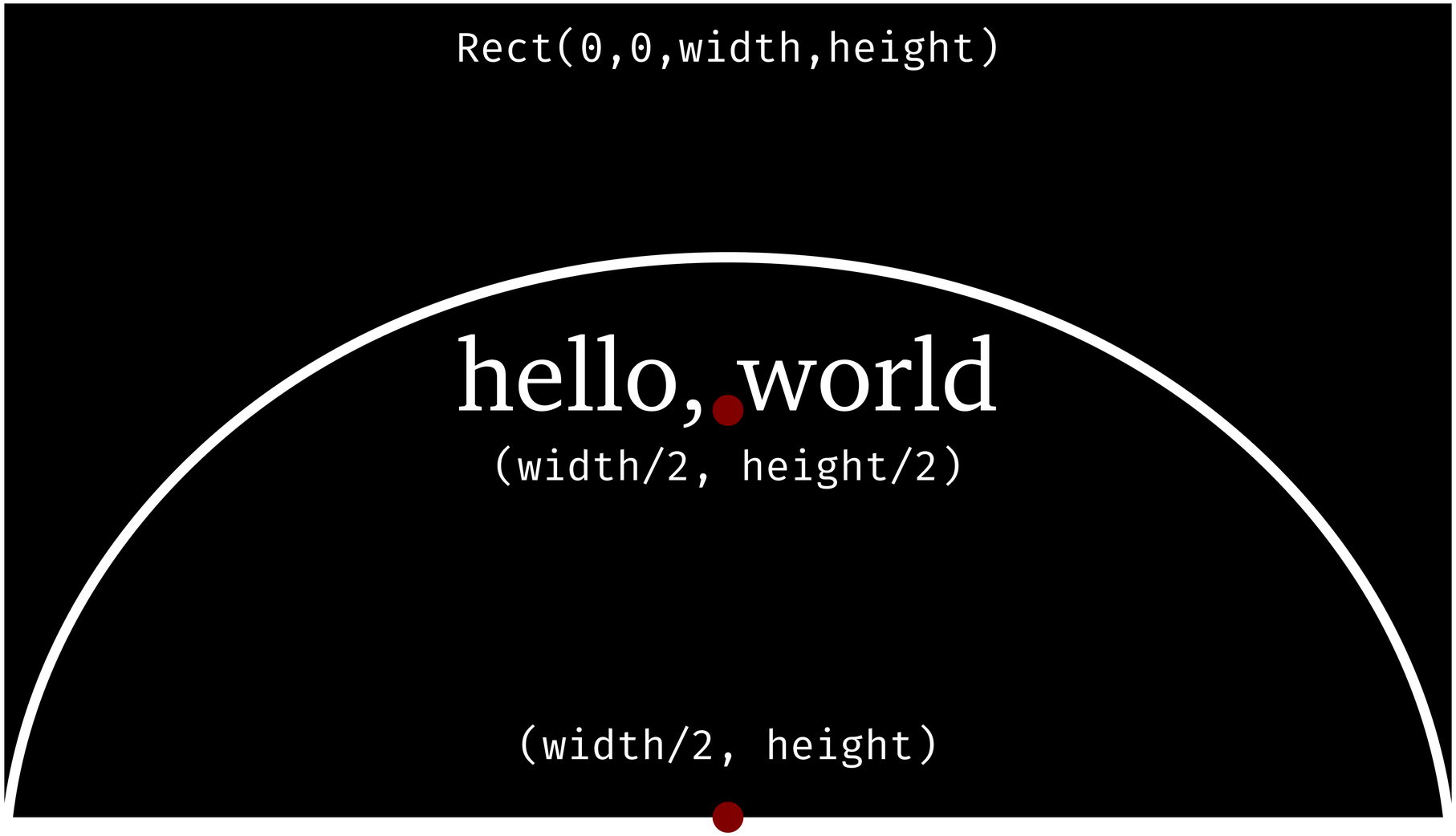
Element Arguments Attributes

```
Rect (100,200,250,125,
    `id='box'`, `fill='gray'`, `stroke='blue'`)
```

```
<rect x="100" y="200" width="250" height="125"
      id='box' fill='gray' stroke='blue'/>
```



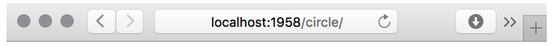

Y
(0,0)
(width/2, height/2)
height
(width,height)
X
width

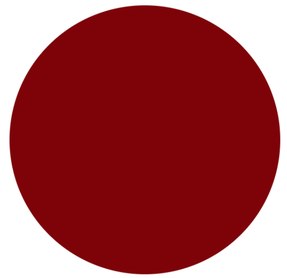

Rect(0,0,width,height)
hello, world
(width/2, height/2)
(width/2, height)

```
package main

import (
    "os"
    "github.com/ajstarks/svgo"
)

func main() {
    width := 960
    height := 540
    style := "fill:white;font-size:60pt;text-anchor:middle"
    canvas := svg.New(os.Stdout)
    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height)
    canvas.Circle(width/2, height, width/2, "fill:rgb(44,77,232)")
    canvas.Text(width/2, height/2, "hello, world", style)
    canvas.End()
}
```

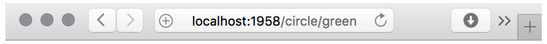
```
const defaultstyle = "fill:rgb(127,0,0)"

func circle(w http.ResponseWriter, req *http.Request) {
    w.Header().Set("Content-Type", "image/svg+xml")
    s := svg.New(w)
    s.Start(500, 500)
    s.Title("Circle")
    s.Circle(250, 250, 125, shapestyle(req.URL.Path))
    s.End()
}

func shapestyle(path string) string {
    i := strings.LastIndex(path, "/") + 1
    if i > 0 && len(path[i:]) > 0 {
        return "fill:" + path[i:]
    }
    return defaultstyle
}
```

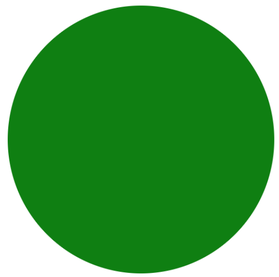
/circle





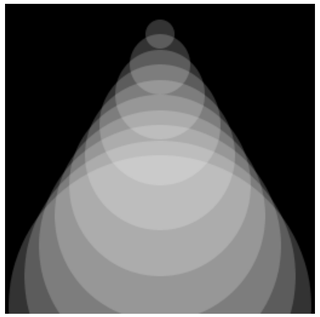
/circle/green



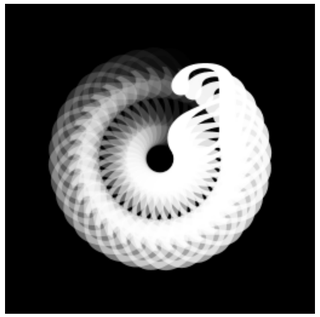


https://ajstarks.org:1958/

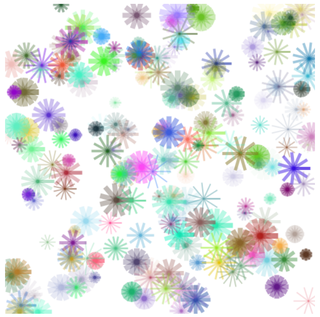
clock



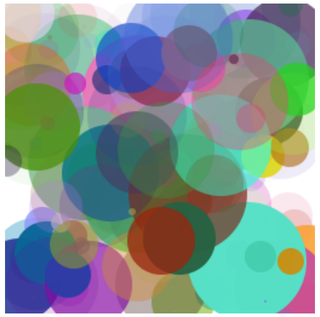
funnel



rotext

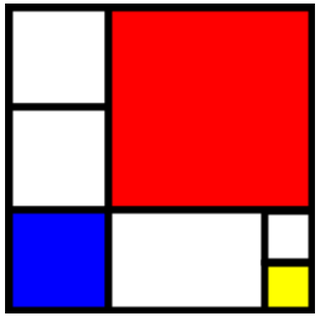


flower

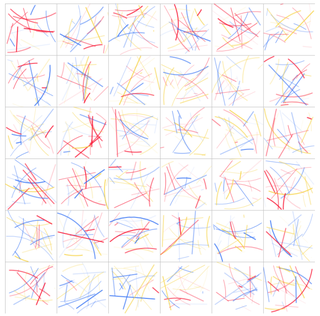


rshape

cube



mondrian

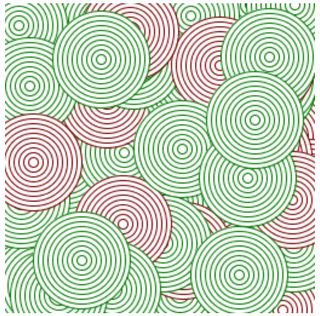


lewitt

face

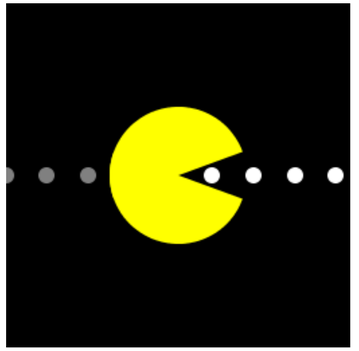
pacman

tux



concentric

/pacman/?angle=20



/pacman/?angle=40

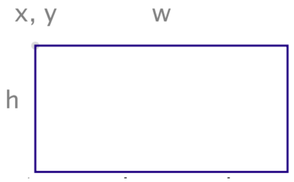
/pacman/?angle=70

API Reference

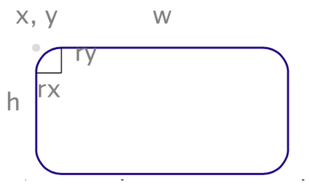
SVGo Method Categories

- Shapes
- Paths, Lines and Curves
- Text and Images
- Colors and Gradients
- Transforms
- Animation
- Filter Effects
- Starting and Ending
- Metadata
- Utility
- Groups and Definitions
- Markers and Patterns
- Links
- Masking and Clipping

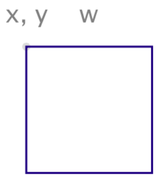
Shapes



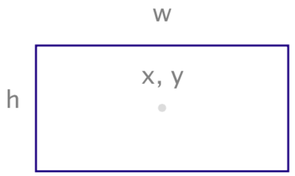
Rect(x, y, w, h int, s string...)



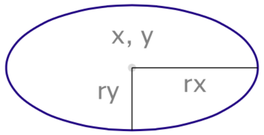
Roundrect(x, y, w, h, rx, ry int, s string...)



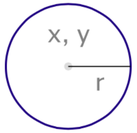
Square(x, y, l int, s string...)



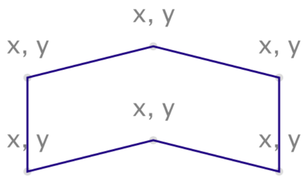
CenterRect(x, y, w, h int, s string...)



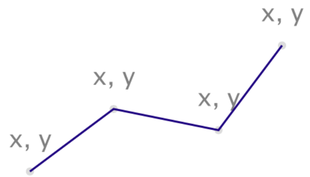
Ellipse(x, y, w, h int, s string...)



Circle(x, y, r int, s string...)



Polygon(x, y []int, s string...)

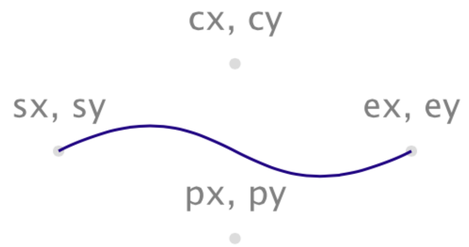


Polyline(x, y []int, s string...)

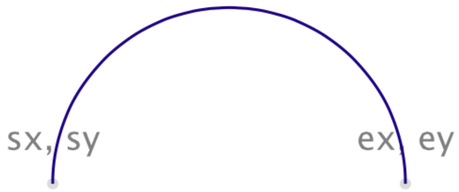
Line(x1, y2, x2, y2 int, s string...)

Curves and Paths

Bezier(sx, sy, cx, cy, ex, ey int, s string...)



QBez(sx, sy, cx, cy, px, py, ex, ey int, s string...)



Arc(sx, ax, ay, r int, dir, large bool, ex, ey int, s string...)

Path(d string, s string...)

Text and Images

hello, this is SVG

x, y

Text(x, y int, s string, s ...string)

Textpath(t string, pathid string, s ...string)

New(w io Writer)
Start(w, h int, options ...string)/End()
Startview(w, h, minx, miny, vw, vh int)
Group(s ...string)/Gend()
Gstyle(s string)/Gend()

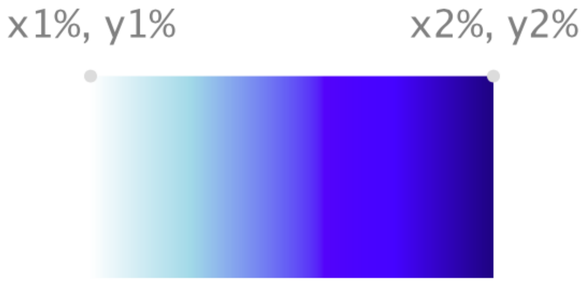
Textlines(x, y int, s []string, size, spacing int, fill, align string)

x, y w

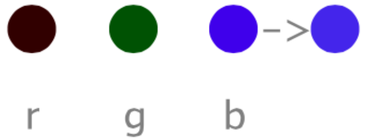
h

Image(x, y, w, h int, link string, s ...string)

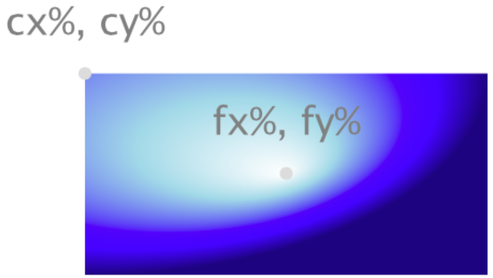
Gradients and Colors



LinearGradient(id string, x1, y1, x2, y2 uint8, sc []Offcolor)



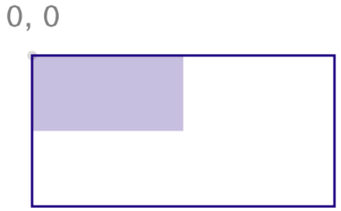
RGB(r, g, b int)



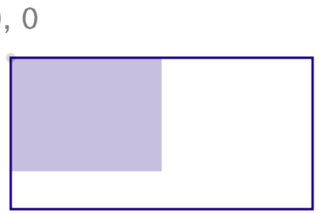
RadialGradient(id string, cx, cy, r, fx, fy uint8, sc []Offcolor)

RGBA(r, g, b int, a float64)

Transforms

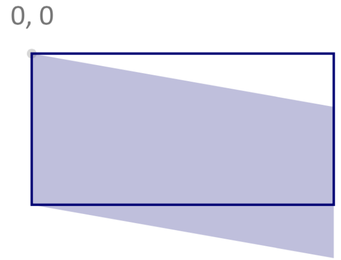


Scale(n float64)

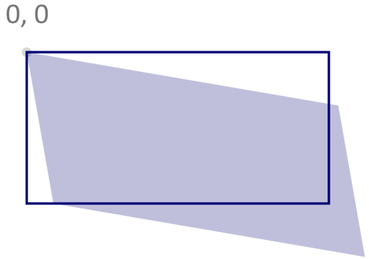


ScaleXY(dx, dy float64)

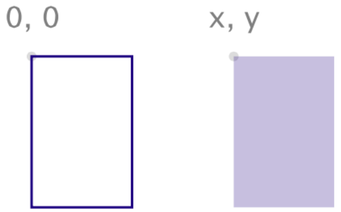
SkewX(a float64)



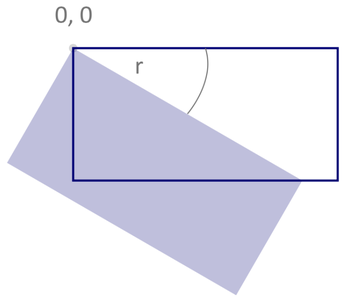
SkewY(a float64)



SkewXY(ax, ay float64)

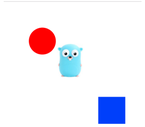


Translate(x, y int)

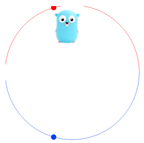


Rotate(a float64)

Animation

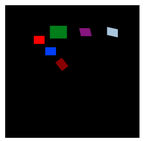


Animate(link, attr string, from, to int, duration float64, repeat int, s ...string)



AnimateMotion(link, path string, duration float64, repeat int, s ...string)

AnimateTranslate(link string, fx, fy, tx, ty int, duration float64, repeat int, s ...string)



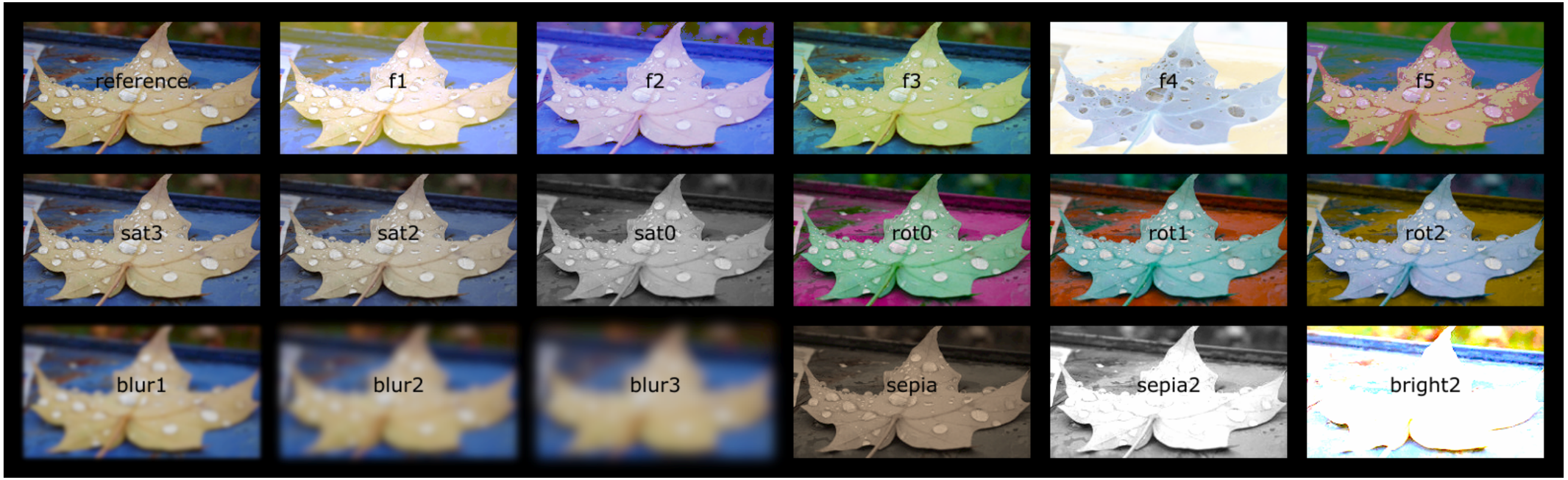
AnimateRotate(link string, fs, fc, fe, ts, tc, te int, duration float64, repeat int, s ...string)

AnimateScale(link string, from, to, duration float64, repeat int, s ...string)

AnimateSkewX(link string, from, to, duration float64, repeat int, s ...string)

AnimateSkewY(link string, from, to, duration float64, repeat int, s ...string)

Filter Effects

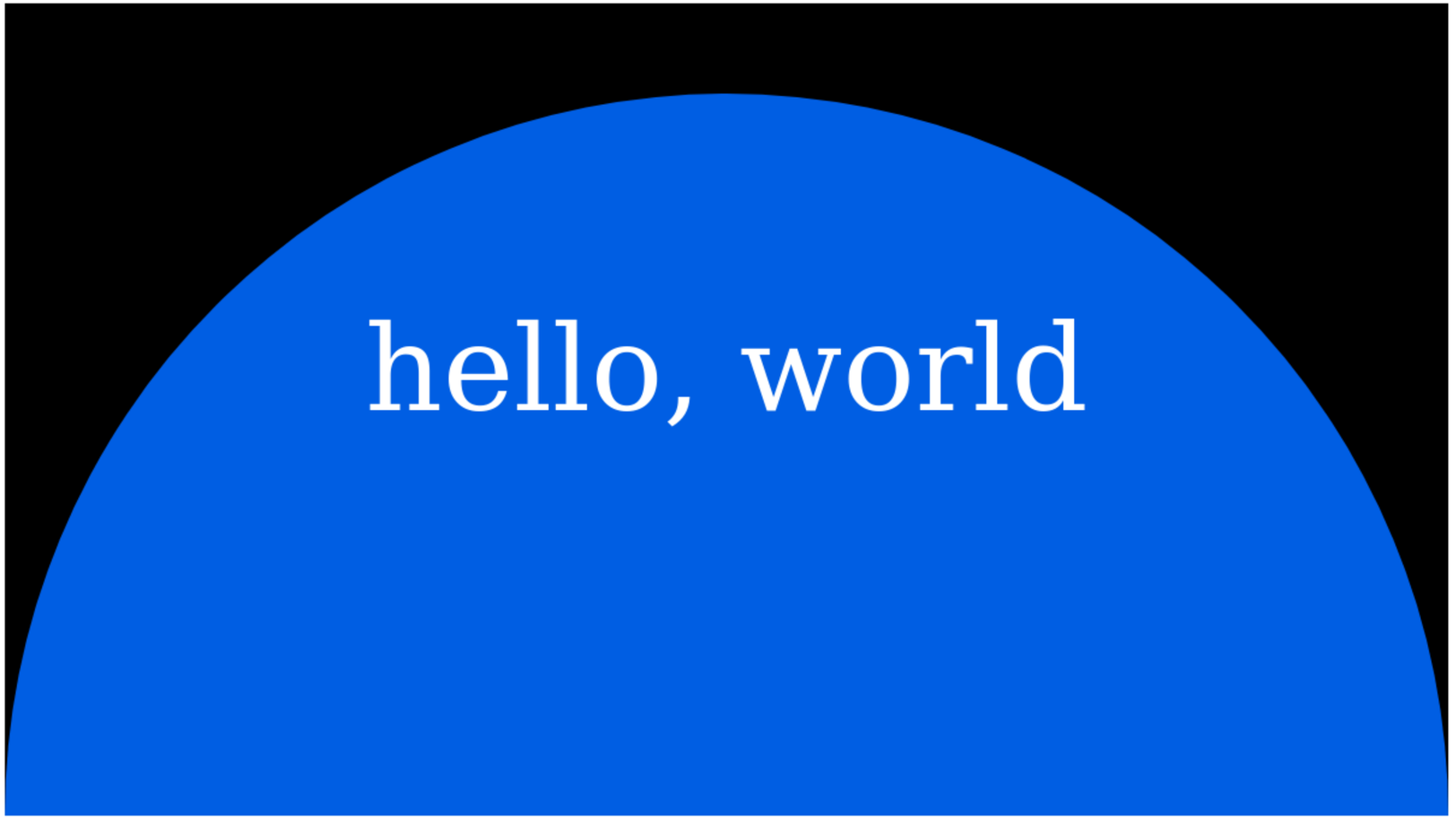


Sketching with code

svgplay: SVGo sketching

localhost:1999/code/hwsvg.go

```
package main

import (
   "os"
   "github.com/ajstarks/svgo"
)

func main() {
   width := 960
   height := 540
   style := "fill:white;font-size:60pt;text-anchor:middle"
   canvas := svg.New(os.Stdout)
   canvas.Start(width, height)
   canvas.Rect(0, 0, width, height)
   canvas.Circle(width/2, height, width/2, "fill:rgb(44,77,232)")
   canvas.Text(width/2, height/2, "hello, world", style)
   canvas.End()
}
```

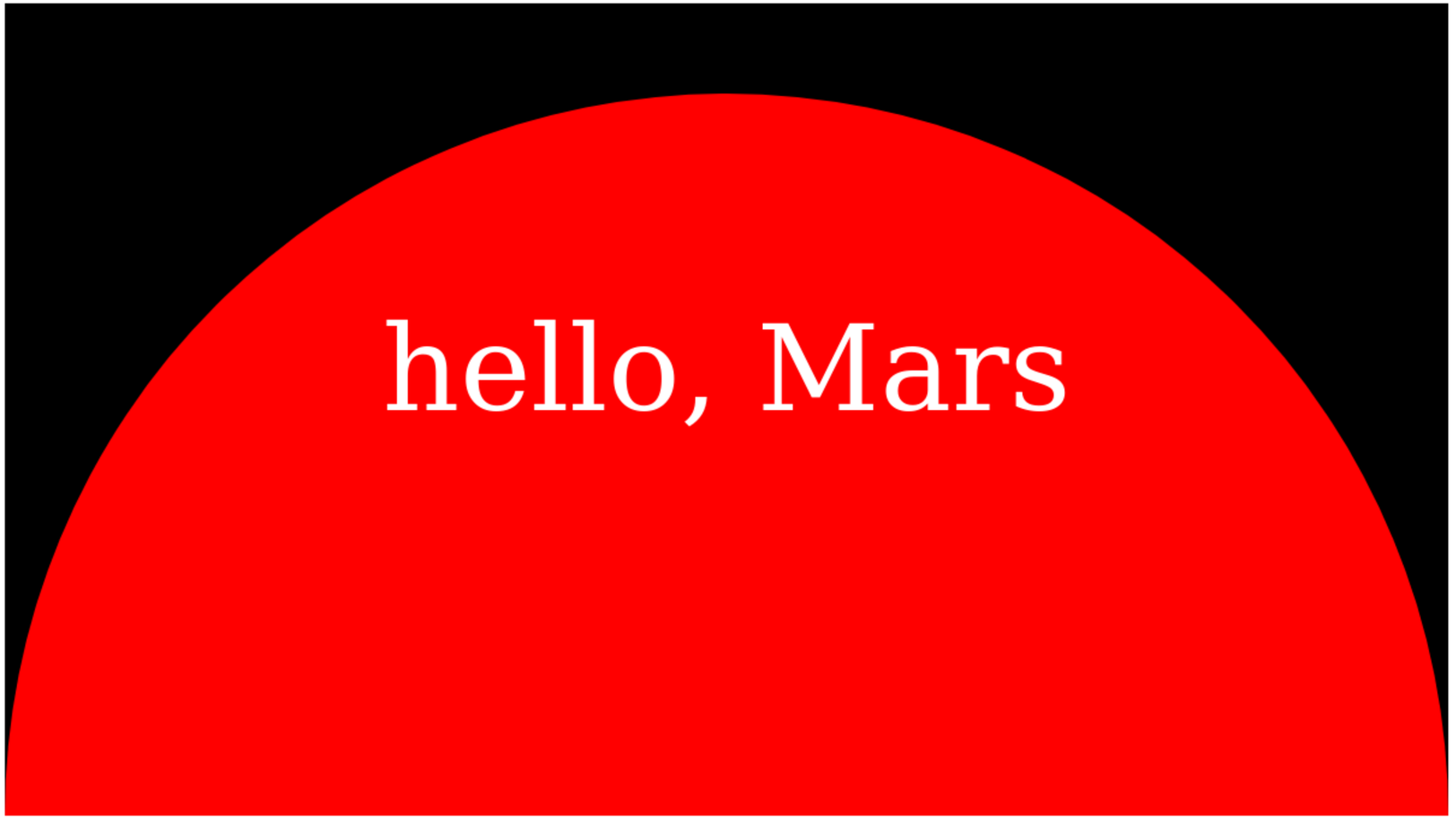


(Shift-Enter to compile and run.) ☐ *Compile and run after each keystroke*

svgplay: SVGo sketching

localhost:1999/code/hwsvg.go

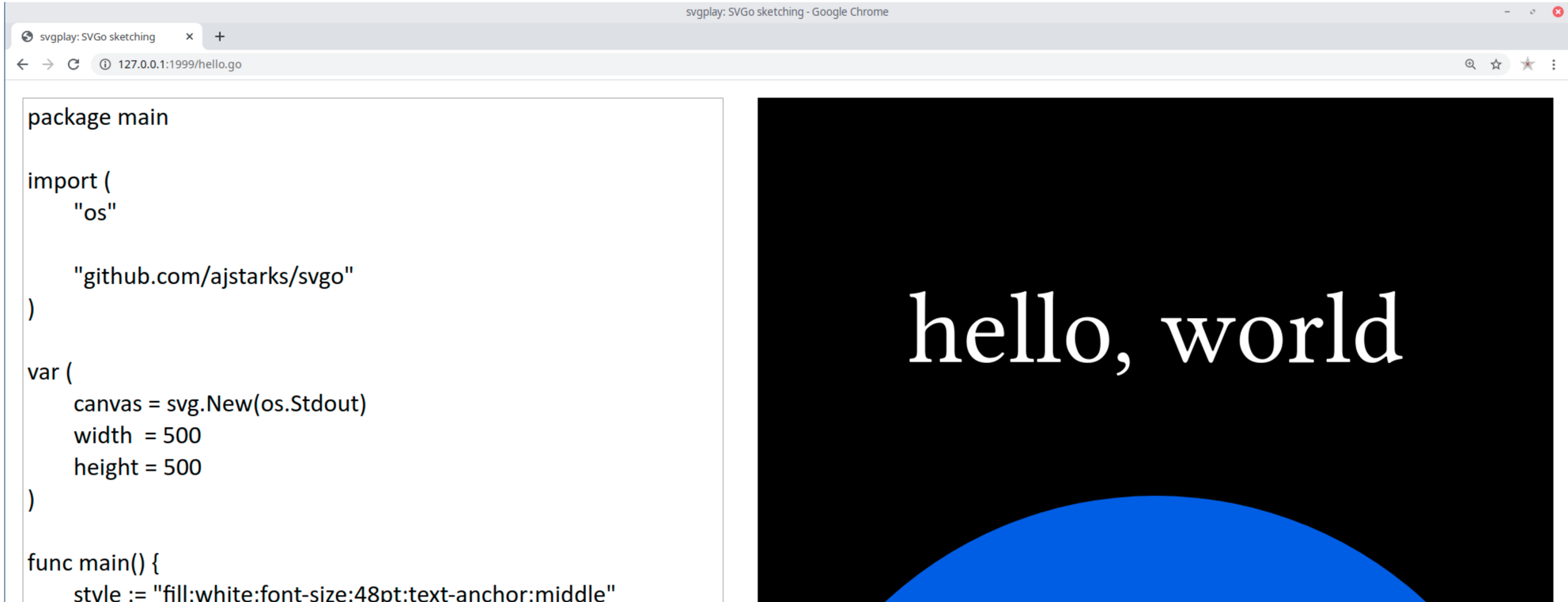
```
package main

import (
   "os"
   "github.com/ajstarks/svgo"
)

func main() {
   width := 960
   height := 540
   style := "fill:white;font-size:60pt;text-anchor:middle"
   canvas := svg.New(os.Stdout)
   canvas.Start(width, height)
   canvas.Rect(0, 0, width, height)
   canvas.Circle(width/2, height, width/2, "fill:red")
   canvas.Text(width/2, height/2, "hello, Mars", style)
   canvas.End()
}
```



(Shift-Enter to compile and run.) *Compile and run after each keystroke*

Running SVGo

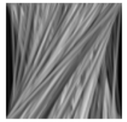
```
$ git clone https://github.com/ajstarks/svgo-workshop
$ cd svgo-workshop/code/svgplay-samples
$ svglay
svgplay 2020/02/28 17:58:43 ☠ ☠ ☠ Warning: this server allows a client connecting
to 127.0.0.1:1999 to execute code on this computer ☠ ☠ ☠
```



SVGo Examples

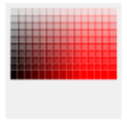
hello.go

colorhash.go


rl.go

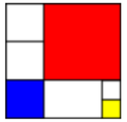

concentric.go

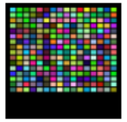

concentric2.go

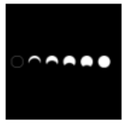

cgrid.go


diag.go


shining.go


mondrian.go


richter.go


eclipse.go


pacman.go


pyramid.go


cloud.go


color-clouds.go


creepy.go


d4h.go


sunearth.go

conception.go

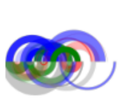

conception2.go


randbox.go


randspot.go

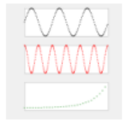

cube.go


schotter.go


randarc.go


recurse.go

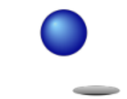

clock.go


plotfunc.go

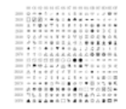

therm.go

pattern.go

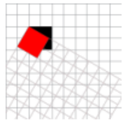

marker.go


gradient.go

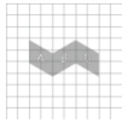
fontrange.go


uni.go


lorem.go


rotate.go


rotext.go


skewabc.go


gear.go


star.go

starx.go

swoosh.go

textpath.go

imfade.go

fe.go

artist.go

go.go

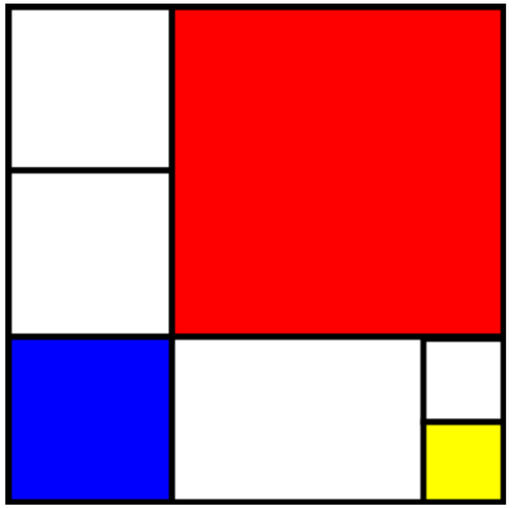
am.go

code/svgplay-samples

```
package main

import (
    "fmt"
    "github.com/ajstarks/svgo"
    "os"
)

func main() {
    xp := []int{50, 70, 70, 50, 30, 30}
    yp := []int{40, 50, 75, 85, 75, 50}
    xl := []int{0, 0, 50, 100, 100}
    yl := []int{100, 40, 10, 40, 100}
    bgcolor := "rgb(227,78,25)"
    bkcolor := "rgb(153,29,40)"
    stcolor := "rgb(65,52,44)"
    stwidth := 12
    stylefmt := "stroke:%s;stroke-width:%d;fill:%s"
    canvas := svg.New(os.Stdout)
    width, height := 500, 500
    canvas.Start(width, height)
    canvas.Def()
    canvas.Gid("unit")
    canvas.Polyline(xl, yl, "fill:none")
    canvas.Polygon(xp, yp)
    canvas.Gend()
    canvas.Gid("runit")
    canvas.TranslateRotate(150, 180, 180)
    canvas.Use(0, 0, "#unit")
    canvas.Gend()
    canvas.Gend()
    canvas.DefEnd()
    canvas.Rect(0, 0, width, height, "fill:"+bgcolor)
    canvas.Gstyle(fmt.Sprintf(stylefmt, stcolor, stwidth, bkcolor))
    for y := 0; y < height; y += 130 {
        for x := -50; x < width; x += 100 {
            canvas.Use(x, y, "#unit")
            canvas.Use(x, y, "#runit")
        }
    }
    canvas.Gend()
    canvas.End()
}
```

shining.go [8]

```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

var (
    canvas = svg.New(os.Stdout)
    width  = 500
    height = 500
)

// Piet Mondrian - Composition in Red, Blue, and Yellow
func main() {
    w3 := width / 3
    w6 := w3 / 2
    w23 := w3 * 2
    canvas.Start(width, height)
    canvas.Gstyle("stroke:black;stroke-width:6")
    canvas.Rect(0, 0, w3, w3, "fill:white")
    canvas.Rect(0, w3, w3, w3, "fill:white")
    canvas.Rect(0, w23, w3, w3, "fill:blue")
    canvas.Rect(w3, 0, w23, w23, "fill:red")
    canvas.Rect(w3, w23, w23, w3, "fill:white")
    canvas.Rect(width-w6, height-w3, w3-w6, w6, "fill:white")
    canvas.Rect(width-w6, height-w6, w3-w6, w6, "fill:yellow")
    canvas.Gend()
    canvas.Rect(0, 0, width, height, "fill:none;stroke:black;stroke-width:12")
    canvas.End()
}
```

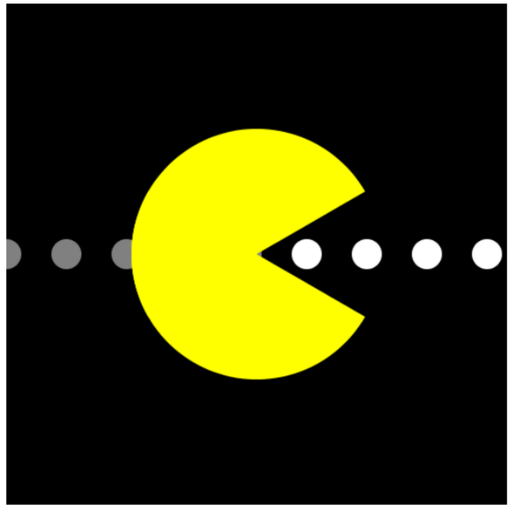


mondrian.go [9]

```
package main

import (
    "github.com/ajstarks/svgo"
    "os"
)

var (
    canvas = svg.New(os.Stdout)
    width  = 500
    height = 500
)

func main() {

    blues := "stroke:blue"
    reds := "stroke:red"
    greens := "stroke:green"
    organges := "stroke:orange"

    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height, "fill:white")

    canvas.Gstyle("fill:none;stroke-opacity:0.5;stroke-width:35;stroke-linecap:round")
    // g
    canvas.Arc(20, 200, 30, 30, 0, false, true, 220, 200, blues)
    canvas.Arc(20, 200, 30, 30, 0, false, false, 220, 200, reds)
    canvas.Line(220, 100, 220, 300, greens)
    canvas.Arc(20, 320, 30, 30, 0, false, false, 220, 300, organges)
    // o
    canvas.Arc(280, 200, 30, 30, 0, false, true, 480, 200, reds)
    canvas.Arc(280, 200, 30, 30, 0, false, false, 480, 200, blues)
    canvas.Gend()
    canvas.End()
}
```

go.go [47]

```
package main

import (
    "github.com/ajstarks/svgo"
    "os"
)

var (
    canvas = svg.New(os.Stdout)
    width  = 500
    height = 500
)

func main() {
    angle, cx, cy := 30.0, width/2, height/2
    r := width / 4
    p := r / 8

    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height)
    canvas.Gstyle("fill:white")
    for x := 0; x < 100; x += 12 {
        if x < 50 {
            canvas.Circle((width*x)/100, cy, p, "fill-opacity:0.5")
        } else {
            canvas.Circle((width*x)/100, cy, p, "fill-opacity:1")
        }
    }
    canvas.Gend()

    canvas.Gstyle("fill:yellow")
    canvas.TranslateRotate(cx, cy, -angle)
    canvas.Arc(-r, 0, r, r, 30, false, true, r, 0)
    canvas.Gend()

    canvas.TranslateRotate(cx, cy, angle)
    canvas.Arc(-r, 0, r, r, 30, false, false, r, 0)
    canvas.Gend()

    canvas.Gend()
    canvas.End()
}
```



pacman.go [12]

Read/Parse/Draw Pattern

Read

from the data source

Parse

into data structures

Draw

from the structures

Data

Picture

Read Parse Draw

https://api.nytimes.com/svc/topstories/v2/food.json?api-key=...


```
{
  "status": "OK",
  "copyright": "Copyright (c) 2019 The New York Times Company. All Rights Reserved.",
  "section": "food",
  "last_updated": "2019-06-26T07:57:20-04:00",
  "num_results": 26,
  "results": [
    {
      "section": "Food",
      "subsection": "",
      "title": "Thomas Keller Brings Country Club Cuisine to the City",
      "abstract": "At TAK Room at Hudson Yards, the chef salutes the strait-laced, spice-free food that rich Americans used to feed on.",
      "url": "https://www.nytimes.com/2019/06/25/dining/tak-room-review-thomas-keller.html",
      "byline": "By PETE WELLS",
      "item_type": "Article",
      "updated_date": "2019-06-25T11:59:42-04:00",
      "created_date": "2019-06-25T11:59:42-04:00",
      "published_date": "2019-06-25T11:59:42-04:00",
      "material_type_facet": "",
      "kicker": "",
      "des_facet": [
        "Restaurants"
      ],
      "org_facet": [
        "TAK Room (Manhattan, NY, Restaurant)"
      ],
      "per_facet": [
        "Keller, Thomas"
      ],
      "geo_facet": [
        "Hudson Yards (Manhattan, NY)"
      ],
      "multimedia": [
        {
          "url": "https://static01.nyt.com/images/2019/06/26/dining/25REST-slide-H6ZN/25REST-slide-H6ZN-thumbStandard.jpg",
          "format": "Standard Thumbnail",
          "height": 75,
          "width": 75,
          "type": "image",
          "subtype": "photo",
          "caption": "",
          "copyright": "Jeenah Moon for The New York Times"
        },
        {
          "url": "https://static01.nyt.com/images/2019/06/26/dining/25REST-slide-H6ZN/25REST-slide-H6ZN-thumbLarge.jpg",
          "format": "thumbLarge",
          "height": 150,
```


Imports

```
package main

import (
    "encoding/json"
    "flag"
    "fmt"
    "io"
    "net/http"
    "os"
    "time"

    "github.com/ajstarks/svgo"
)
```

Constants

```
// API Info and formats
const (
  NYTAPIkey = "HxTV50RYD7LyGzHxUy7XCThV3kQ3HYvk"
  NYTfmt    = "https://api.nytimes.com/svc/topstories/v2/%s.json?api-key=%s"
  style     = "font-size:9pt;font-family:sans-serif;text-anchor:middle;fill:white"
  errfmt    = "unable to get network data for %s (%s)"
  headfmt   = "New York Times Top Stories for %s"
  datefmt   = "Monday Jan 2, 2006"
  usage     = `section choices:
arts, automobiles, books, business, fashion,
food, health, home, insider, magazine, movies,
national, nyregion, obituaries, opinion, politics,
realestate, science, sports, sundayreview, technology,
theater, tmagazine, travel, upshot, world`
)
```

Data Structures

```
// NYTStories is the headline info from the New York Times
type NYTStories struct {
    StoryCount int      `json:"num_results"`
    Results    []result `json:"results"`
}
type result struct {
    Title      string       `json:"title"`
    URL        string       `json:"url"`
    Multimedia []multimedia `json:"multimedia"`
}
type multimedia struct {
    URL    string `json:"url"`
    Width  int    `json:"width"`
    Height int    `json:"height"`
    Format string `json:"format"`
}
```

Main

```
func main() {
    var section = flag.String("s", "home", usage)
    var nstories = flag.Int("n", 25, "number of stories")
    flag.Parse()

    width, height := 1000, (*nstories/5)*150
    canvas := svg.New(os.Stdout)
    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height)
    nytStories(canvas, width, *section, *nstories)
    canvas.End()
}
```

Network

```
// netread derefernces a URL, returning the Reader, with an error
func netread(url string) (io.ReadCloser, error) {
    client := &http.Client{Timeout: 30 * time.Second}
    resp, err := client.Get(url)
    if err != nil {
        return nil, err
    }
    if resp.StatusCode != http.StatusOK {
        return nil, fmt.Errorf(errfmt, url, resp.Status)
    }
    return resp.Body, nil
}
```

Read and Decode

```
// nytStories retrieves Top Stories data
// from the New York Times API, decodes and displays it.
func nytStories(canvas *svg.SVG, width int, section string, n int) {
    r, err := netread(fmt.Sprintf(NYTfmt, section, NYTAPIkey))
    if err != nil {
        fmt.Fprintf(os.Stderr, "headline read error: %v\n", err)
        return
    }
    defer r.Close()
    var data NYTStories
    if err = json.NewDecoder(r).Decode(&data); err != nil {
        fmt.Fprintf(os.Stderr, "decode: %v\n", err)
        return
    }
    drawStories(canvas, width, data, n)
}
```

Drawing (setup)

```
// drawStories walks the result structure,
// displaying title and thumbnail in a grid
func drawStories(canvas *svg.SVG, width int, data NYTStories, n int) {
    top, left := 100, 150
    titley := top - 50
    x, y := left, top
    ts := fmt.Sprintf(headfmt, time.Now().Format(datefmt))
    if n > data.StoryCount {
        n = data.StoryCount
    }
```

Drawing

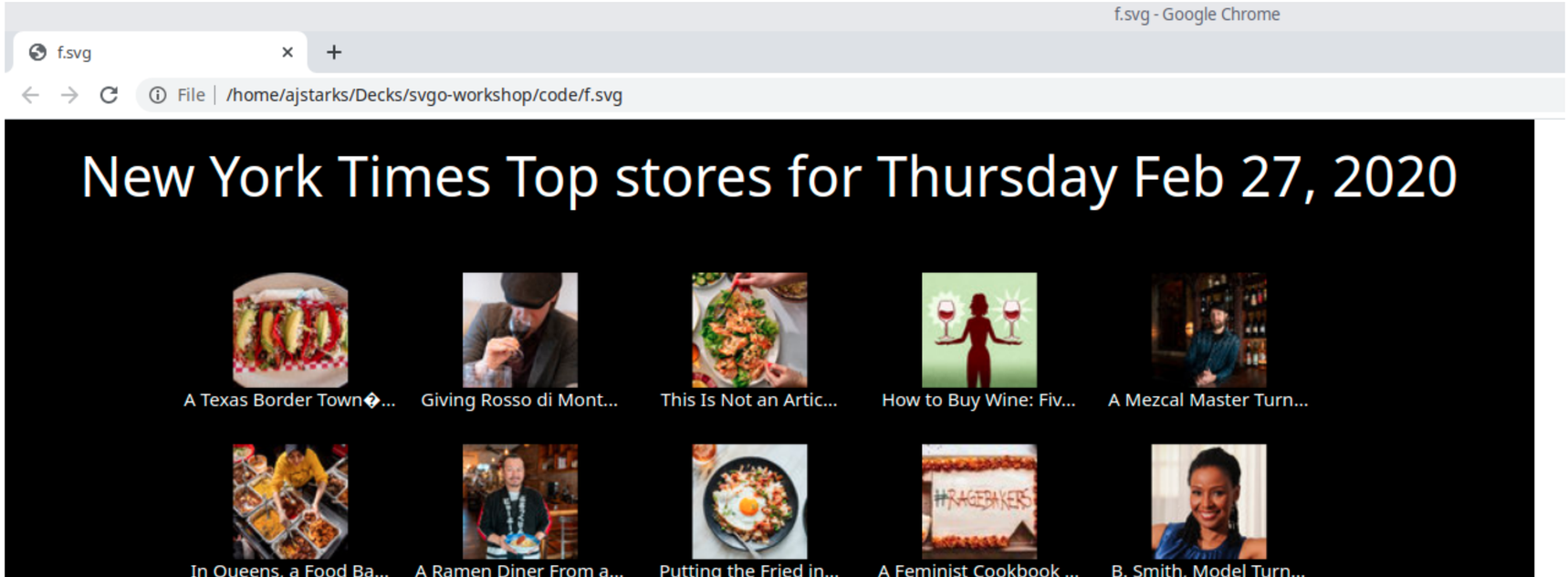
```
    canvas.Gstyle(style)
    canvas.Text(width/2, titley, ts, "font-size:300%")
    for i := 0; i < n; i++ {
        d := data.Results[i]
        tw, th, imagelink := imageinfo(d)
        if i > 0 && i%5 == 0 {
            x = left
            y += th + (th / 2)
        }
        canvas.Link(d.URL, d.Title)
        canvas.Image(x, y, tw, th, imagelink)
        canvas.Text(x+tw/2, y+th+12, struncate(d.Title, 20))
        canvas.LinkEnd()
        x += tw * 2
    }
    canvas.Gend()
}
```

Helpers

```
// struncate truncates a string with ellipsis
func struncate(s string, n int) string {
    if len(s) <= n {
        return s
    }
    return s[0:n] + "..."
}

// imageinfo returns the thumbnail image information
func imageinfo(data result) (int, int, string) {
    for _, m := range data.Multimedia {
        if m.Format == "Standard Thumbnail" {
            return m.Width, m.Height, m.URL
        }
    }
    return 75, 75, ""
}
```

Running nytsvg

```
$ git clone https://github.com/ajstarks/svgo-workshop
$ cd svgo-workshop/code
$ go run nytsvg.go -s food
```



Thank You

https://github.com/ajstarks/svgo-workshop

Anthony Starks
@ajstarks