



Workshop



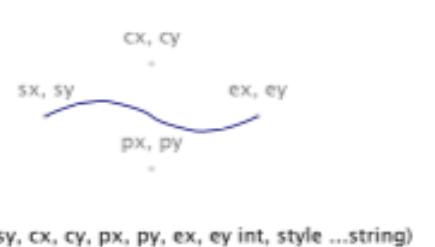
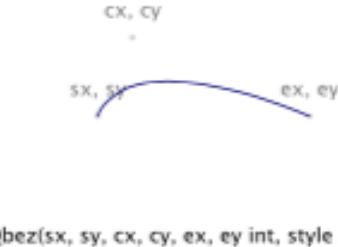
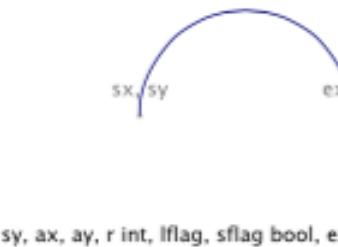
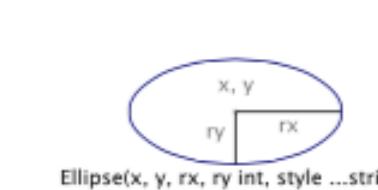
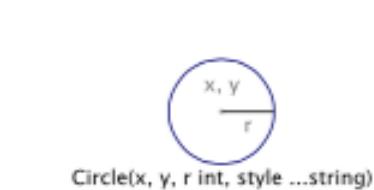
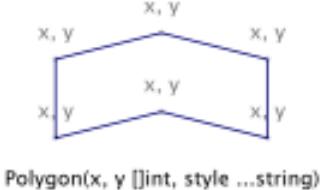
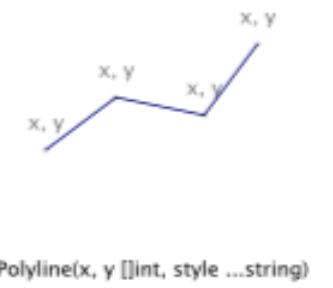
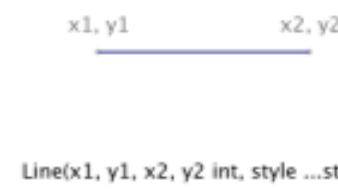
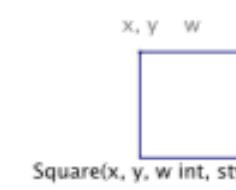
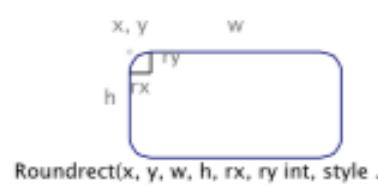
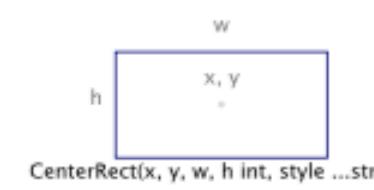
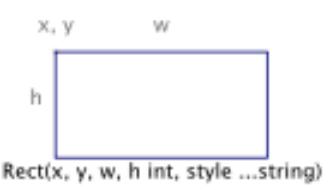
Scalable Vector Graphics



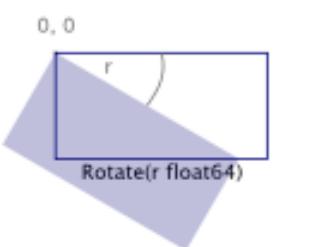
io.Writer

SVG Go Library

github.com/ajstarks/svg



\$ go get github.com/ajstarks/svg



hello, this is SVG

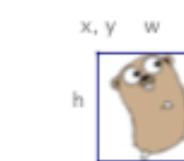
It's time to draw text along
dandy

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

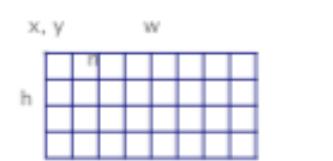
Textpath(s, pathid string, style ...string)



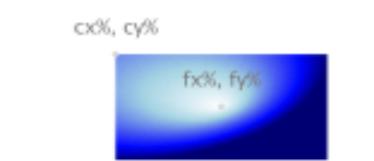
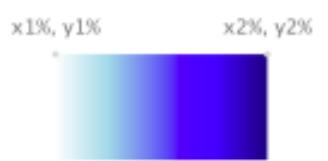
Path(s string, style ...string)



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)

RGB(r, g, b int)

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

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

New(w io Writer)
Start(w, h int, options ...string)/End()
Startview(w, h, minx, miny, vw, vh int)
Groups(...string)/End()
Gstyle(s string)/End()
Gtransfrom(s string)/End()
Gid(id string)/End()
ClipPath(s ...string)/ClipEnd()
Defn()/DefEnd()
Marker() /MarkerEnd()
Pattern() /PatternEnd()
Desc(s string)
Title(s string)
Script(type, data ...string)
Mask(id string, x,y,w,h int, style ...string)/MaskEnd()
Link(href string, title string)/LinkEnd()
Use(x int, y int, link string, style ...string)

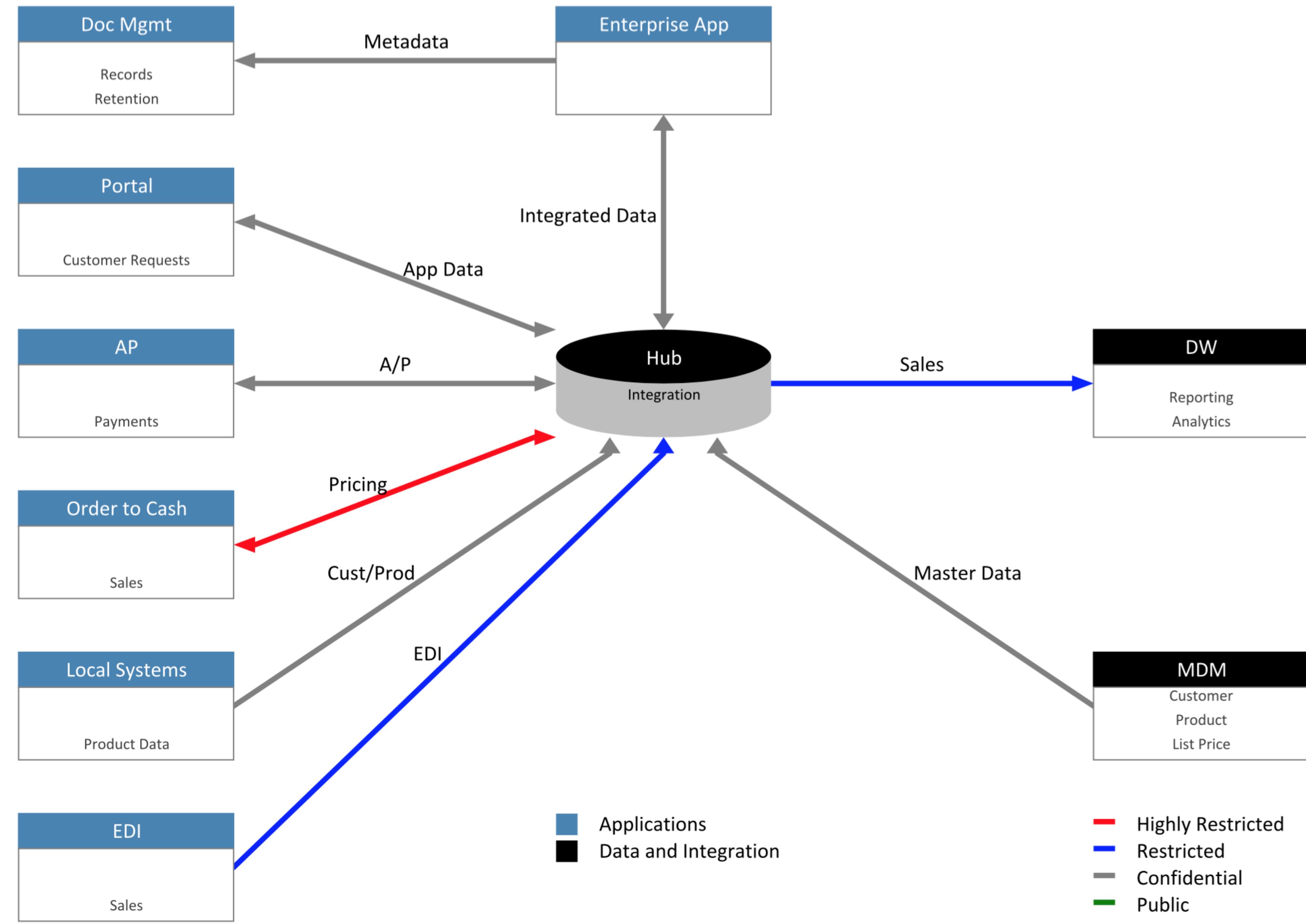
specify destination
begin/end the document
begin/end the document with viewport
begin/end group with attributes
begin/end group style
begin/end group transform
begin/end group id
begin/end clip path
begin/end a definition block
begin/end markers
begin/end pattern
set the description element
set the title element
define a script
begin/end mask element
begin/end link to href, with a title
use defined objects

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

SVGo Clients in the Repo

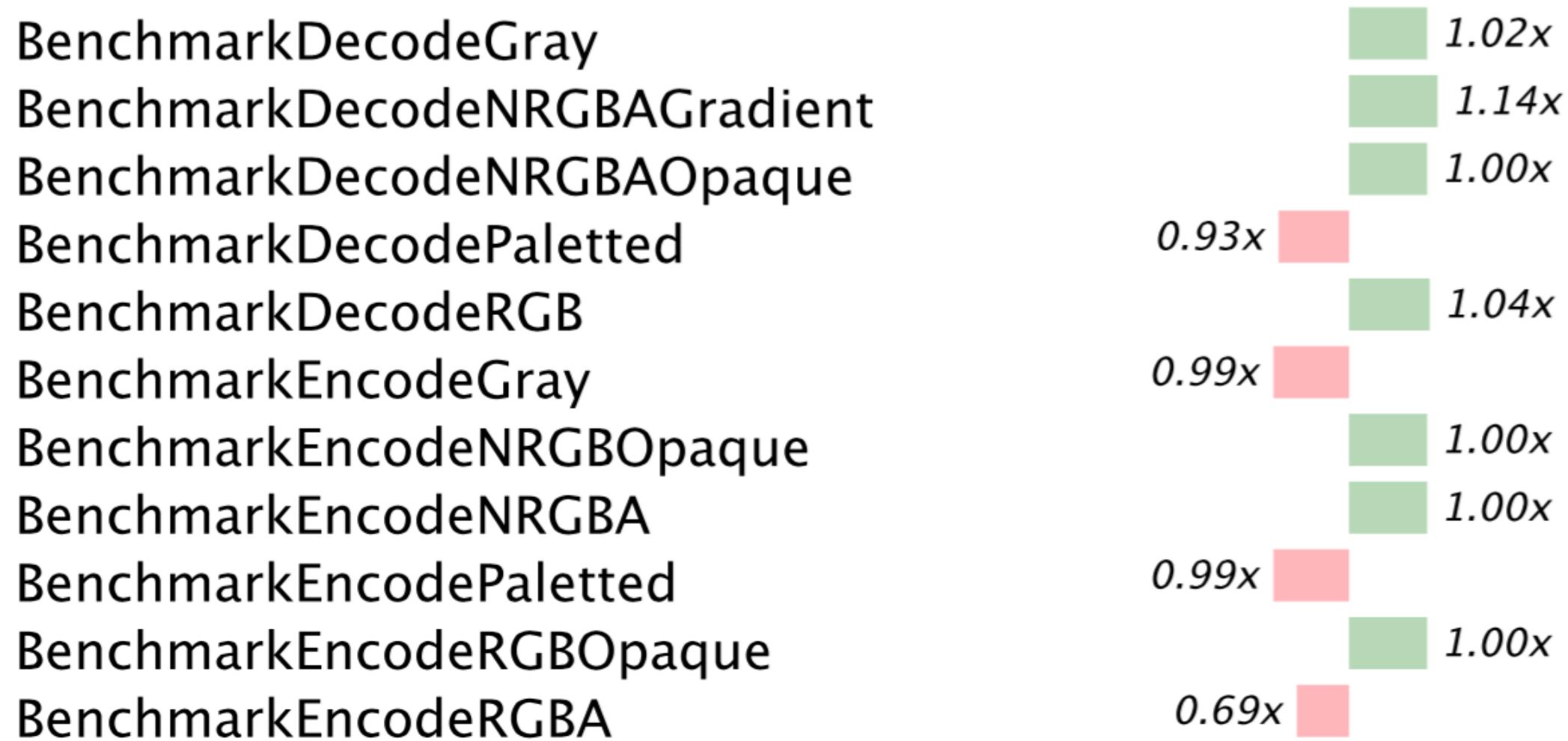
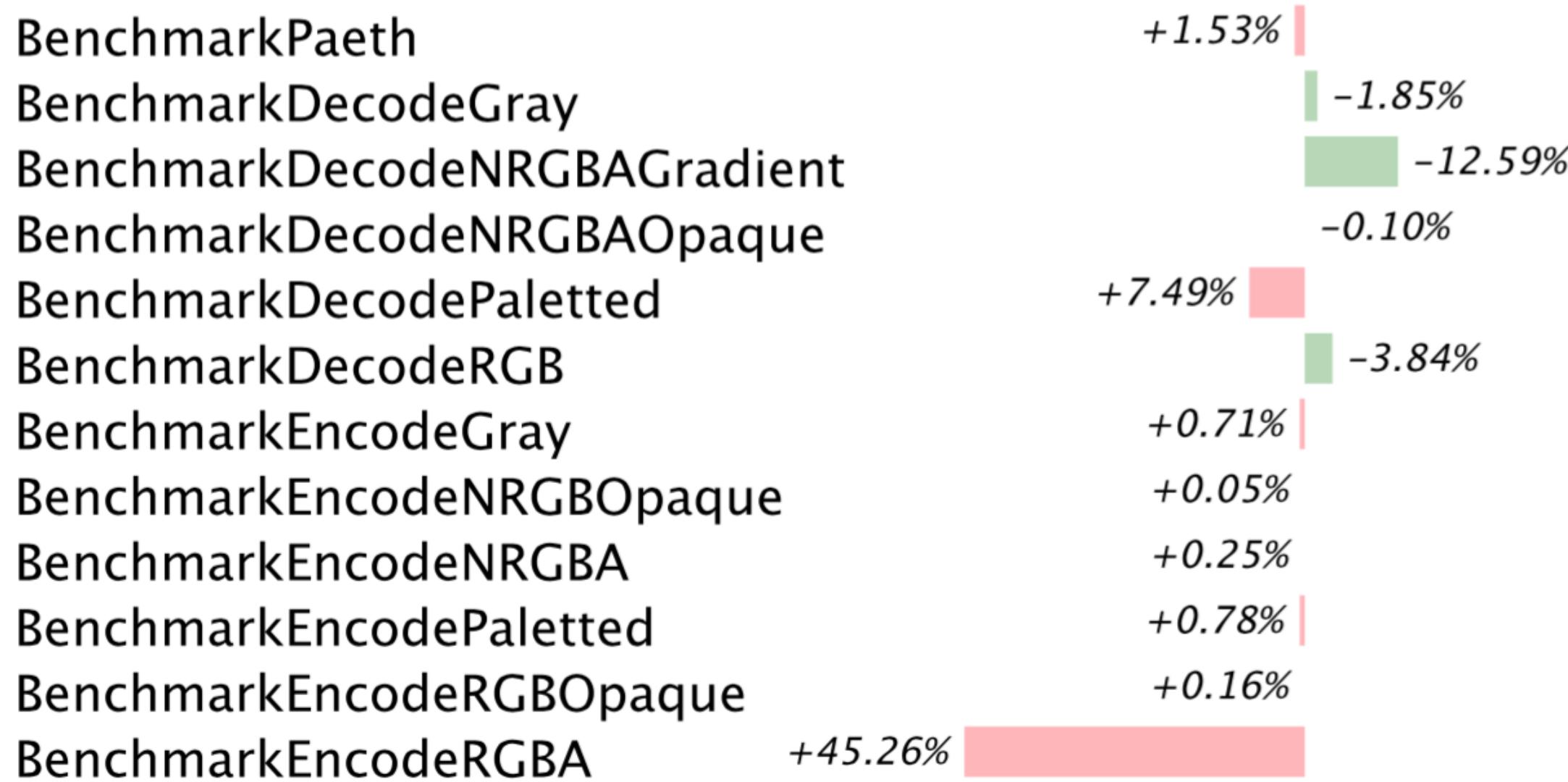
svgdef	Creates a SVG representation of the API	pattern	Test patterns
android	The Android logo	planets	Show the scale of the Solar system
bubtrail	Bubble trails	pmap	Proportion maps
bulletgraph	Bullet Graphs (via Stephen Few)	randcomp	Compare random number generators
colortab	Display SVG named colors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

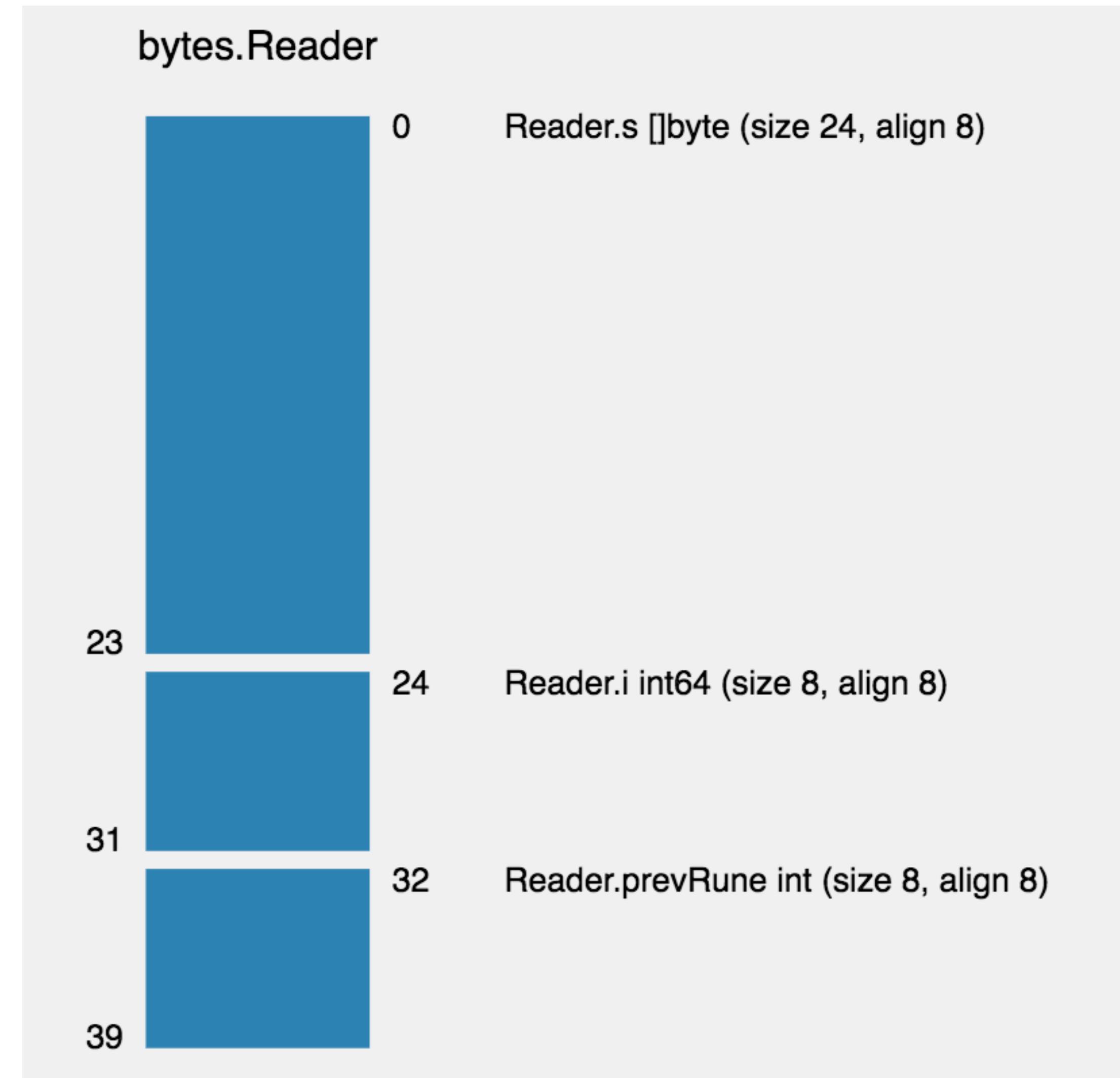
SVGo Clients



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

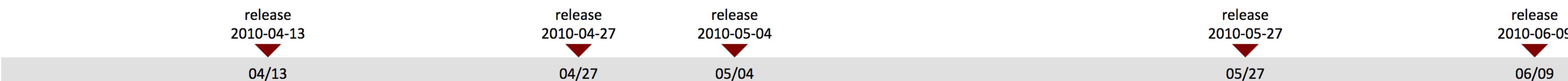
4Q 2009



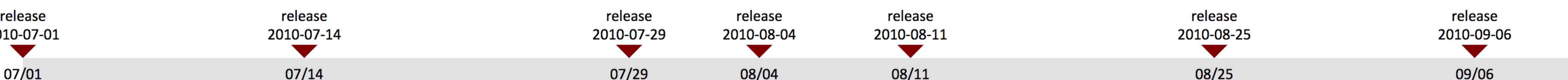
1Q 2010



2Q 2010

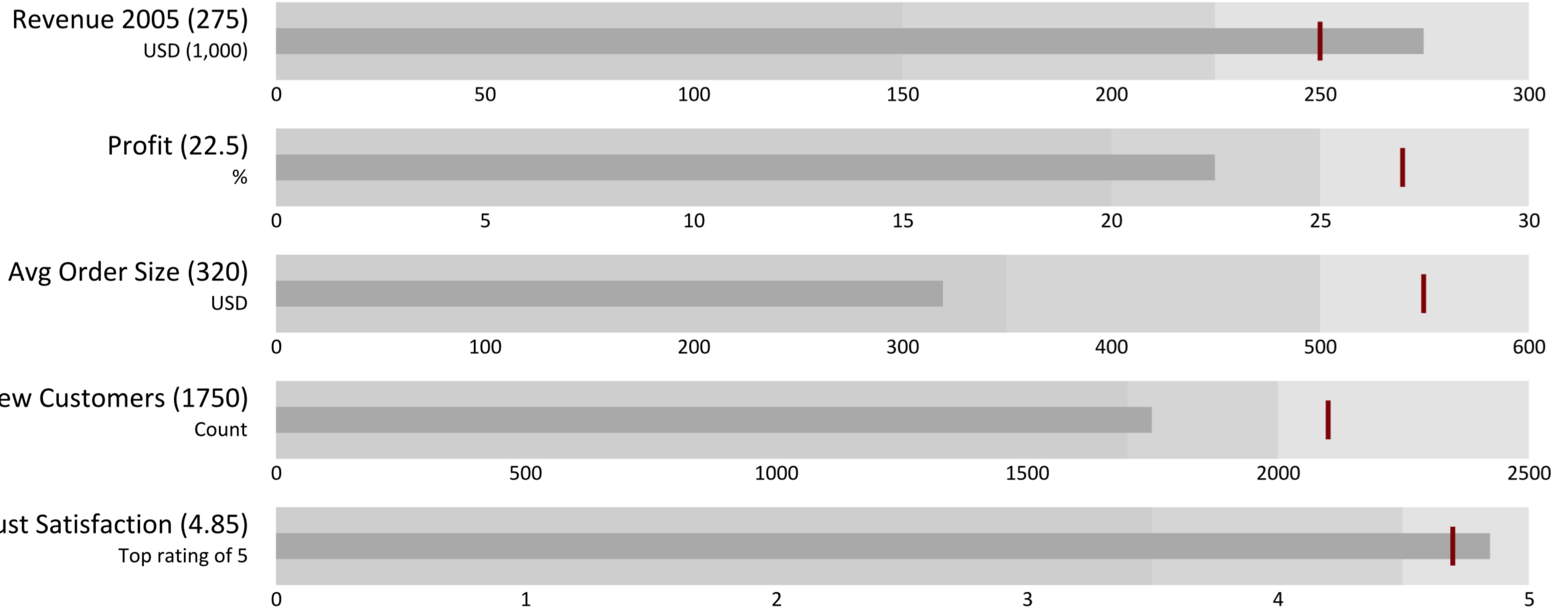


3Q 2010



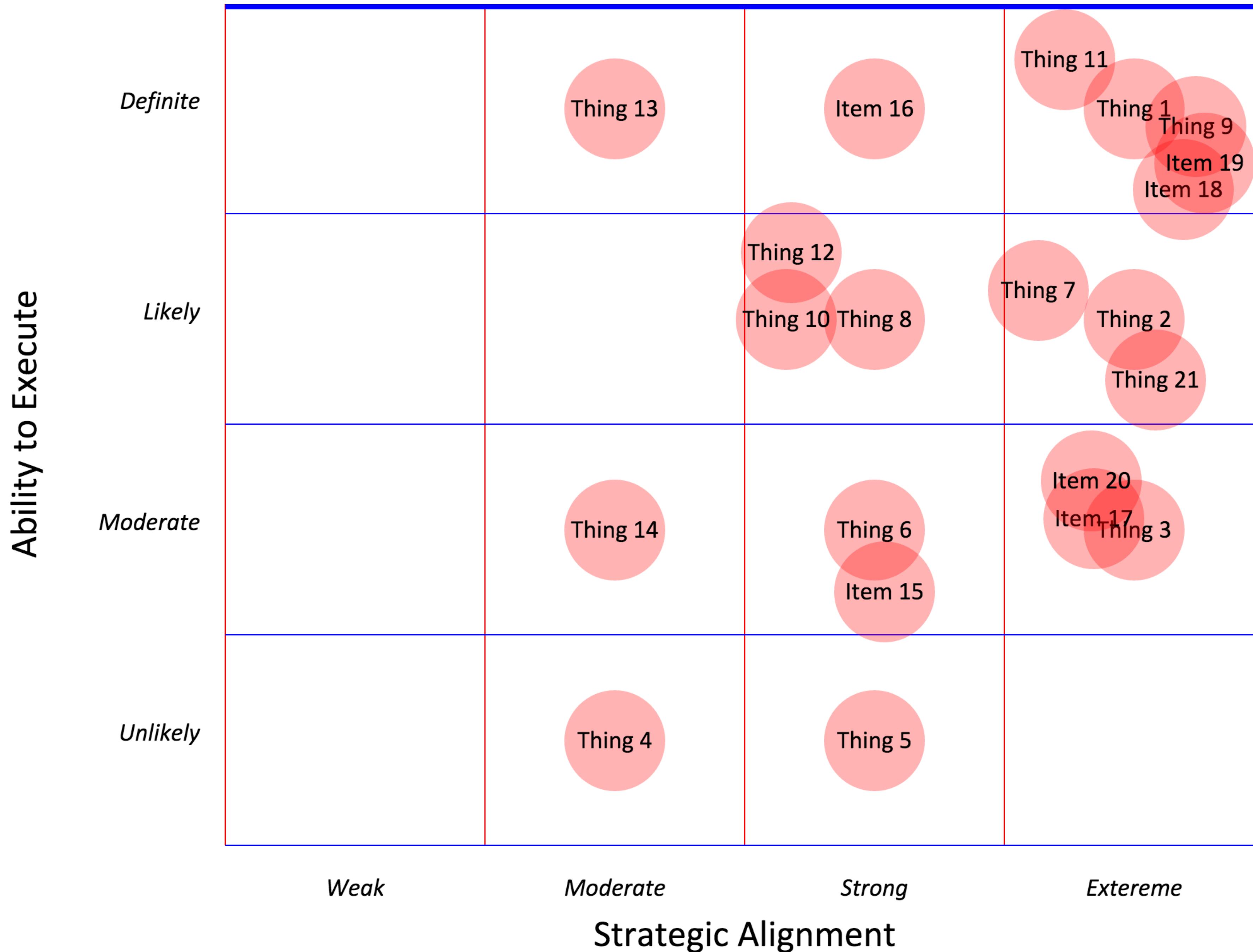
4Q 2010



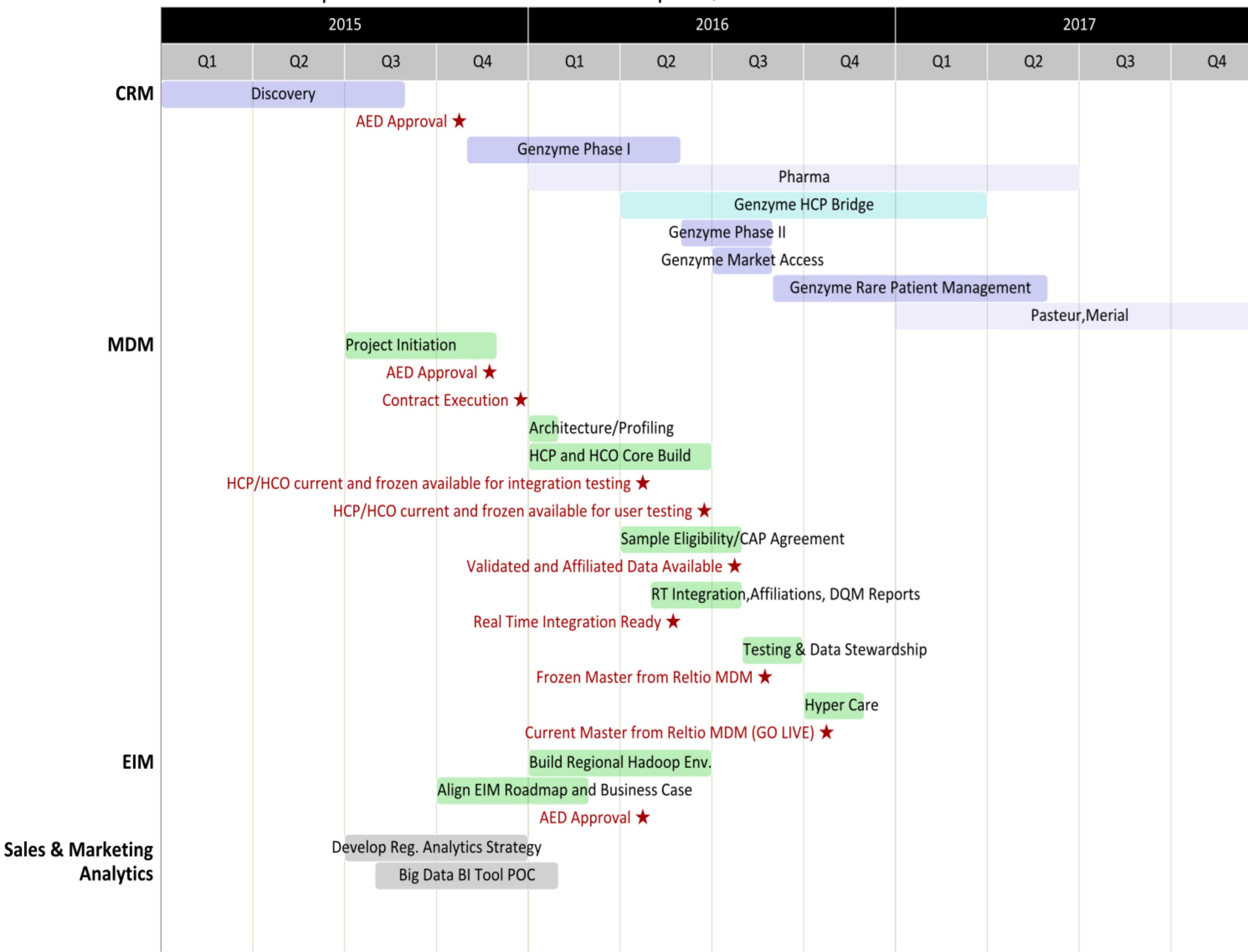


Sample Bullet Graph

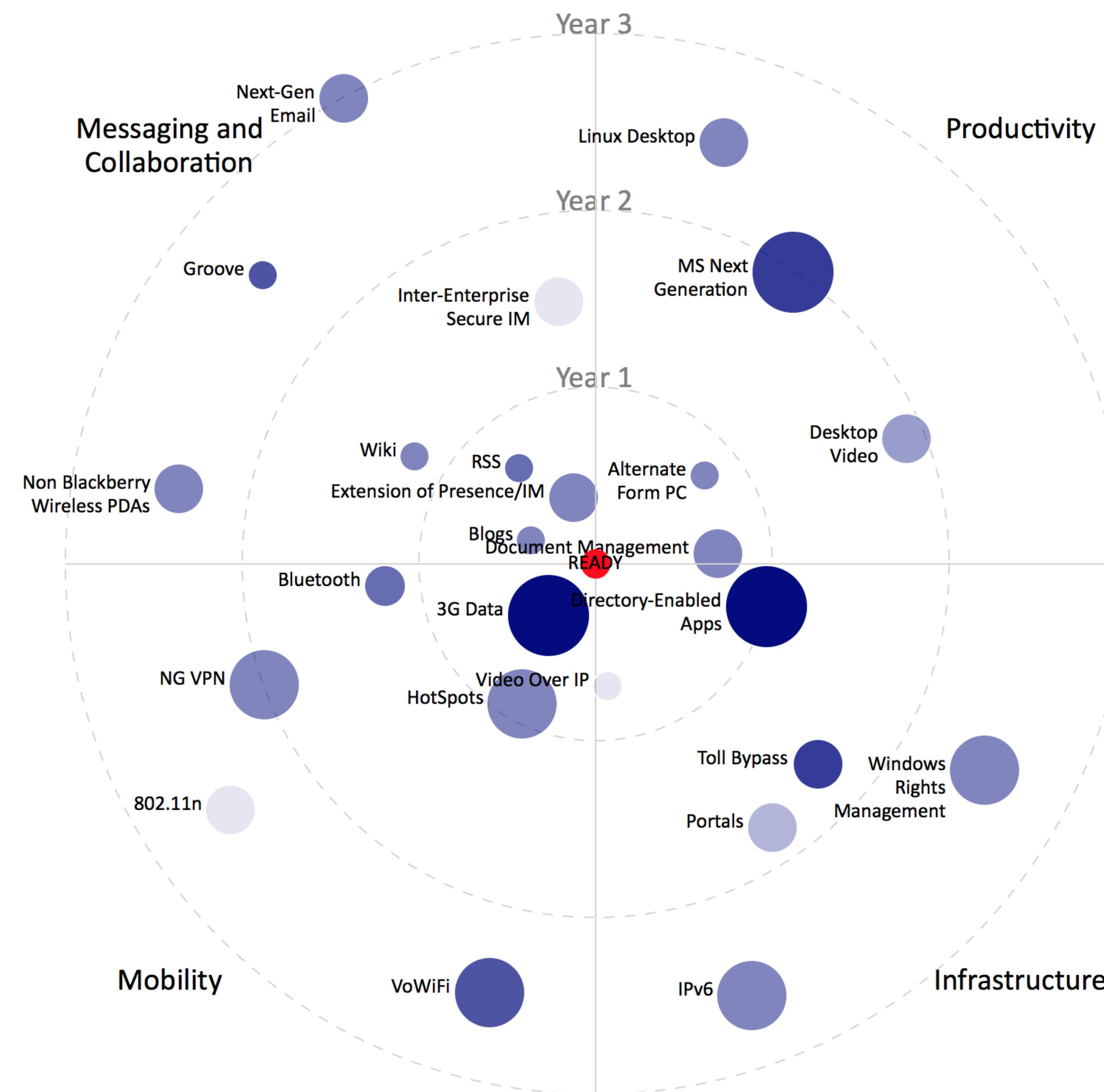
The bullet graph features a single, primary measure (for example, current year-to-date revenue) compares that measure to one or more other measures to enrich its meaning, for example, compared to a target), and displays it in the context of qualitative ranges of performance, such as poor, satisfactory,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



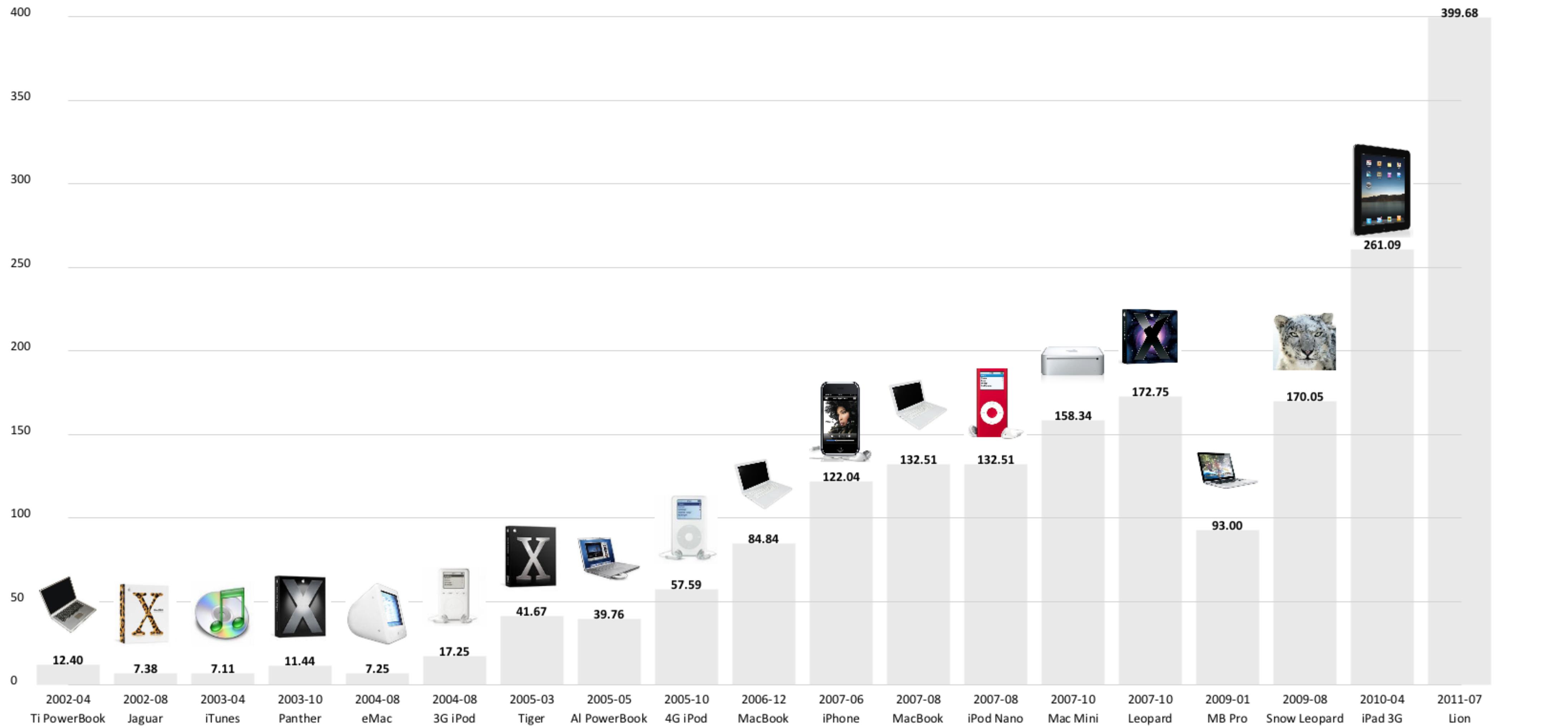
Commercial Capabilities Consolidated Roadmap: 4Q 2015



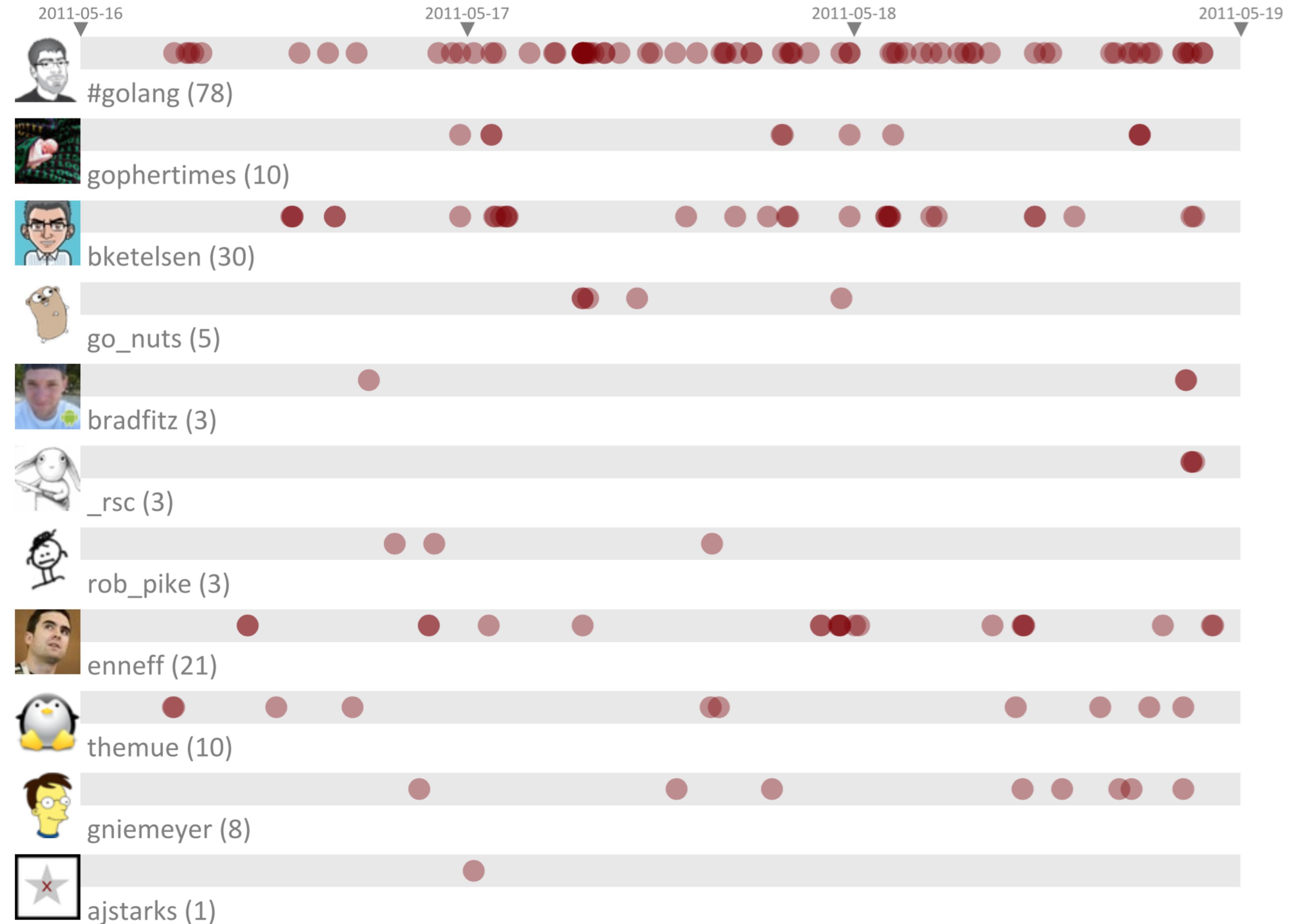




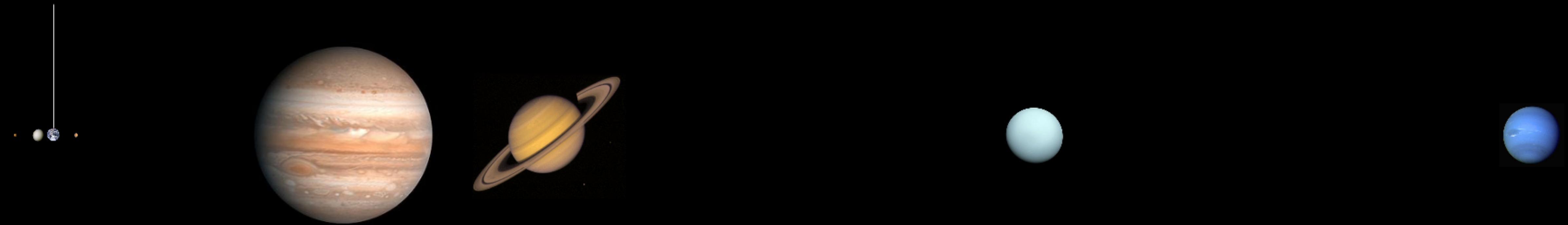
Apple Purchases and Stock Price



Twitter Update Frequency



You are here



JONES

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

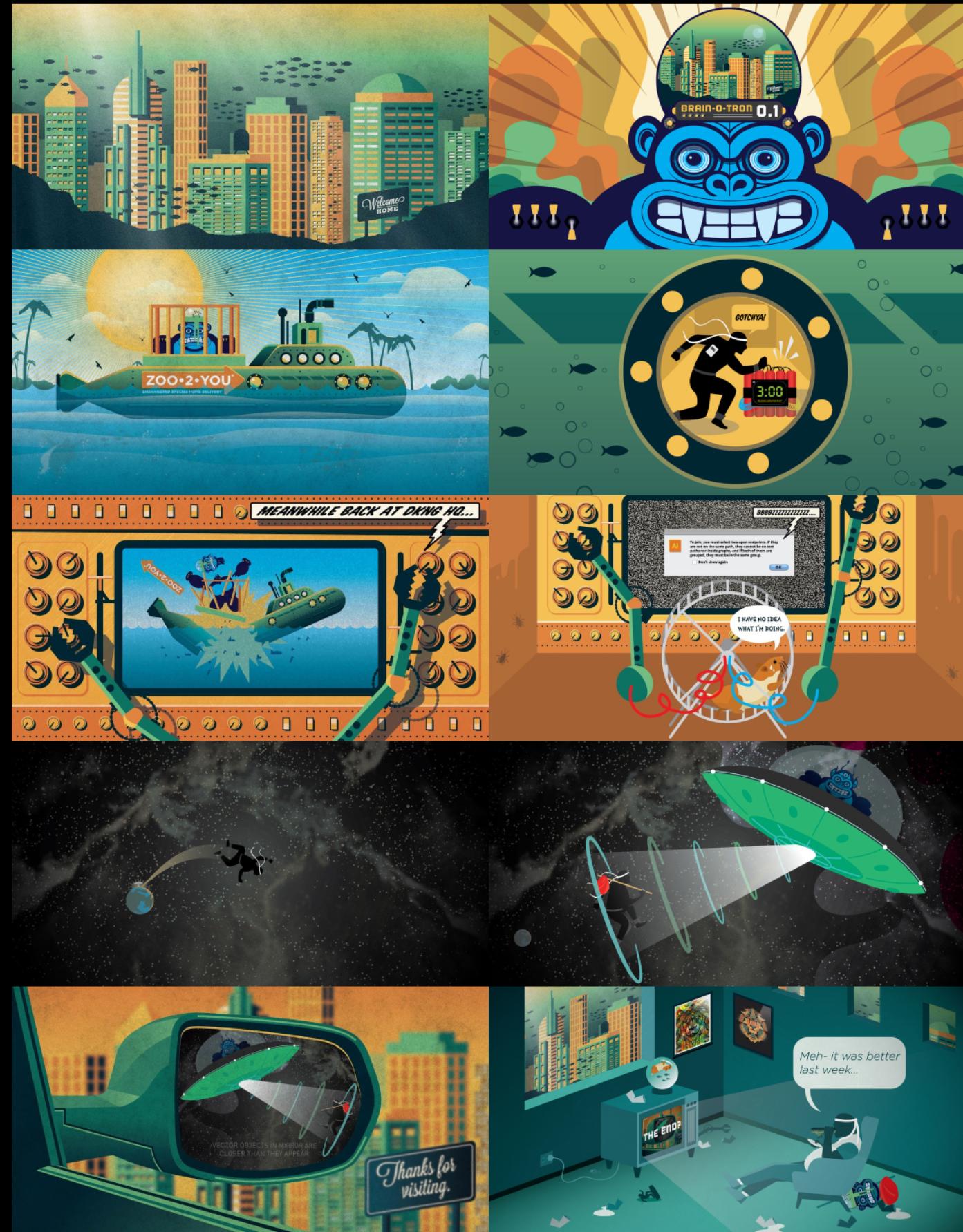
MONTIEL



DKNG

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

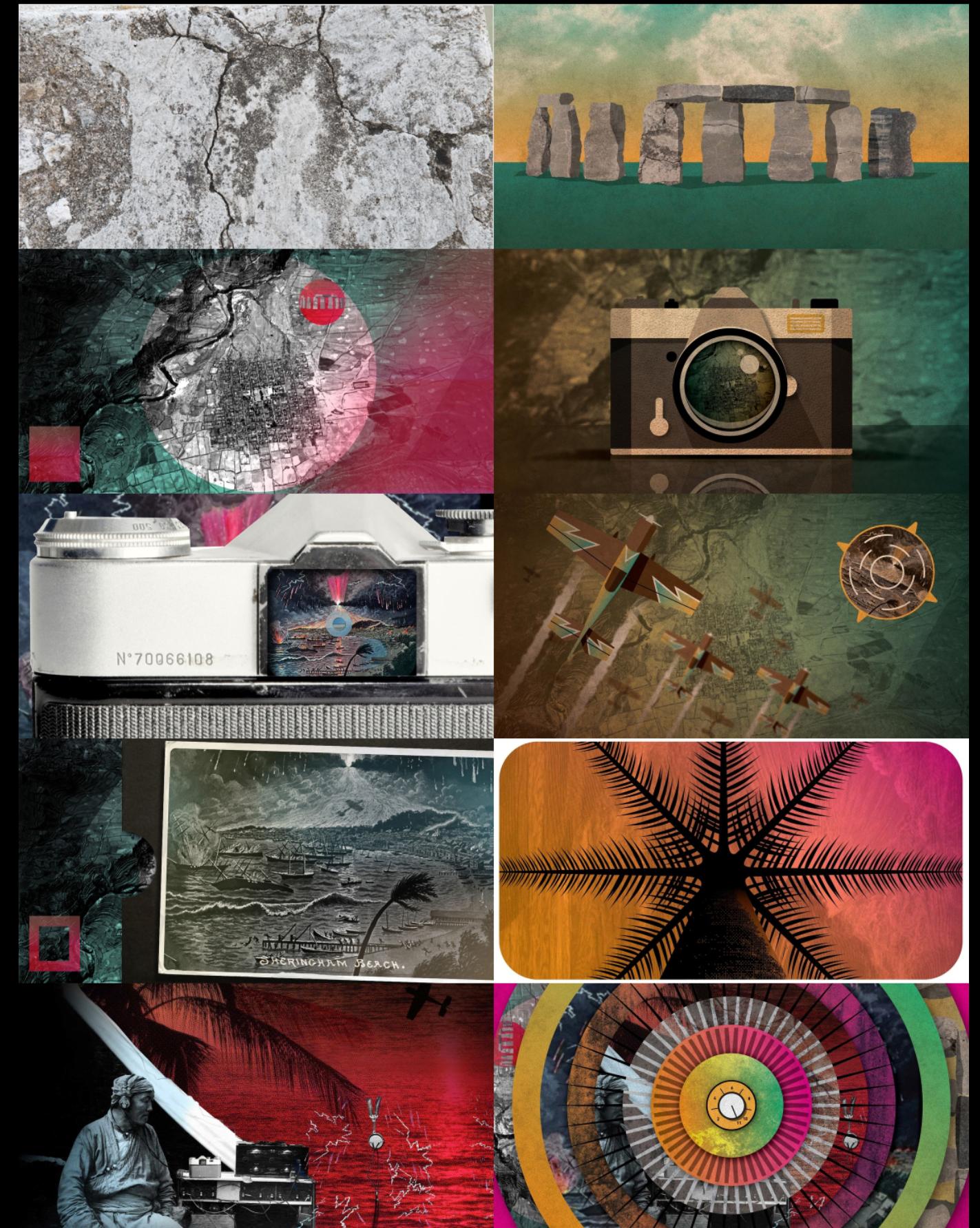
DDL



ANDERSON

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

DKNG



CONTINO
CASSARO



PUTNAM
STOCKS



WHITE
TAYLOR



HALL
WARREN



STRAWBERRY LUNA
DOUBLENAUT



JONES
MONTIEL



DKNG
DDL



SHAWNA X
STEVENS



TAYLOR
JONES



PUTNAM
RAJKUMAR



REYES
WHITE



ANDERSON
DKNG



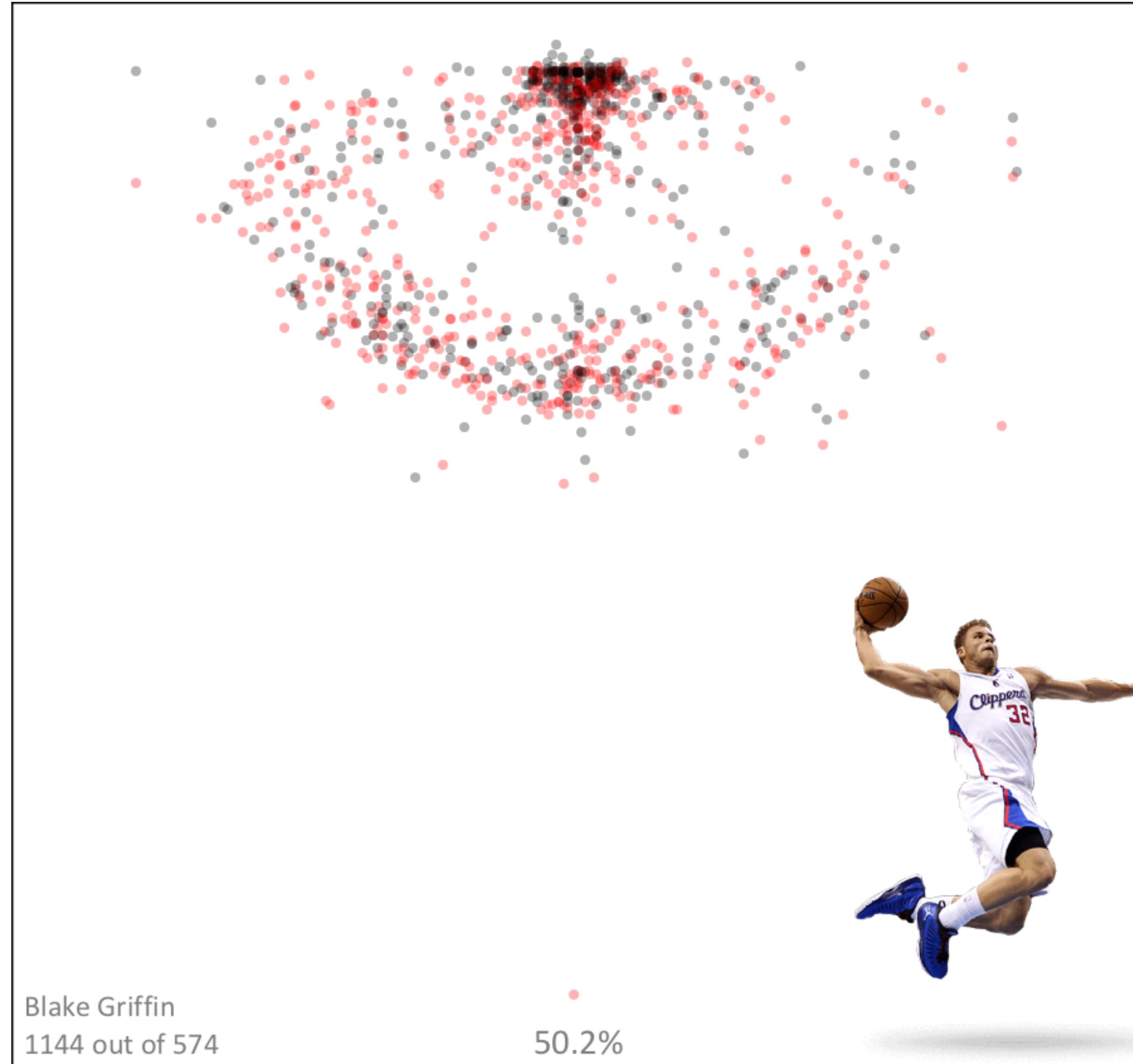
TAYLOR
WHITE

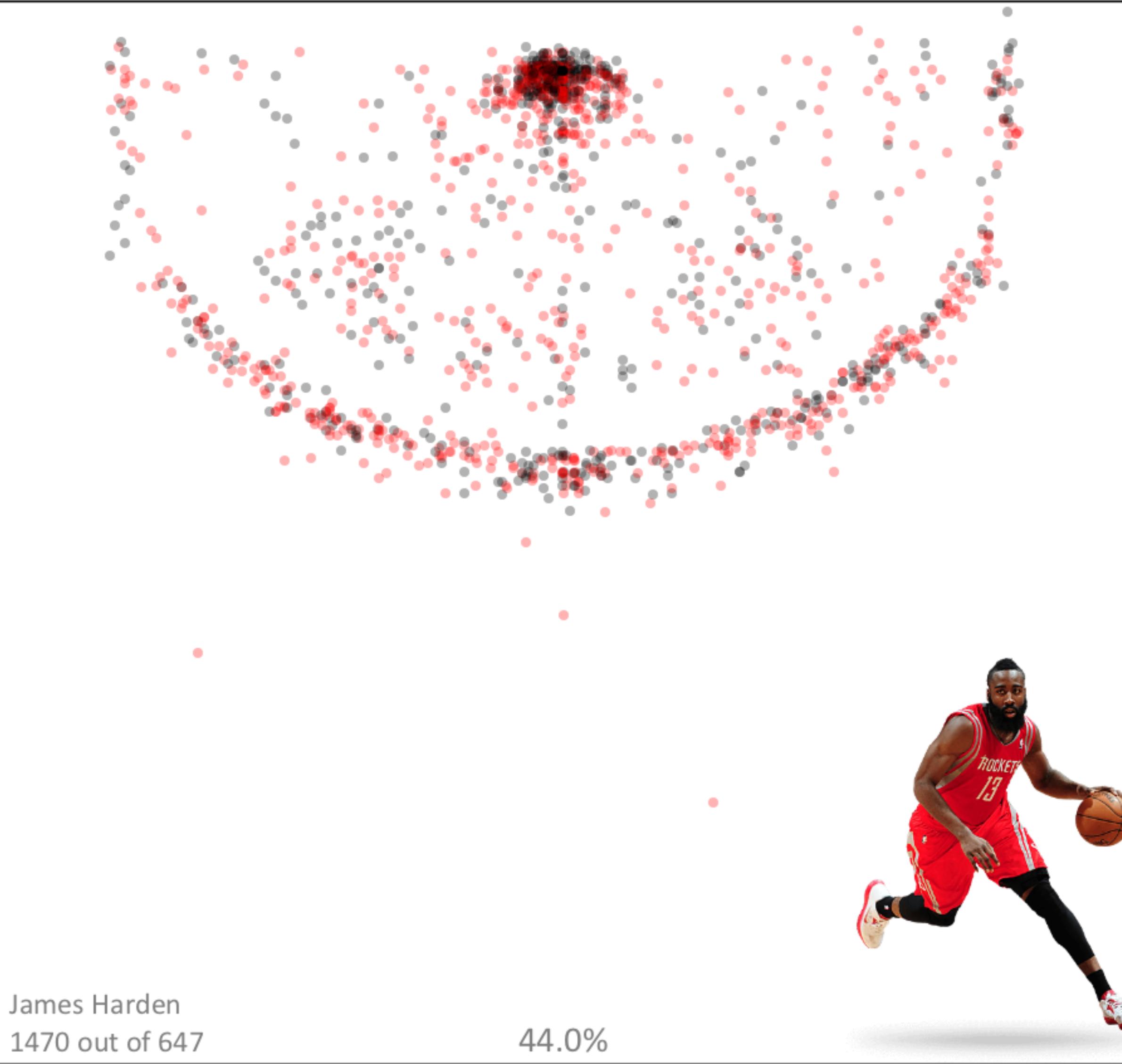


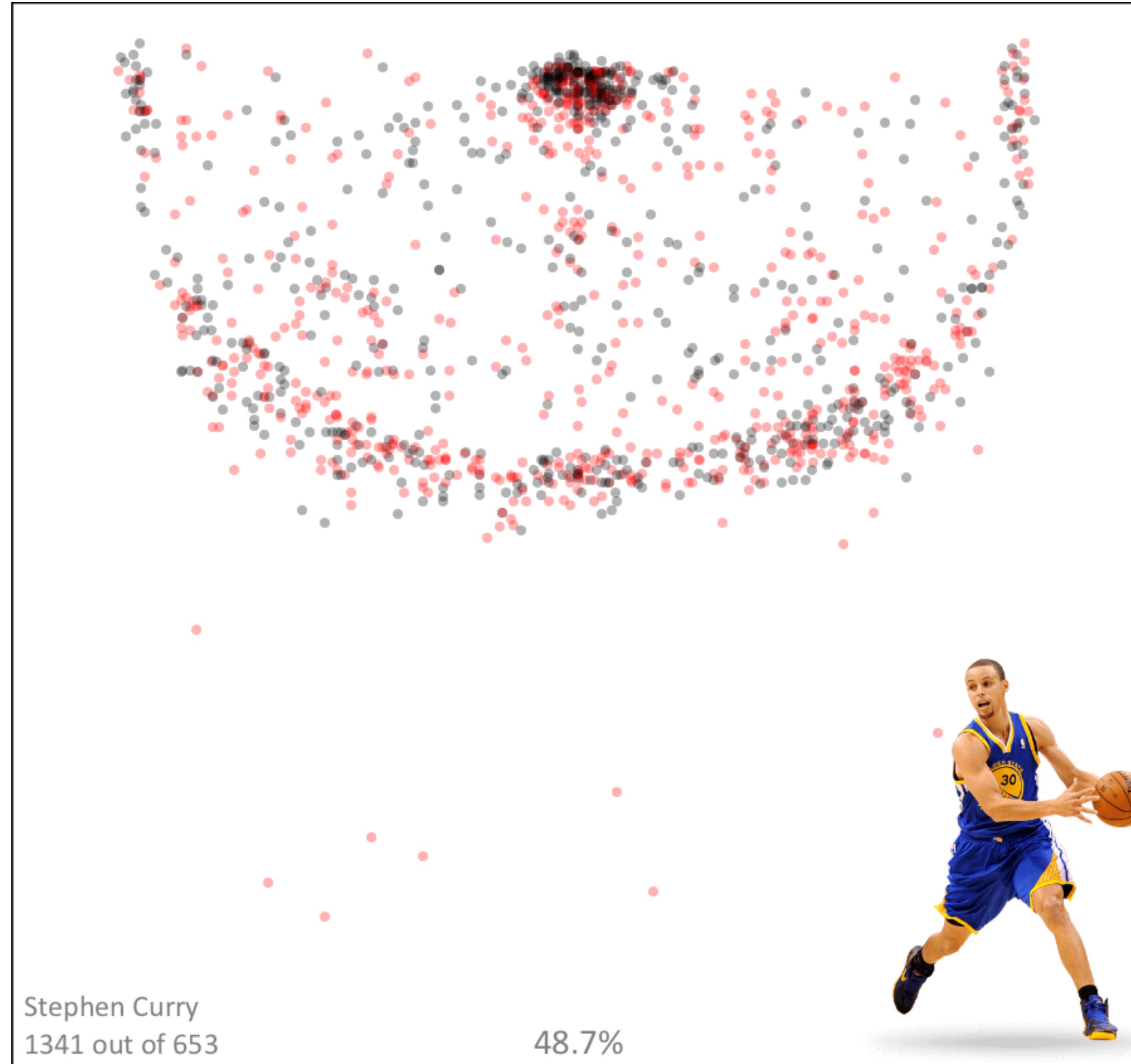
RAJKUMAR
ANDERSON



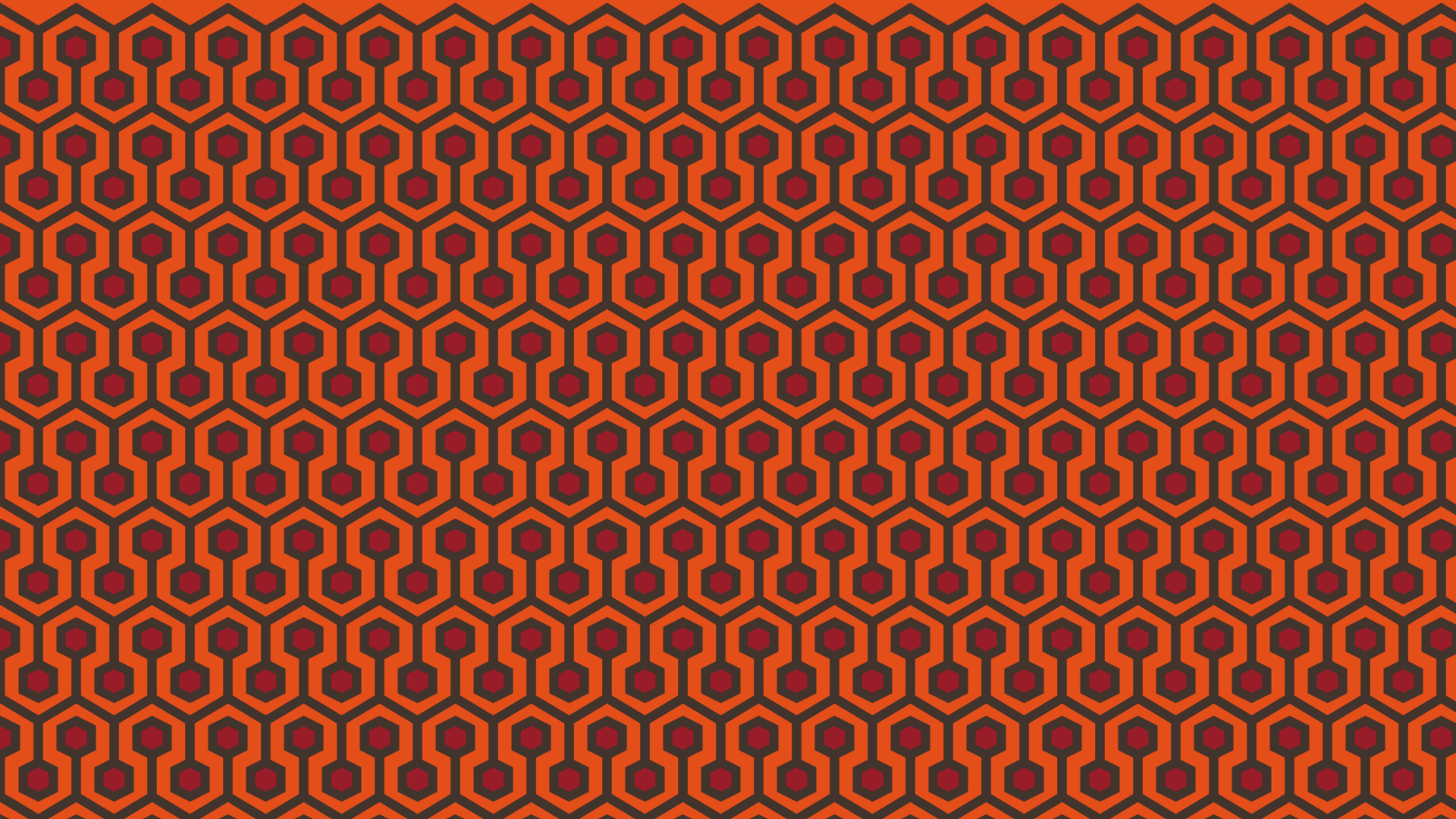
ANDERSON
WHITE













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

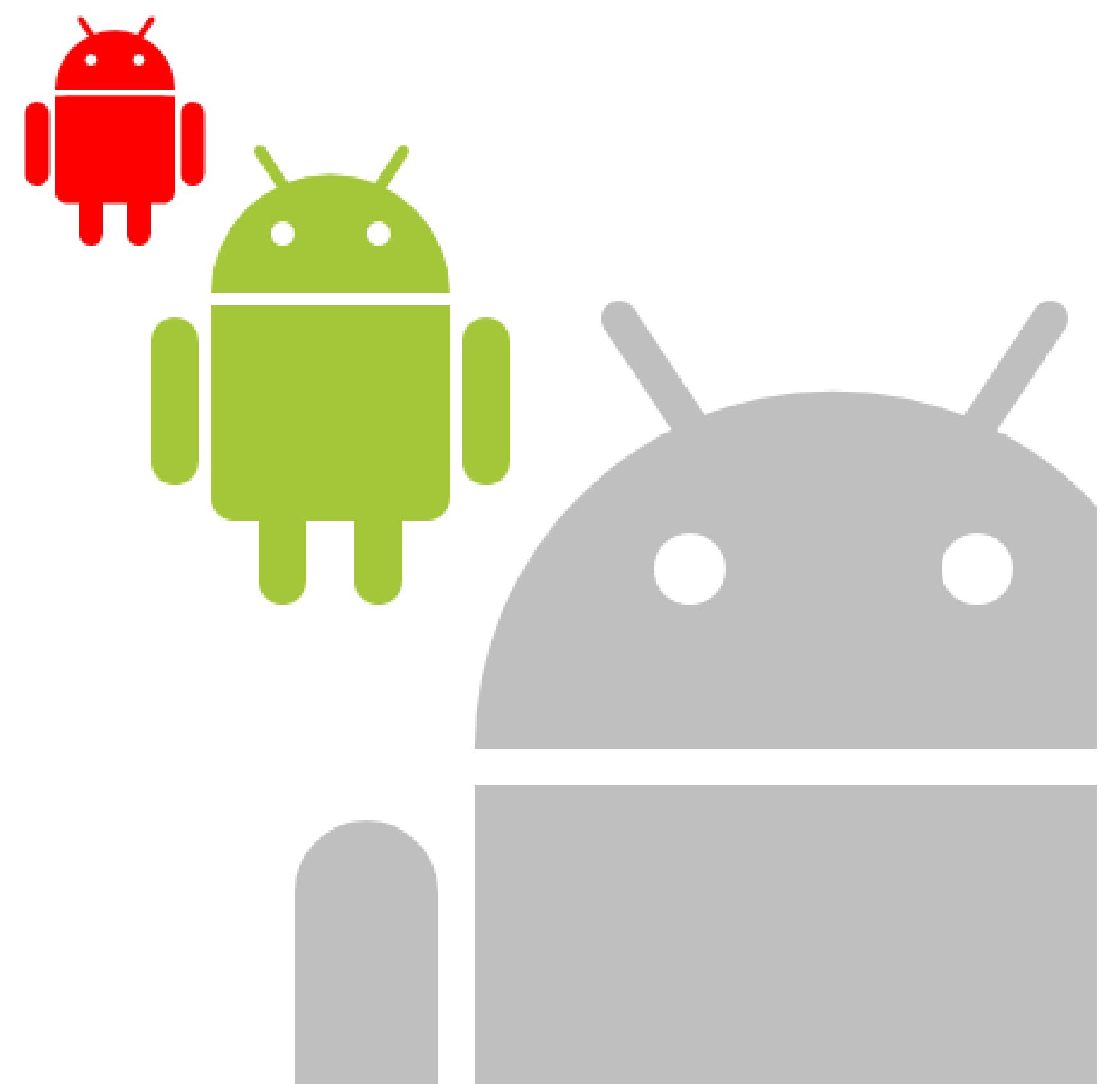


```
canvas.Gid("runit")
canvas.TranslateRotate(150, 180, 180)
canvas.Use(0, 0, "#unit")
canvas.Gend()
canvas.Gend()
canvas.DefEnd()
```



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

SVGo API Design



Scale



Roundrect



rgb(164,198,57)

Line

Circle

Arc

Line

Rect

Element

Rect

Arguments

(100,200,250,125)

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

(100, 200)



125

250

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"/>
```

(100, 200)



125

250

Element

Rect

Arguments

(100,200,250,125,

`id='box'`, `fill='gray'`, `stroke='blue'`)

Attributes

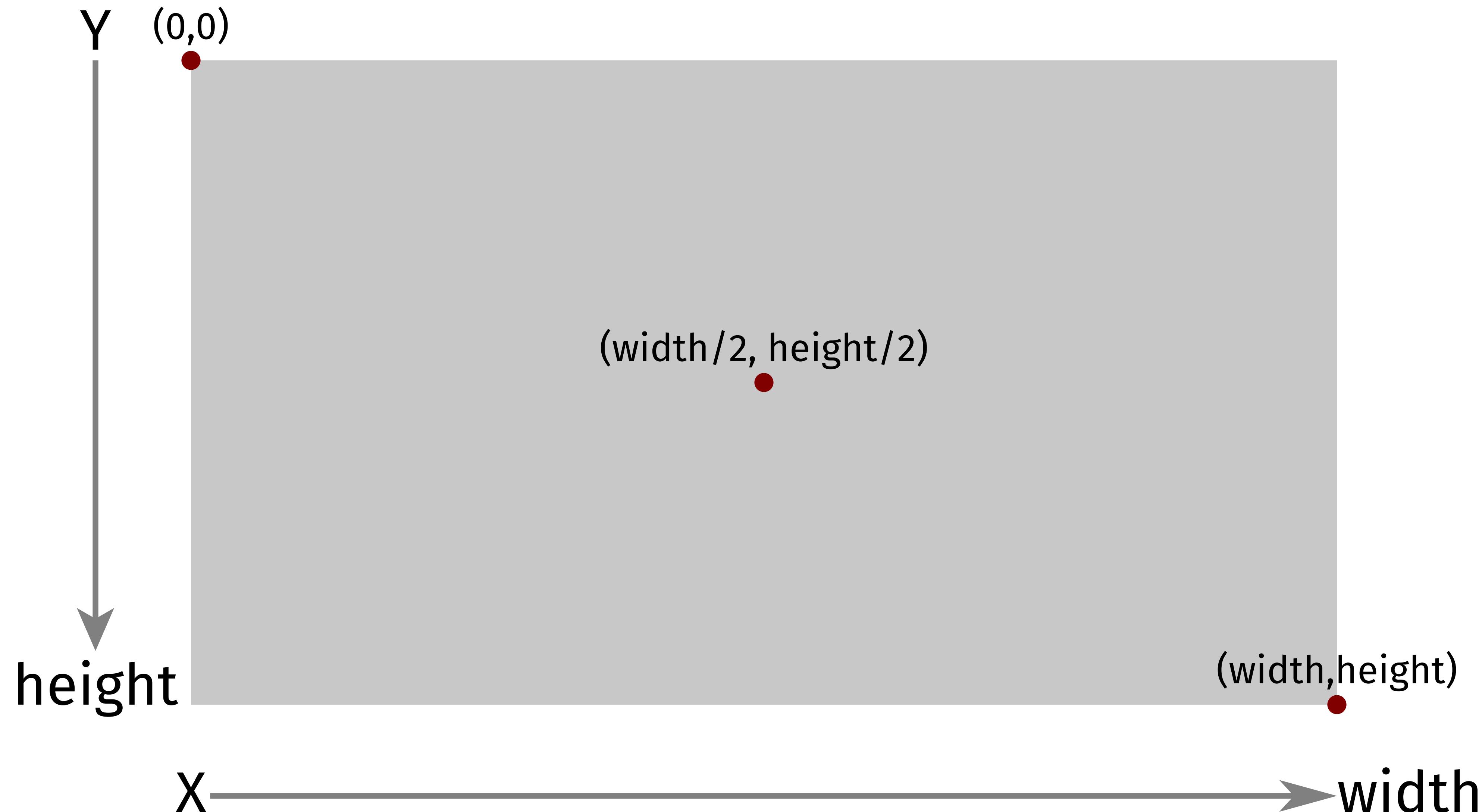
```
<rect x="100" y="200" width="250" height="125"  
      id='box' fill='gray' stroke='blue'/>
```

(100, 200)



125

250



Rect(0,0,width,height)

hello,
world

(width/2, height/2)

(width/2, height)

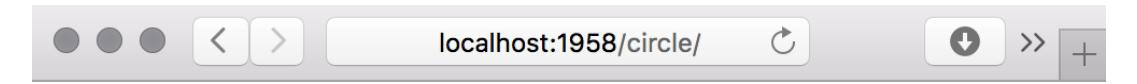
```
package main

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

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()
}
```

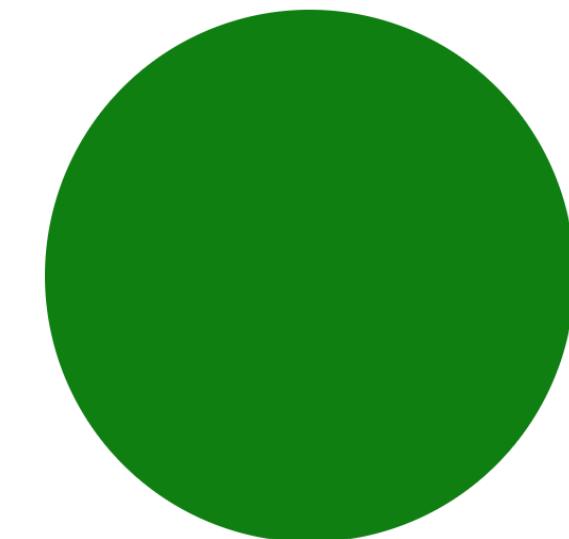
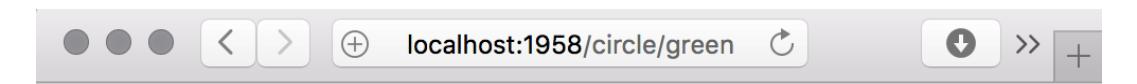


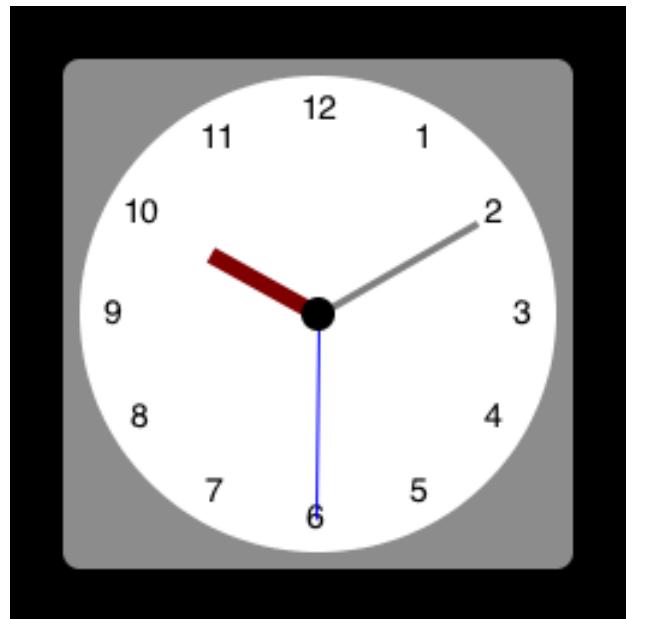
```
const defaultstyle = "fill:rgb(127,0,0)"
```



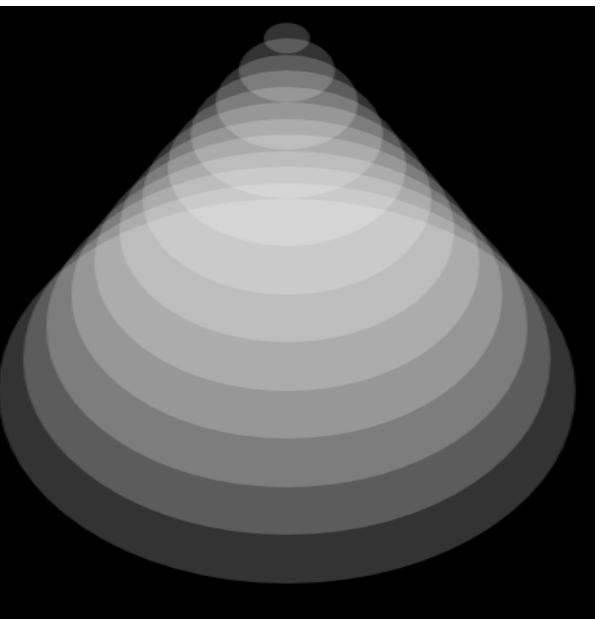
```
func circle(w http.ResponseWriter, req *http.Request) {
    w.Header().Set("Content-Type", "image/svg+xml")
    s := svg.New(w)
    s.Start(500, 500)
    s.Title("Circle")
    s.Circle(250, 250, 125, shapestyle(req.URL.Path))
    s.End()
}
```

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

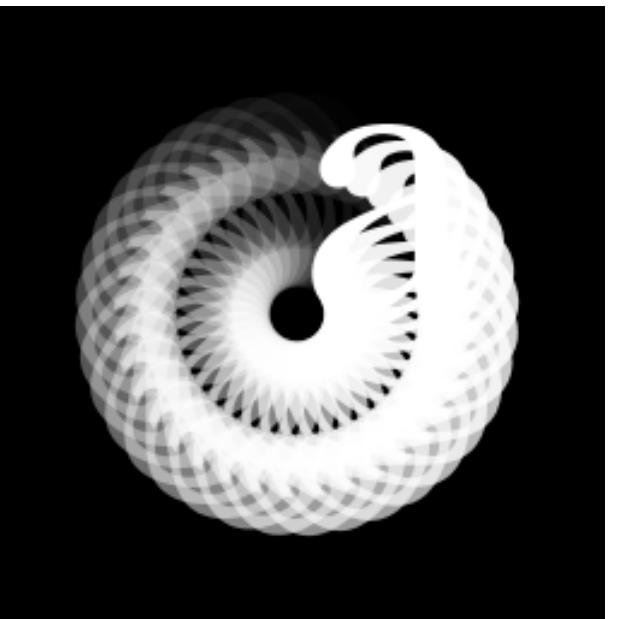




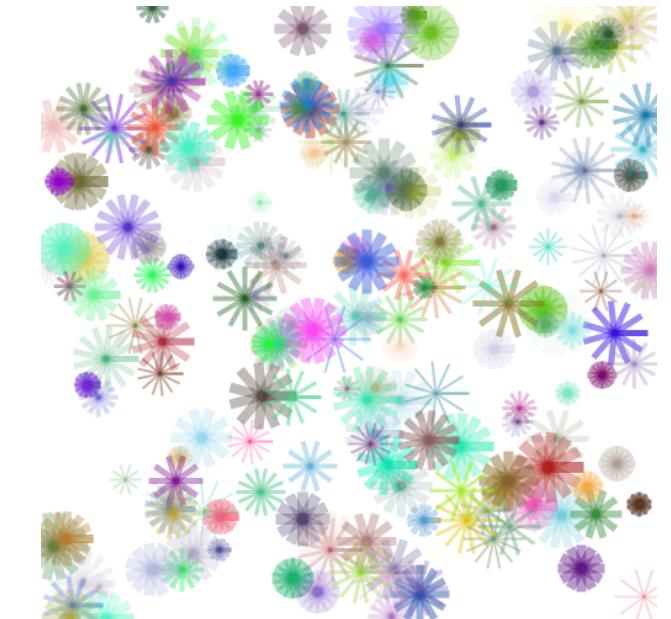
clock



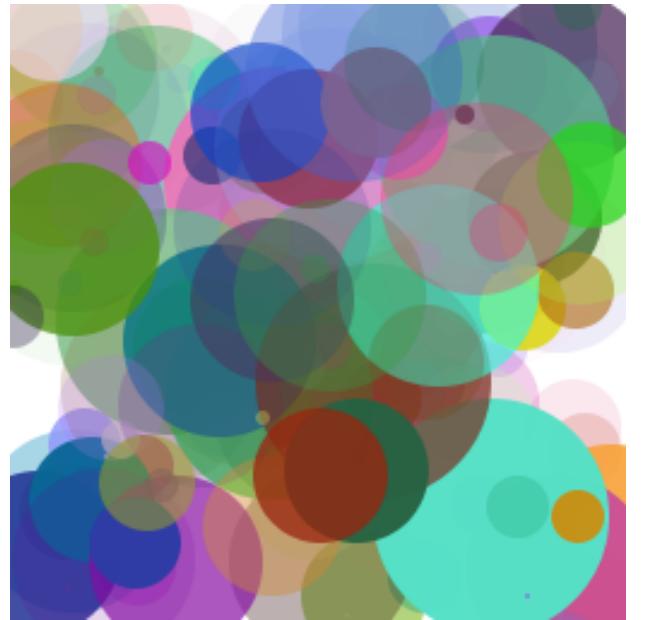
funnel



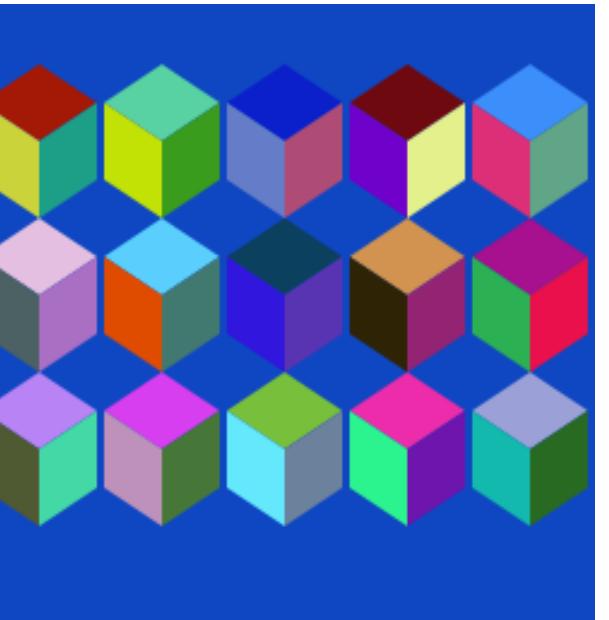
rotext



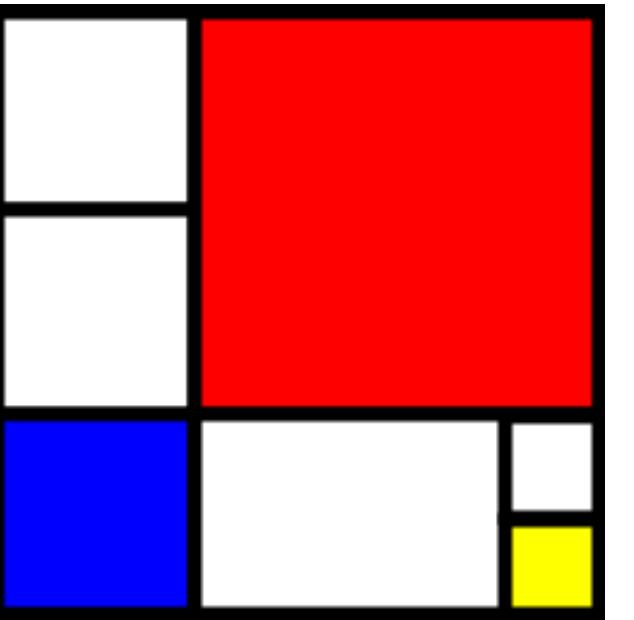
flower



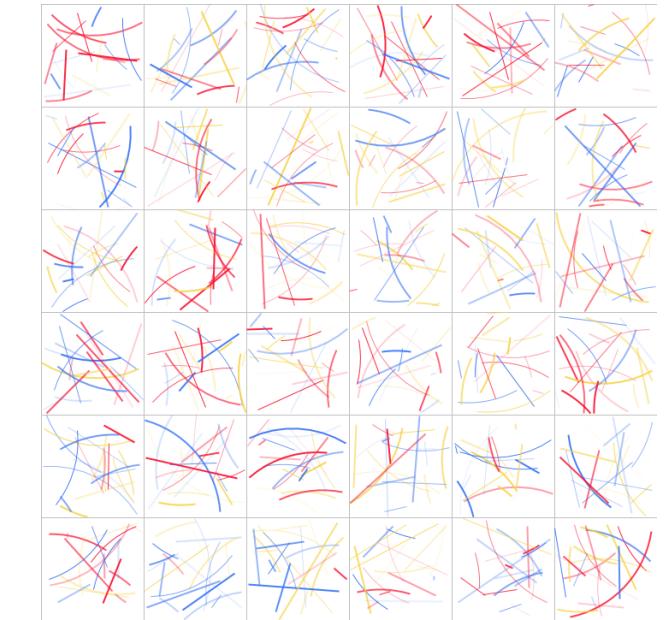
rshape



cube



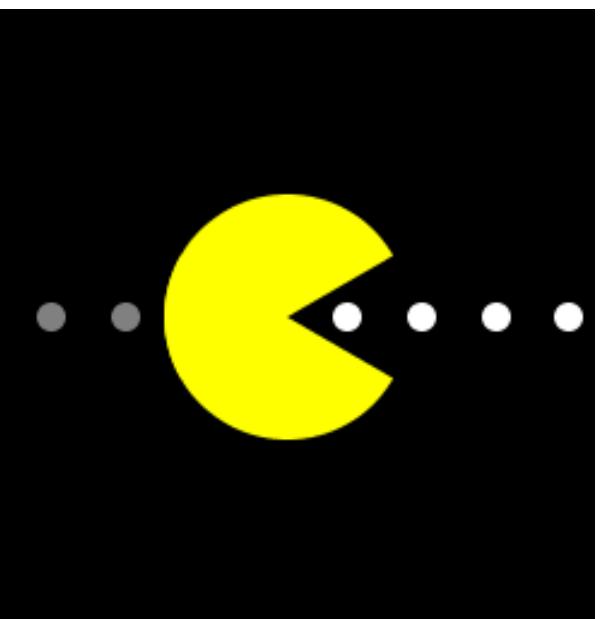
mondrian



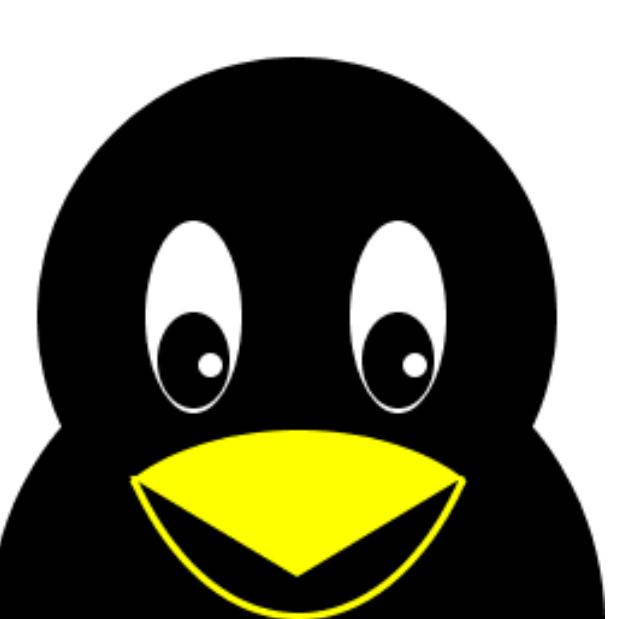
lewitt



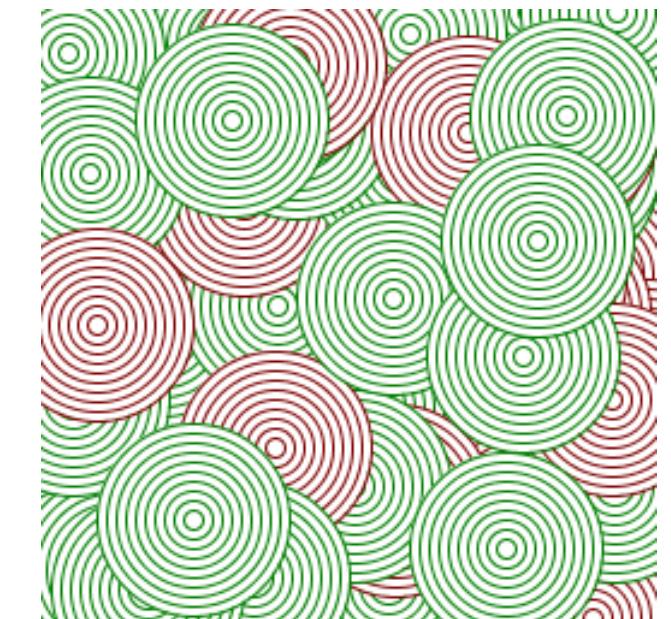
face



pacman

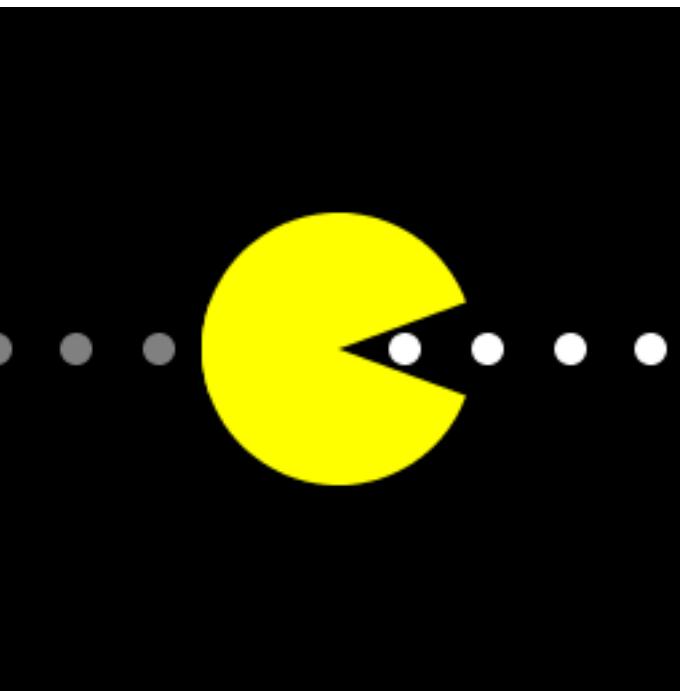


tux

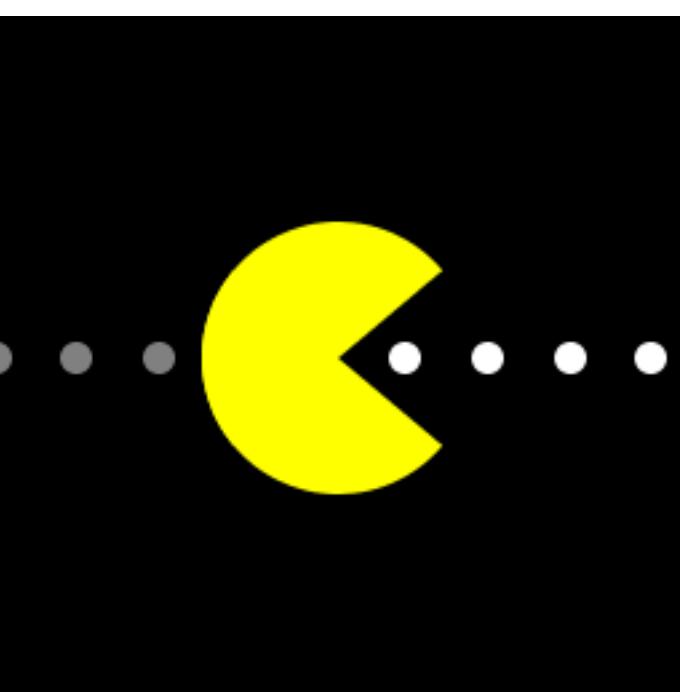


concentric

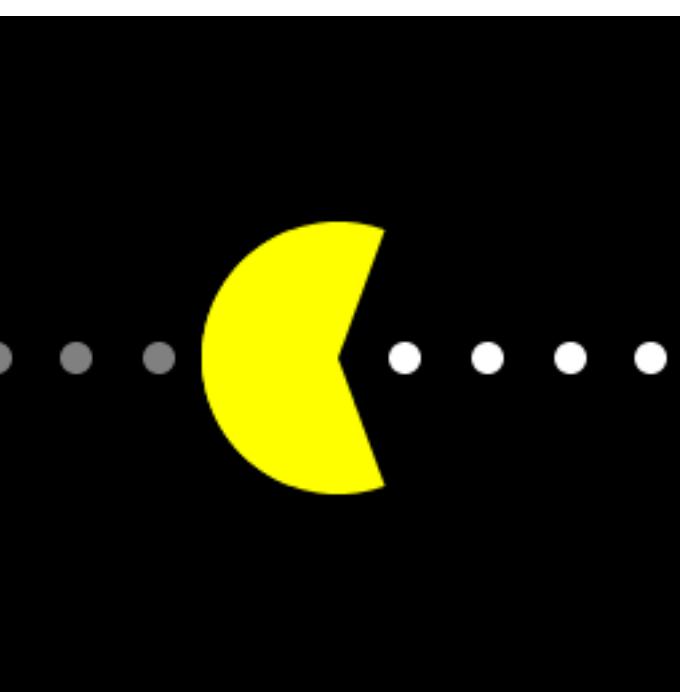
/pacman/?angle=20



/pacman/?angle=40



/pacman/?angle=70

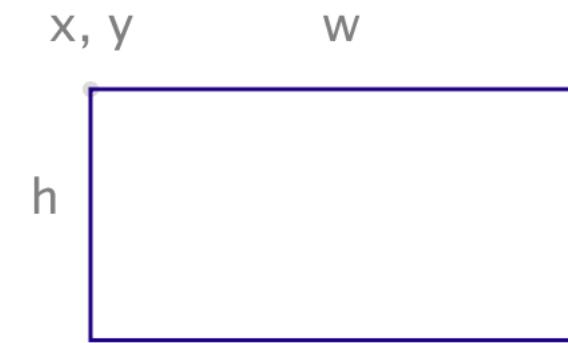


API Reference

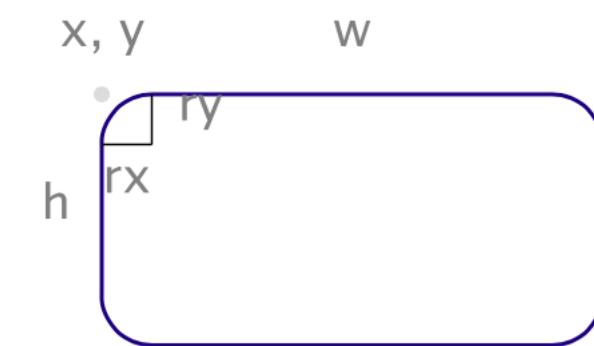
SVGo Method Categories

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

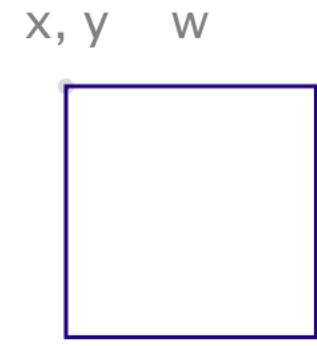
Shapes



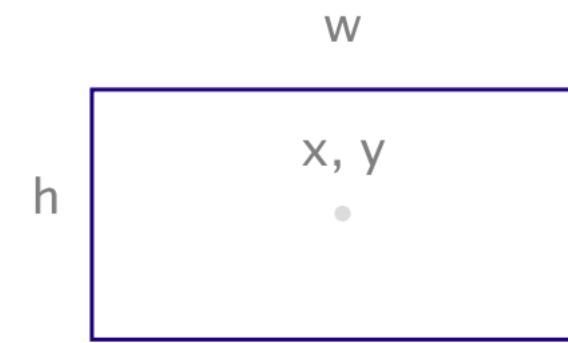
Rect(x, y, w, h int, s string...)



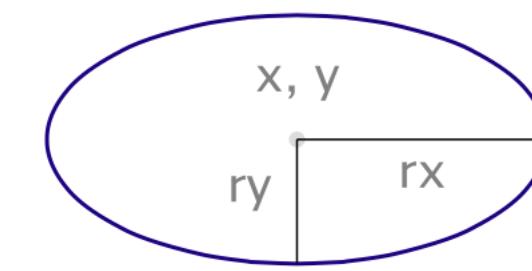
Roundrect(x, y, w, h, rx, ry int, s string...)



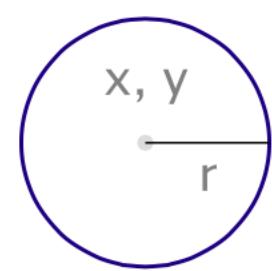
Square(x, y, l int, s string...)



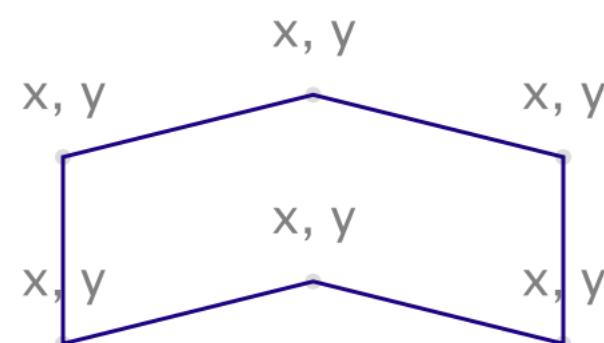
CenterRect(x, y, w, h int, s string...)



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



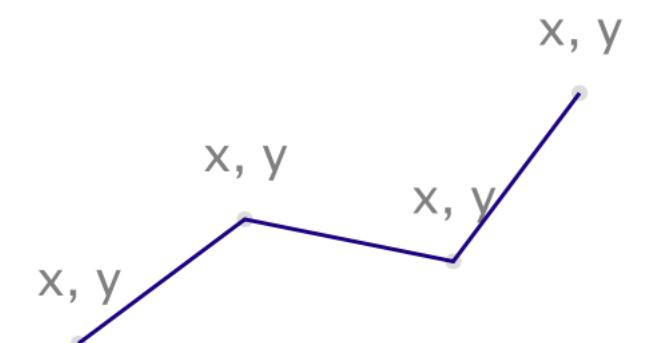
Circle(x, y, r int, s string...)



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



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

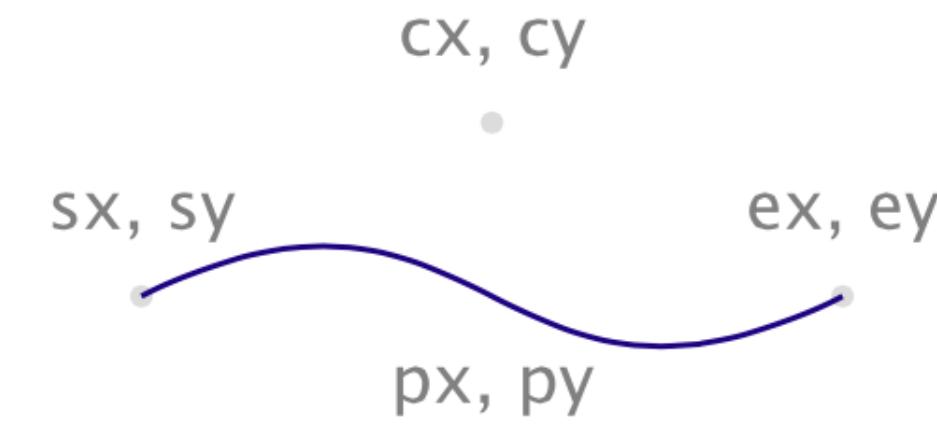


Polyline(x, y []int, s string...)

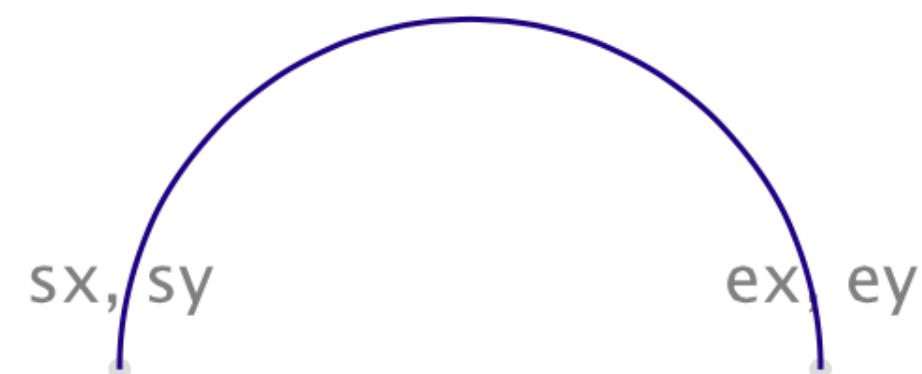
Curves and Paths



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



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

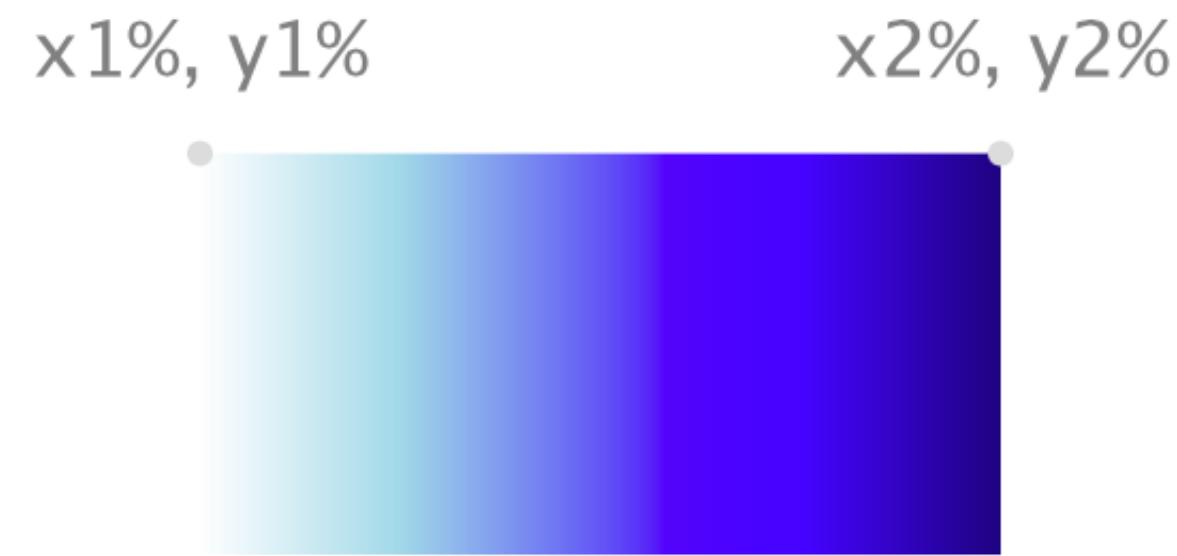


`Arc(sx, ax, ay, r int, dir, large bool, ex, ey int, s string...)`

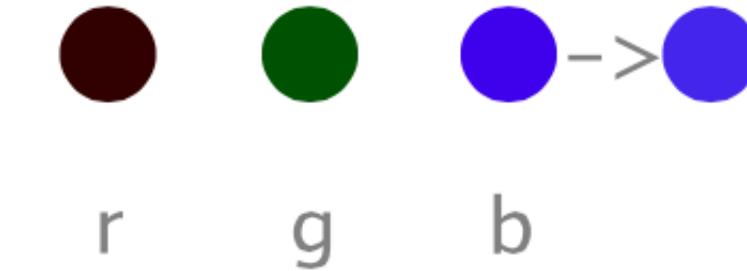


`Path(d string, s string...)`

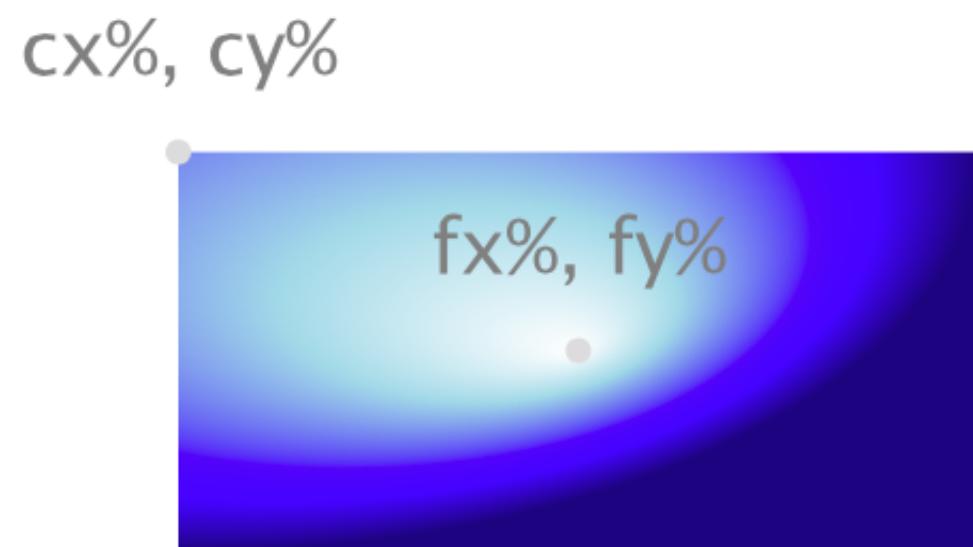
Gradients and Colors



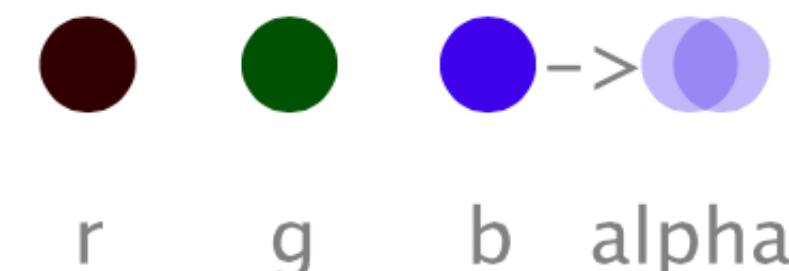
LinearGradient(id string, x1, y1, x2, y2 uint8, sc []Offcolor)



RGB(r, g, b int)

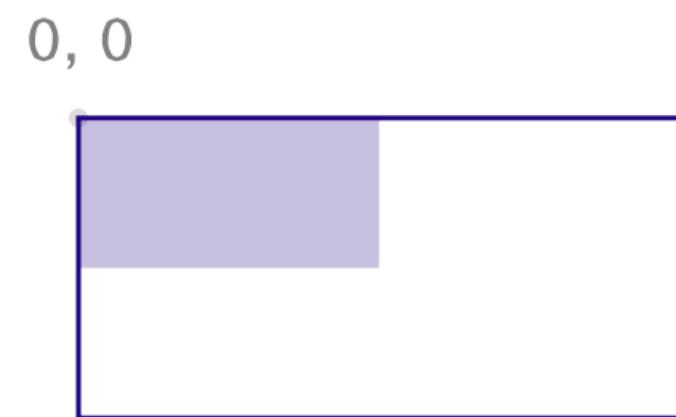


RadialGradient(id string, cx, cy, r, fx, fy uint8, sc []Offcolor)

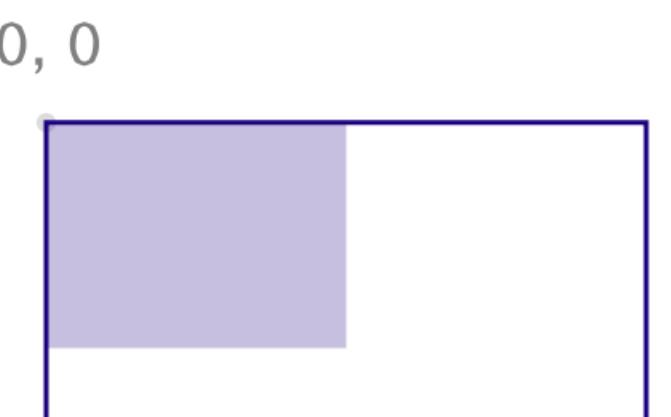


RGBA(r, g, b int, a float64)

Transforms



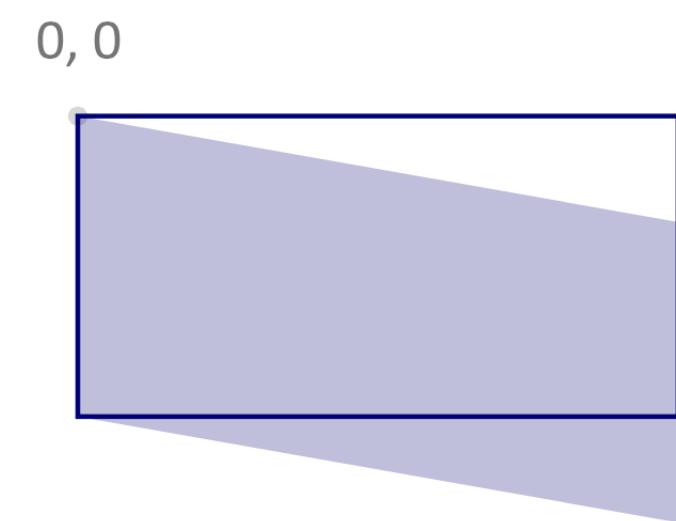
Scale(n float64)



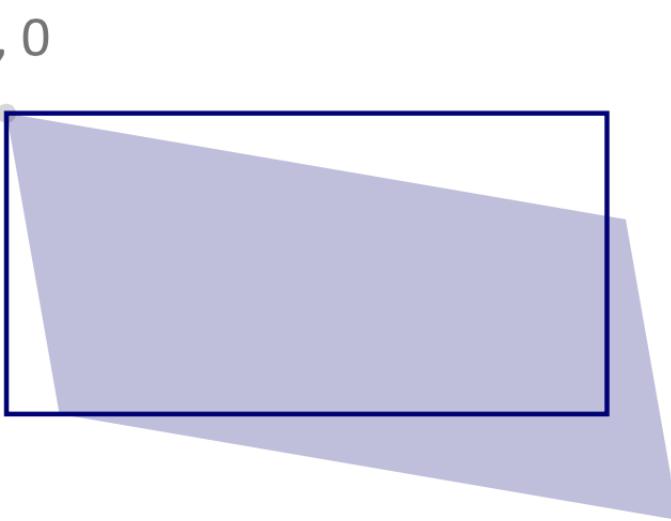
ScaleXY(dx, dy float64)



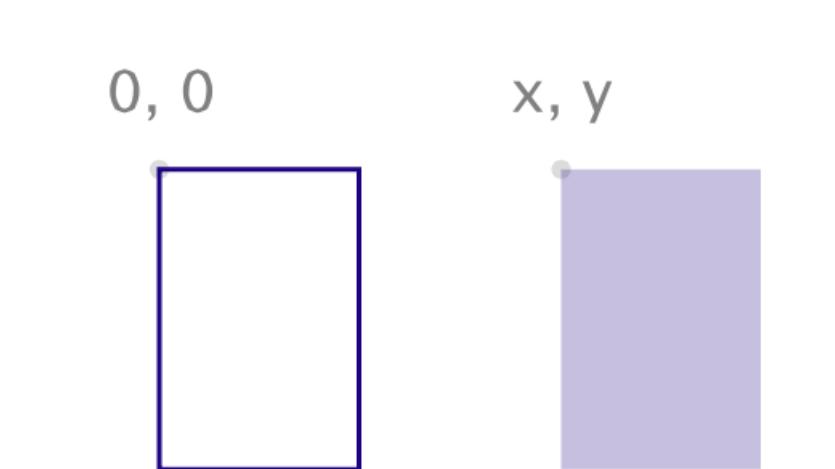
SkewX(a float64)



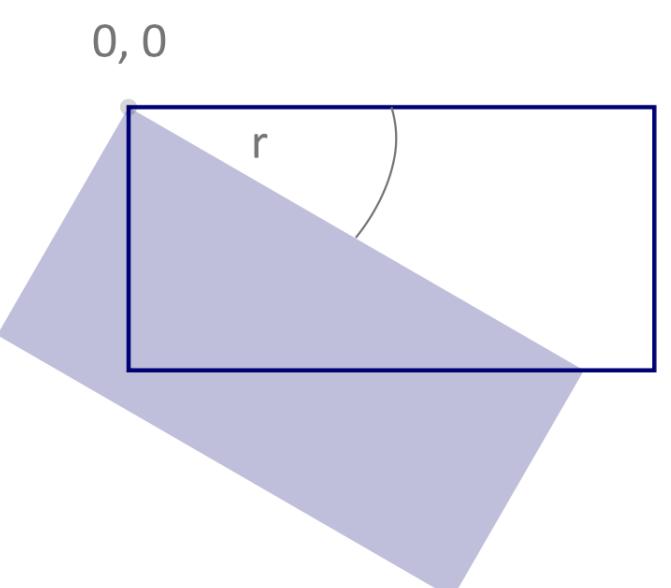
SkewY(a float64)



SkewXY(ax, ay float64)



Translate(x, y int)



Rotate(a float64)

Text and Images

hello, this is SVG
x, y

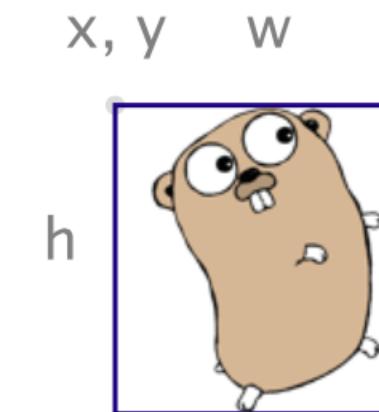
Text(x, y int, s string, s ...string)

It's "fine" & "dandy" to draw text along

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

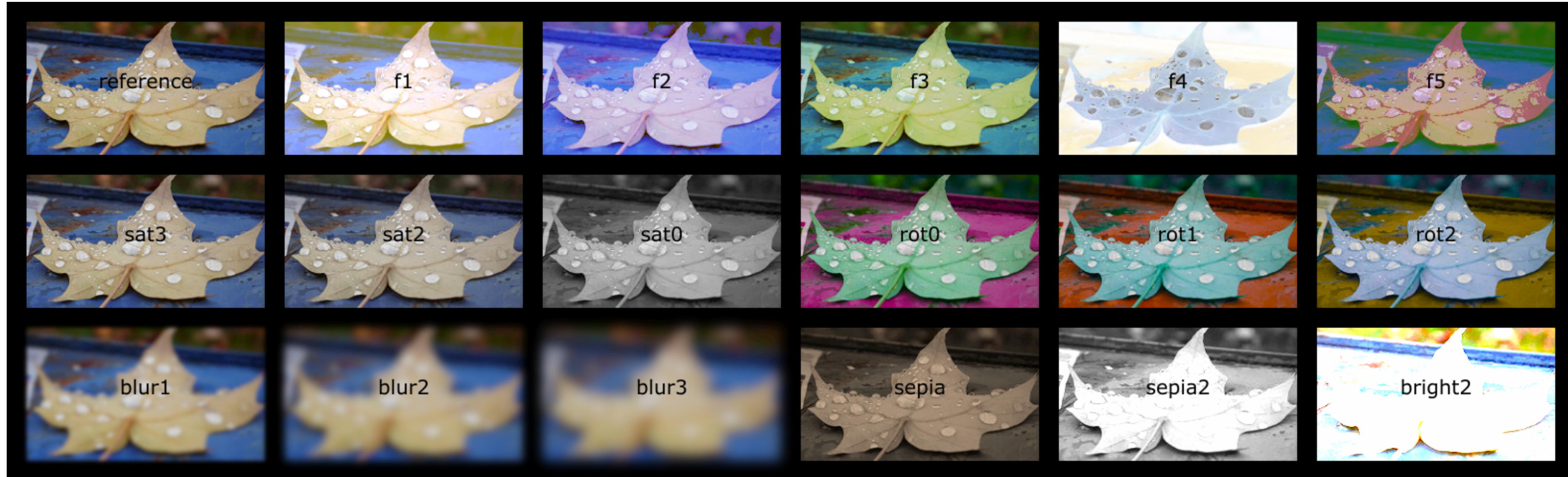
```
New(w io.Writer)  
Start(w, h int, options ...string)/End()  
Startview(w, h, minx, miny, vw, vh int)  
Group(s ...string)/Gend()  
Gstyle(s string)/Gend()
```

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



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

Filter Effects



Sketching with code

svgplay: SVGo sketching × +

localhost:1999

Apps PROTOTYPE -events | Data... Other bookmarks

```
package main

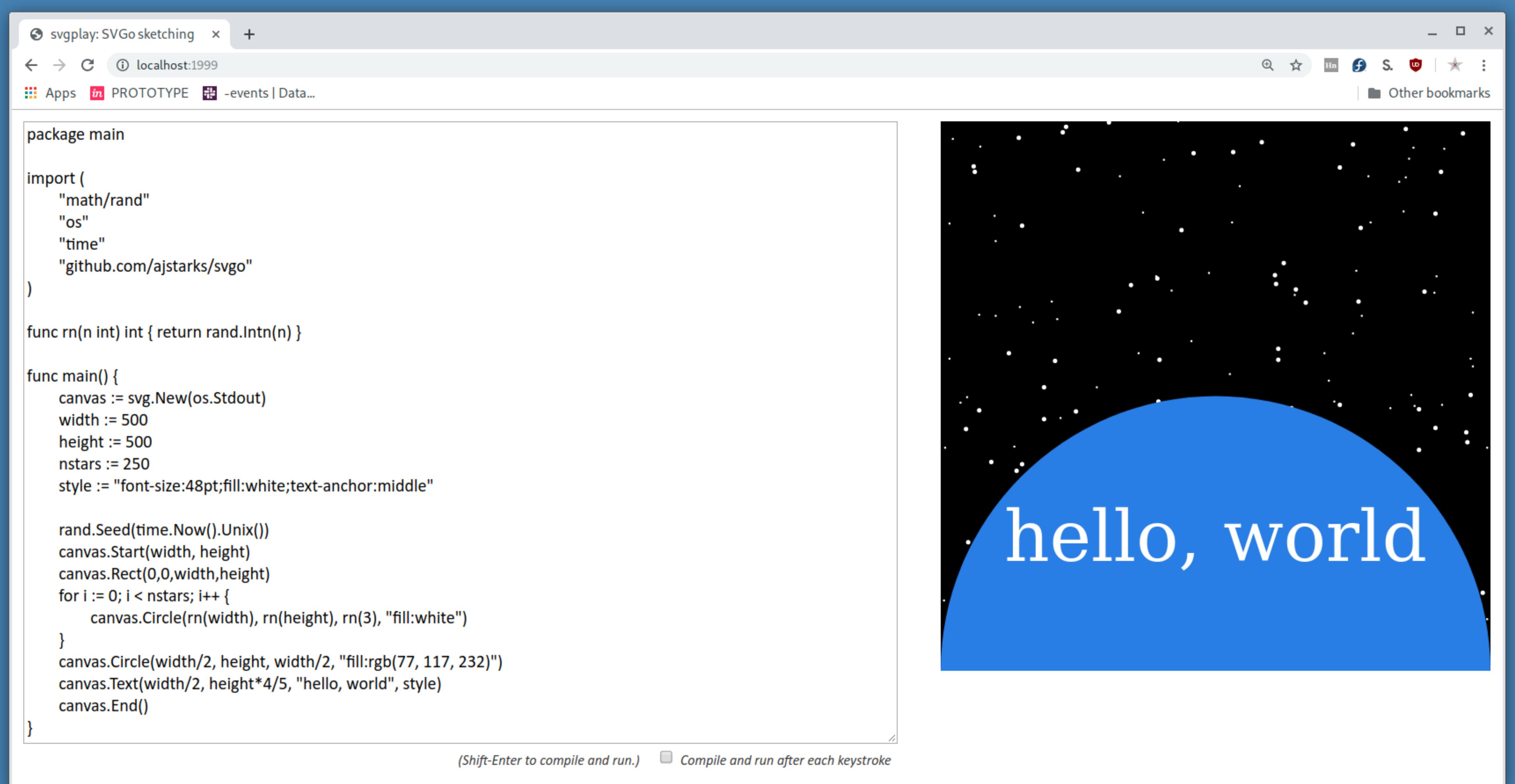
import (
    "math/rand"
    "os"
    "time"
    "github.com/ajstarks/svg"
)

func rn(n int) int { return rand.Intn(n) }

func main() {
    canvas := svg.New(os.Stdout)
    width := 500
    height := 500
    nstars := 250
    style := "font-size:48pt;fill:white;text-anchor:middle"

    rand.Seed(time.Now().Unix())
    canvas.Start(width, height)
    canvas.Rect(0,0,width,height)
    for i := 0; i < nstars; i++ {
        canvas.Circle(rn(width), rn(height), rn(3), "fill:white")
    }
    canvas.Circle(width/2, height, width/2, "fill:rgb(77, 117, 232)")
    canvas.Text(width/2, height*4/5, "hello, world", style)
    canvas.End()
}

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



ajstarks@slab:~

```
$ svgplay
2019/06/18 19:19:44 ⚠️⚠️⚠️ Warning: this server allows a client connecting to 127.0.0.1:1999 to execute code on this computer ⚠️⚠️⚠️
```

localhost:1999/hwsvg.go

```
package main

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

func main() {
    canvas := svg.New(os.Stdout)
    width := 700
    height := 500
    style := "font-size:72pt;fill:white;text-anchor:middle"

    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height)
    canvas.Circle(width/2, height, width/2, "fill:rgb(77, 117, 232)")
    canvas.Text(width/2, height*3/4, "hello, world", style)
    canvas.End()
}
```

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

The image shows a web-based development environment. On the left, there is a code editor window with a light gray background. It contains a Go program that generates an SVG image. The code imports the `svgo` library and creates a new SVG canvas from `os.Stdout`. It sets the width to 700 and height to 500 pixels. A blue circle is drawn at the center (width/2, height) with a radius of width/2, filled with the color `rgb(77, 117, 232)`. The text "hello, world" is centered inside the circle in white font, with a font size of 72pt and the text anchor set to middle. The code ends with `canvas.End()`.

```
localhost:1999/hwsvg.go
```

```
package main

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

func main() {
    canvas := svg.New(os.Stdout)
    width := 700
    height := 500
    style := "font-size:72pt;fill:white;text-anchor:middle"

    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height)
    canvas.Circle(width/2, height, width/2, "fill:rgb(255,0,0)")
    canvas.Text(width/2, height*3/4, "hello, Mars", style)
    canvas.End()
}
```

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

The image shows a web-based development environment. On the left, there is a code editor window with a light gray background. It contains a Go program that generates an SVG visualization. The code uses the `svg` library to create a red circle centered at (350, 375) with a radius of 350 pixels, filled with red. It also contains white text with a font size of 72pt that reads "hello, Mars". On the right, the resulting SVG visualization is displayed. It features a large red circle on a black background. Inside the red circle, the text "hello, Mars" is centered in a large, white, serif font.

Read/Parse/Draw Pattern

Read

<https://api.nytimes.com/svc/news/v3/content/all/world/.json?api-key=...&limit=5>

```
{  
  "status": "OK",  
  "copyright": "Copyright (c) 2019 The New York Times Company. All Rights Reserved.",  
  "num_results": 3313,  
  "results": [  
    {  
      "slug_name": "15notredame",  
      "section": "World",  
      "subsection": "Europe",  
      "title": "Worshiping in Hard Hats: Notre-Dame Offers First Mass Since Blaze",  
      "abstract": "A small group celebrated their faith and a fragile cathedral, which remains closed amid rebuilding.",  
      "url": "https://www.nytimes.com/2019/06/15/world/europe/notre-dame-mass.html",  
      "byline": "By ADAM NOSSITER",  
      "thumbnail_standard": "https://static01.nyt.com/images/2019/06/16/world/16notredame/16notredame-thumbStandard.jpg",  
      "item_type": "Article",  
      "source": "The New York Times",  
      "updated_date": "2019-06-15T16:20:30-04:00",  
      "created_date": "2019-06-15T16:04:47-04:00",  
      "published_date": "2019-06-14T20:00:00-04:00",  
      "first_published_date": "2019-06-15T16:03:17-04:00",  
      "material_type_facet": "News",  
      ...  
    },  
    {  
      "slug_name": "15dc-iran",  
      "section": "World",  
      "subsection": "Middle East",  
      "title": "In Face-Off With Iran, Escalation May Depend on Who Prevails Inside Washington and Tehran",  
      "abstract": "The attacks in the Gulf of Oman emboldens the hard-liners in Iran and the U.S., each able to argue their long-term adversary is itching for war.",  
      "url": "https://www.nytimes.com/2019/06/15/world/middleeast/trump-iran-hard-liners.html",  
      "byline": "By DAVID E. SANGER and DAVID D. KIRKPATRICK",  
      "thumbnail_standard": "https://static01.nyt.com/images/2019/06/16/us/16DC-IRAN/15dc-IRAN-thumbStandard.jpg",  
      "item_type": "Article",  
      "source": "The New York Times",  
      "updated_date": "2019-06-15T15:48:01-04:00",  
      "created_date": "2019-06-15T14:36:14-04:00",  
      "published_date": "2019-06-14T20:00:00-04:00",  
      "first_published_date": "2019-06-15T14:34:43-04:00",  
      "material_type_facet": "News Analysis",  
      "kicker": "News Analysis",  
      "subheadline": null,  
      ...  
    }  
  ]  
}
```

Parse

Draw



New York Times for Saturday Jun 15, 2019

Worshiping in Hard Hats: Notre-Dame Offers First Mass Since Blaze

In Face-Off With Iran, Escalation May Depend on Who Prevails Inside Washington and Tehran

Al-Bashir to Appear in Court to Face Graft Charges

How Hong Kong's Leader Made the Biggest Political Retreat by China Under Xi

Carrie Lam, Hong Kong's Leader, Retreats, but Her Critics Want More

```
package main

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

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

Imports

```
// API Info and formats
const (
    NYTAPIkey = "7f35400618208b235360ee874ed70df0"
    NYTfmt    = "http://api.nytimes.com/svc/news/v3/content/all/%s/"
    param     = ".json?api-key=%s&limit=5"
    style     = "font-size:20pt;font-family:sans-serif;fill:white"
    errfmt    = "unable to get network data for %s (%s)"
    datefmt   = "Monday Jan 2, 2006"
    usage     = "(arts, health, sports, science, technology, u.s., world")
)
```

Constants

```
// NYTHeadlines is the headline info from the New York Times
type NYTHeadlines struct {
    Status      string `json:"status"`
    Copyright   string `json:"copyright"`
    NumResults  int    `json:"num_results"`
    Results     []result `json:"results"`
}

type result struct {
    Section      string `json:"section"`
    Subsection   string `json:"subsection"`
    Title        string `json:"title"`
    Abstract     string `json:"abstract"`
    Thumbnail    string `json:"thumbnail_standard"`
}
```

Data Structures

```
func main() {
    var section = flag.String("h", "u.s.", usage)
    flag.Parse()
    canvas := svg.New(os.Stdout)
    canvas.Start(1200, 900)
    canvas.Rect(0, 0, 1200, 900)
    nytheadlines(canvas, *section)
    canvas.End()
}
```

Main

```
// netread dereferences 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
}
```

Network

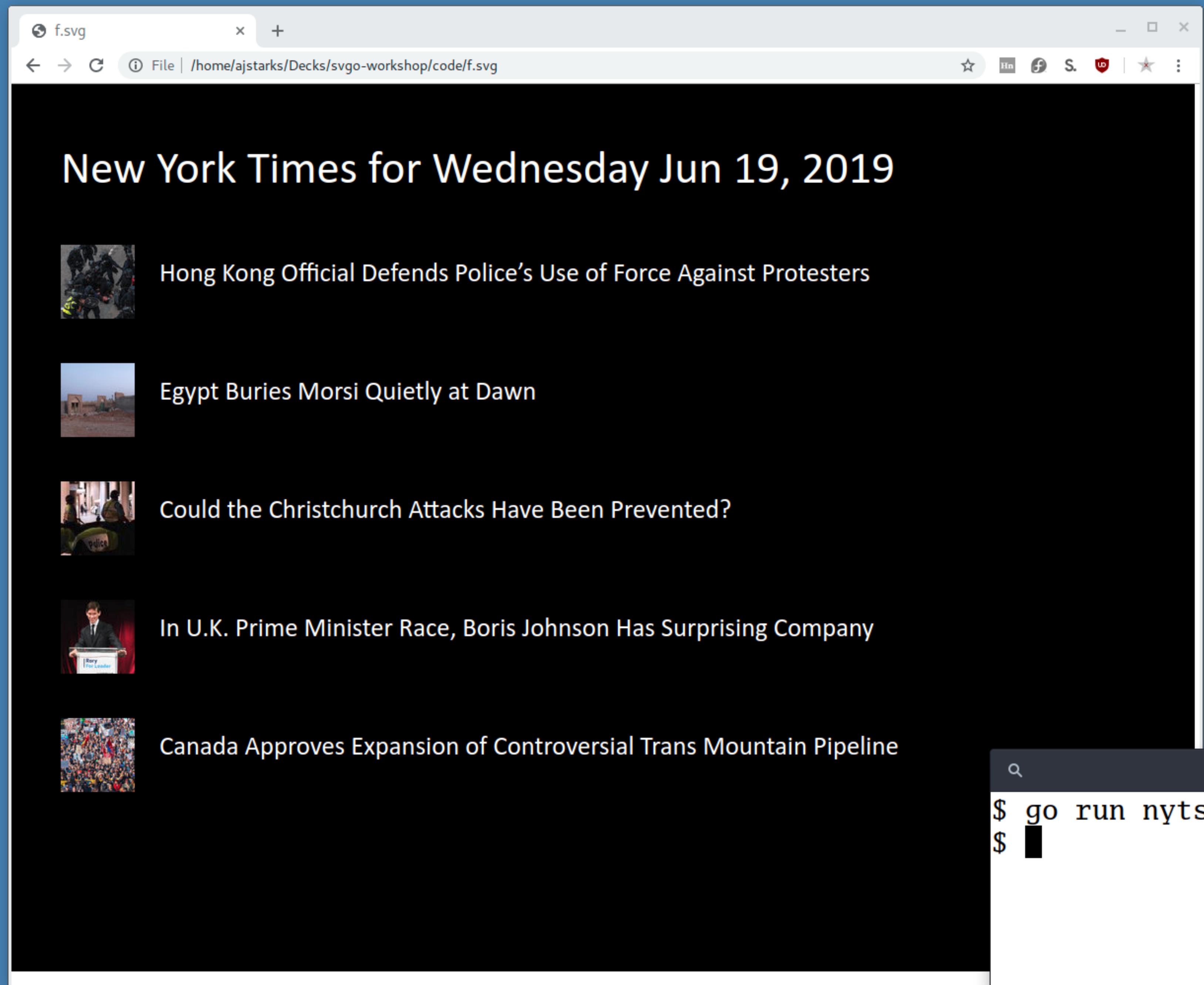
```
// nytheadlines retrieves data from the NYTimes API, decodes and displays it.
func nytheadlines(canvas *svg.SVG, section string) {
    // Read
    r, err := netread(fmt.Sprintf(NYTfmt+param, section, NYTAPIkey))
    if err != nil {
        fmt.Fprintf(os.Stderr, "headline read error: %v\n", err)
        return
    }
    defer r.Close()
    // Decode
    var data NYTHeadlines
    if err = json.NewDecoder(r).Decode(&data); err != nil {
        fmt.Fprintf(os.Stderr, "decode: %v\n", err)
        return
    }
    // drawing code next...
}
```

Read and Decode

```
// Positioning vars
thumbsize := 75
top, left := 200, 150
imx, titley := left - 100, top-100
x, y := left, top
ts := fmt.Sprintf("New York Times for %s", time.Now().Format(datefmt))

// Draw
canvas.Gstyle(style)
canvas.Text(imx, titley, ts, "font-size:175%")
for _, d := range data.Results {
    canvas.Text(x, y, d.Title)
    canvas.Image(imx, y-thumbsize/2, thumbsize, thumbsize, d.Thumbnail)
    y += 120
}
canvas.Gend()
```

Drawing



```
ajstarks@slab:~/Decks/svgo-workshop/code
$ go run nytsvg.go -h world > f.svg
$
```