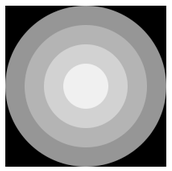
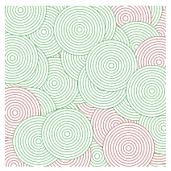
hello.go

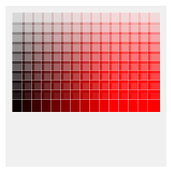
colorhash.go



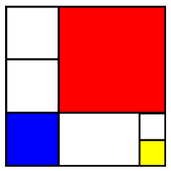
concentric.go



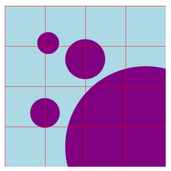
concentric2.go



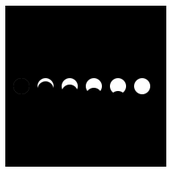
cgrid.go



mondrian.go



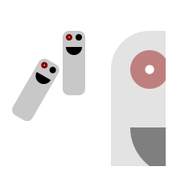
d4h.go



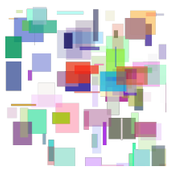
eclipse.go

cloud.go

color-clouds.go

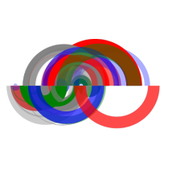


creepy.go

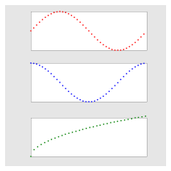


randbox.go

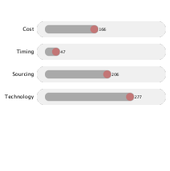
randspot.go



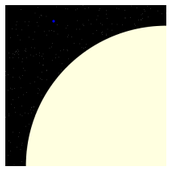
randarc.go



plotfunc.go

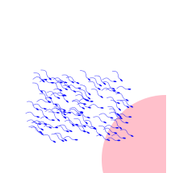


therm.go

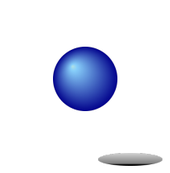


sunearth.go

conception.go

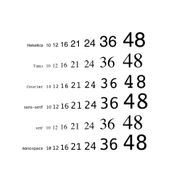


conception2.go



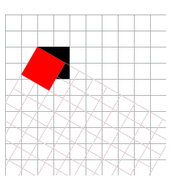
gradient.go

imfade.go

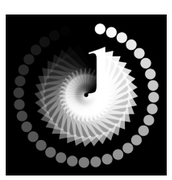


fontrange.go

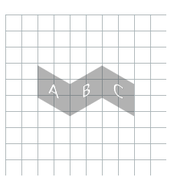
lorem.go



rotate.go



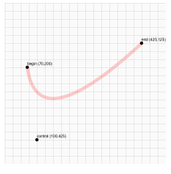
rotext.go



skewabc.go

gear.go

star.go



swoosh.go

textpath.go

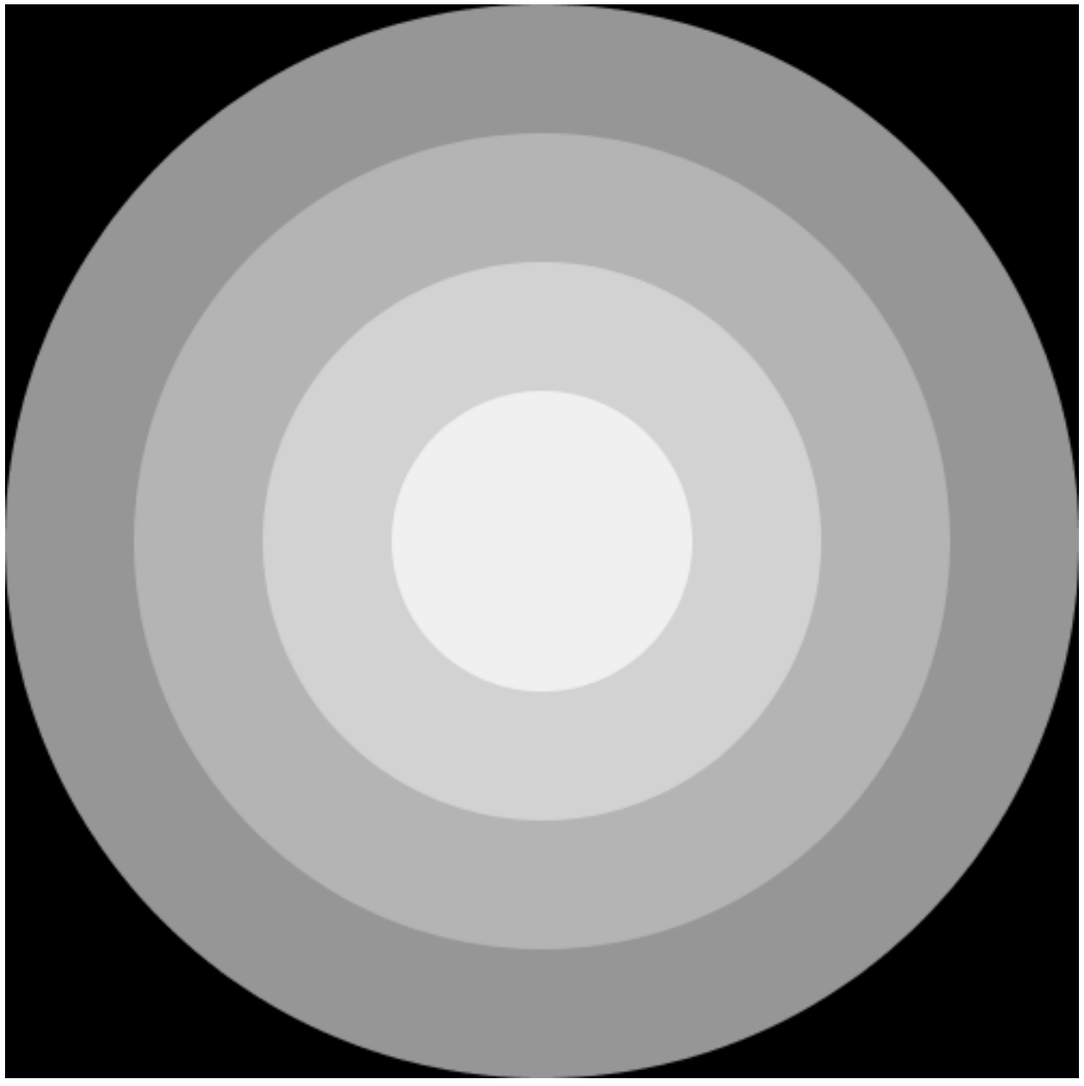
SVGo Examples

```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

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

func main() {
    style := "fill:white;font-size:48pt;text-anchor:middle"
    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/3, "hello, world", style)
    canvas.End()
}
```

hello.go [1]

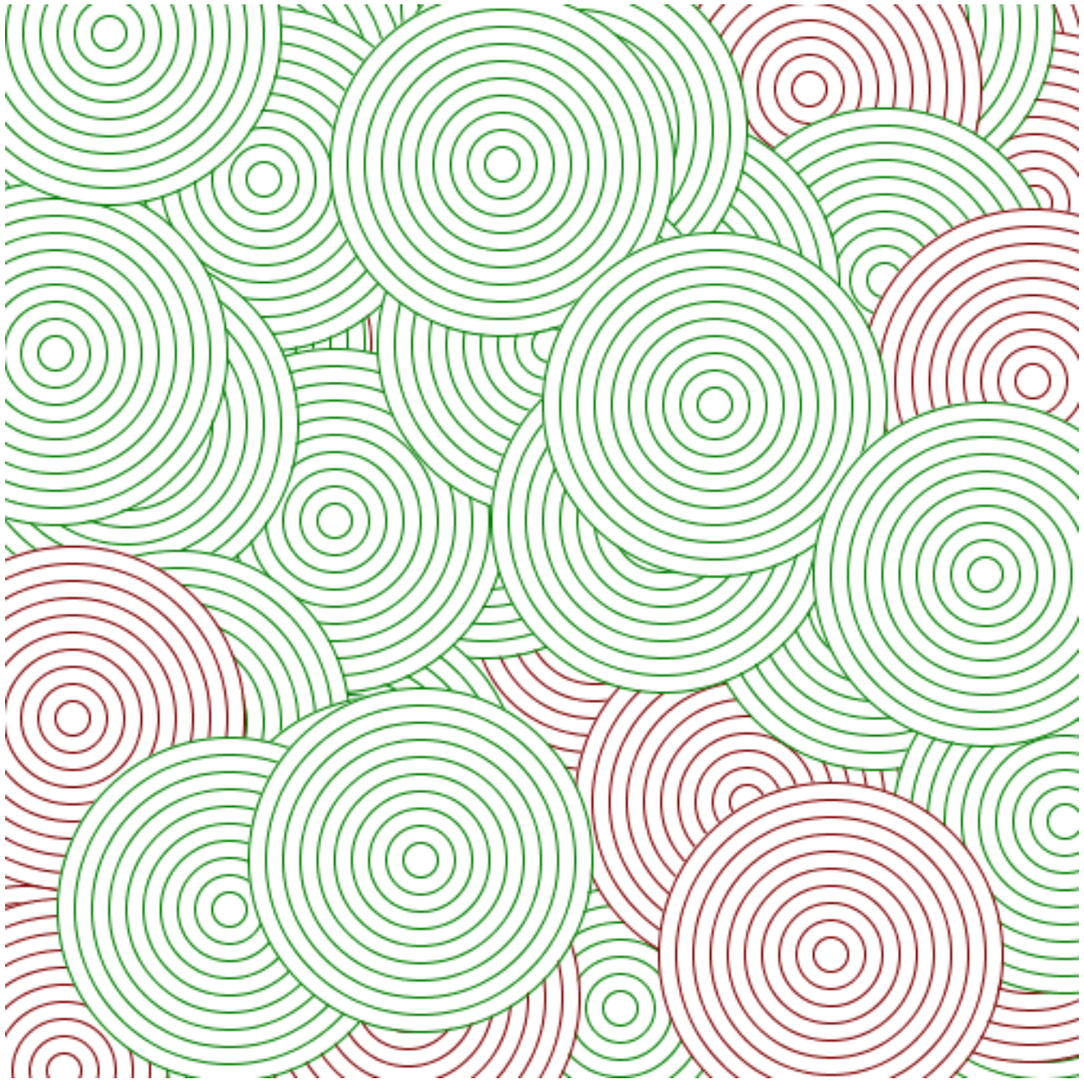
```
package main

import (
    "crypto/md5"
    "os"

    "github.com/ajstarks/svgo"
)

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

func colorhash(s string) (int, int, int) {
    hash := md5.New()
    hash.Write([]byte(s))
    v := hash.Sum(nil)
    return int(v[0]), int(v[1]), int(v[2])
}

func main() {
    name := "SVGo"
    style := "fill:white;text-anchor:middle;font-size:72pt"
    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height, canvas.RGB(colorhash(name)))
    canvas.Text(width/2, height/2, name, style)
    canvas.End()
}
```

```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

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

func main() {
    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height)
    r := height / 2
    for g := 150; g < 255; g += 30 {
        canvas.Circle(width/2, width/2, r, canvas.RGB(g, g, g))
        r -= 60
    }
    canvas.End()
}
```

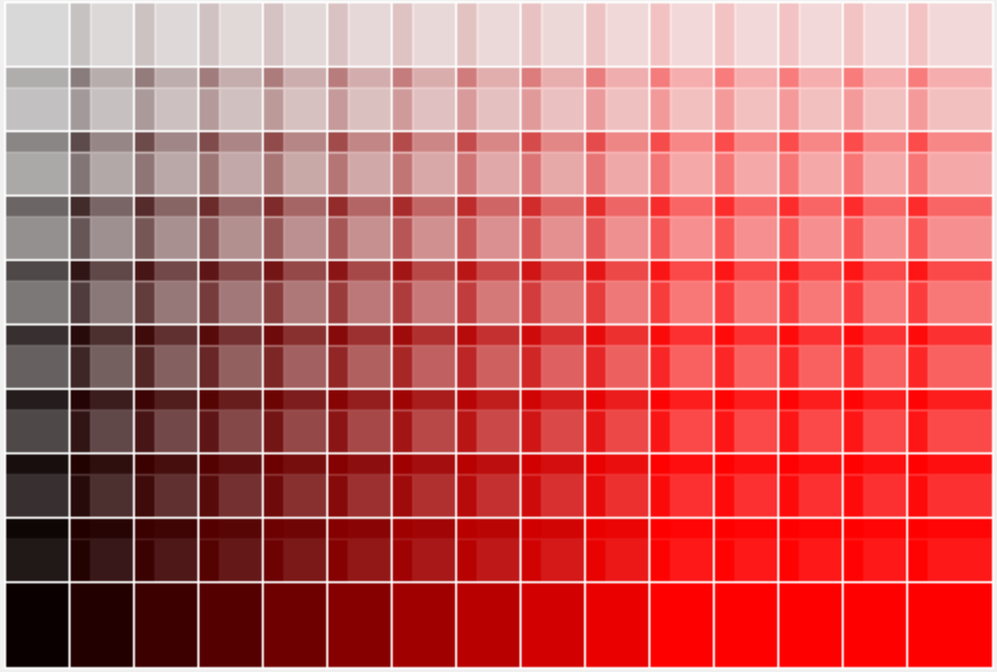


concentric.go [3]

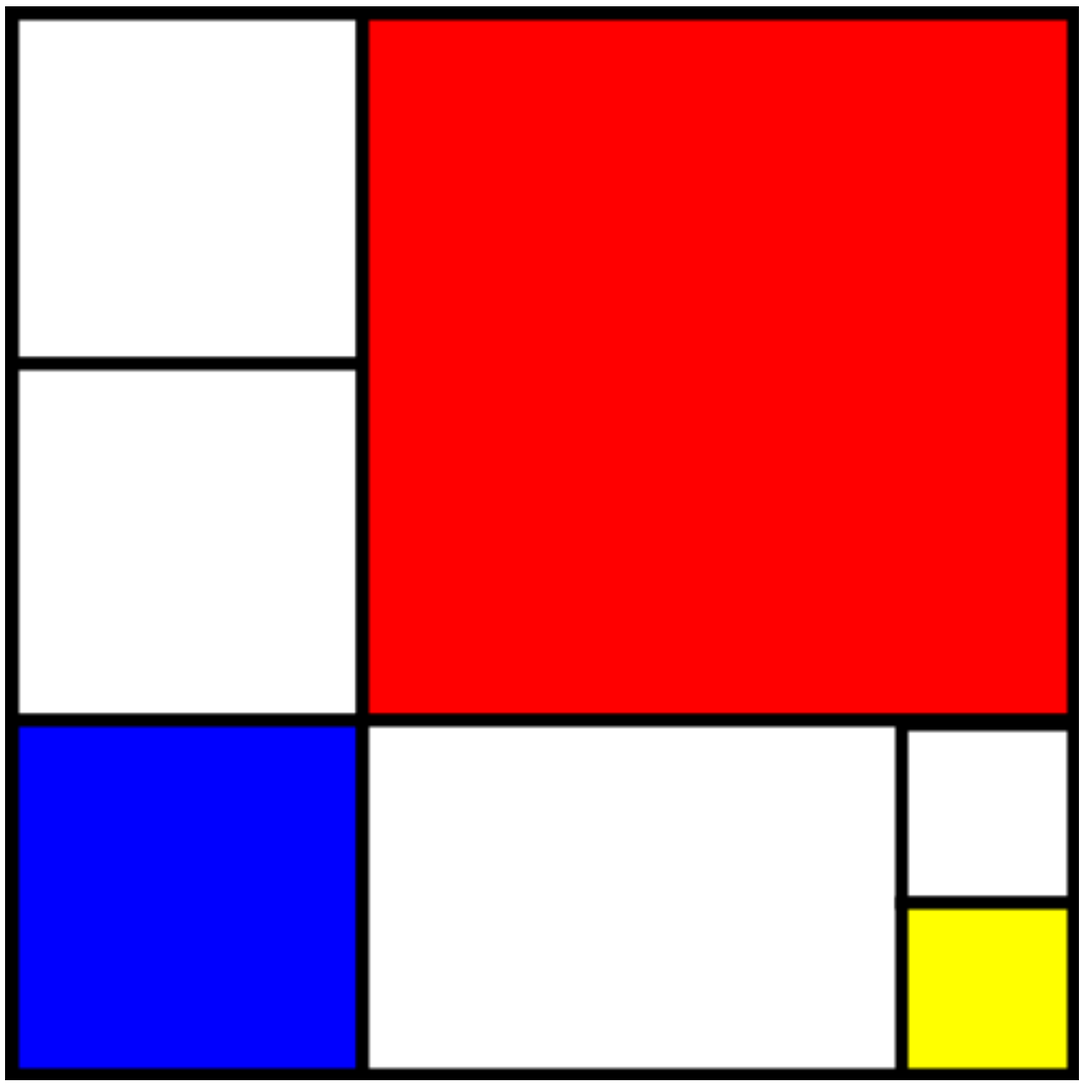
```
package main

import (
    "math/rand"
    "os"

    "github.com/ajstarks/svgo"
)

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

func main() {
    canvas.Start(width, height)
    canvas.Gstyle("fill:white")
    var color string
    radius := 80
    step := 8
    for i := 0; i < 200; i++ {
        if i%4 == 0 {
            color = "rgb(127,0,0)"
        } else {
            color = "rgb(0,127,0)"
        }
        x, y := rand.Intn(width), rand.Intn(height)
        for r, nc := radius, 0; nc < 10; nc++ {
            canvas.Circle(x, y, r, "stroke:"+color)
            r -= step
        }
    }
    canvas.Gend()
    canvas.End()
}
```



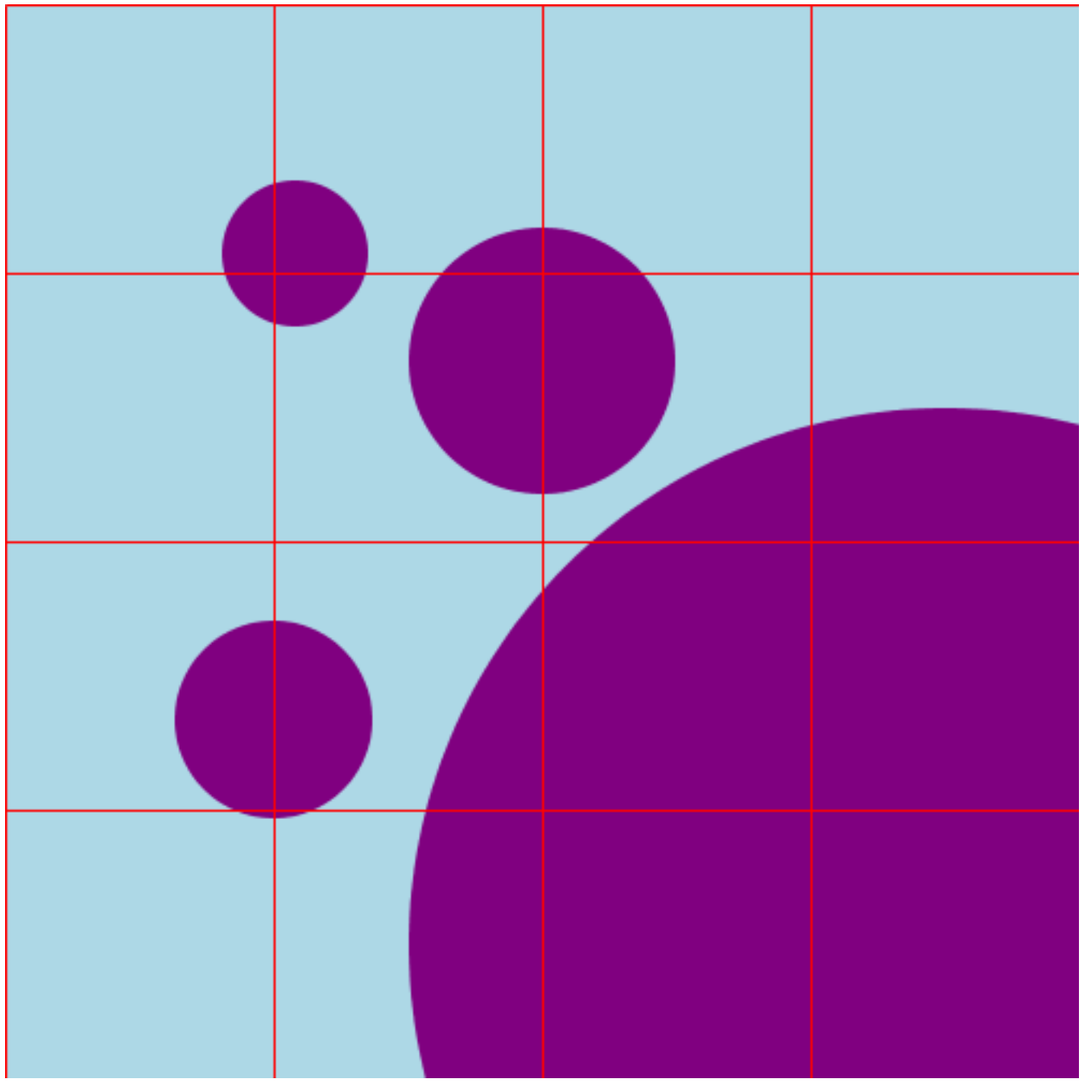
```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

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

func main() {
    y := 20
    v := 10
    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height, "fill:rgb(240,240,240)")
    canvas.Gstyle("stroke:white")
    for x := 20; x < 450; x += 30 {
        op := 0.1
        for i := 0; i < 100; i += 10 {
            canvas.Square(x, y, 20*2, canvas.RGBA(v, 0, 0, op))
            y += 30
            op += 0.1
        }
        y = 20
        v += 25
    }
    canvas.Gend()
    canvas.End()
}
```



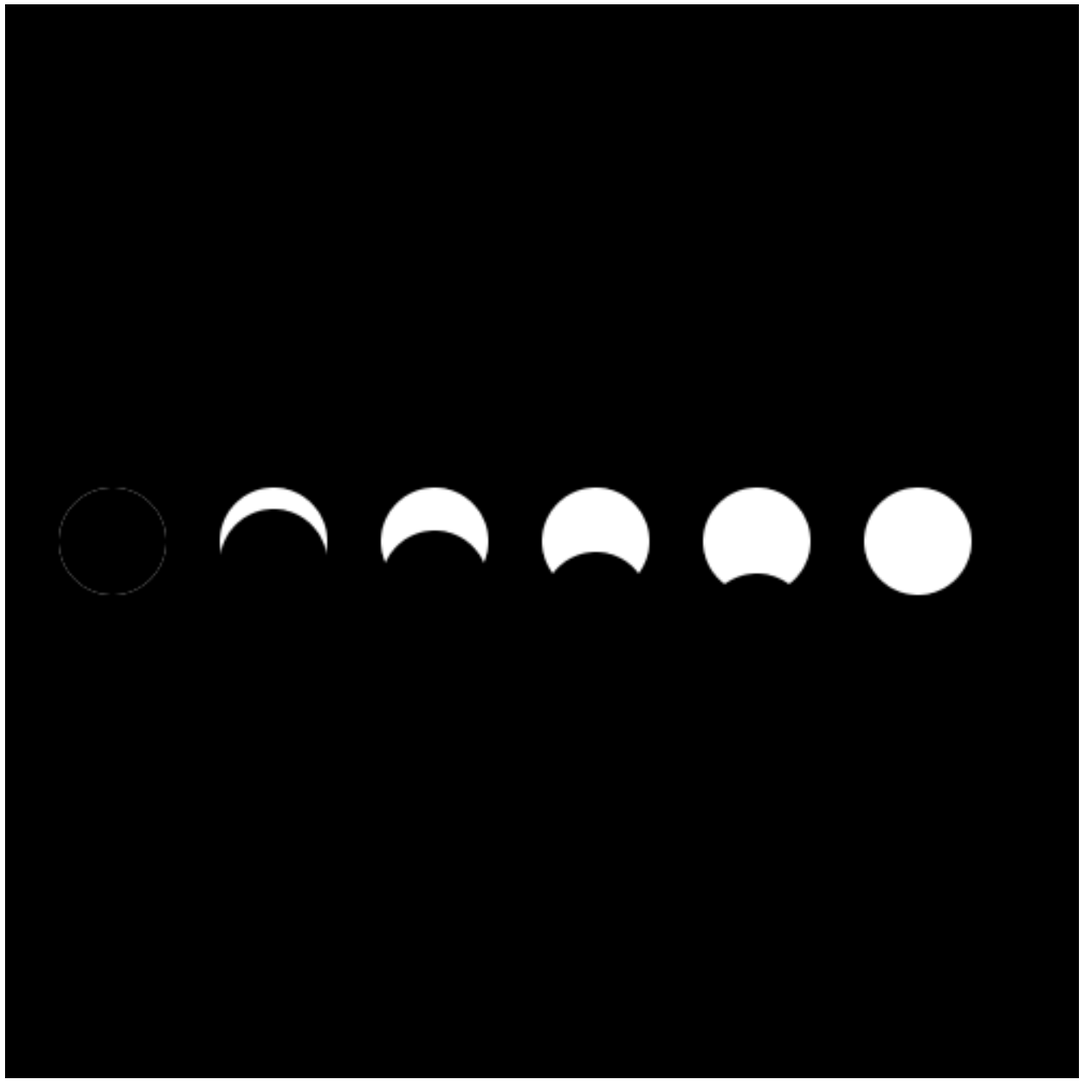
```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

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

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()
}
```



```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

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

func dot(x, y, d int) {
    canvas.Circle(x, y, d/2, "fill:rgb(128,0,128)")
}

// Composition from "Design for Hackers, pg. 129
func main() {
    d1 := height
    d2 := d1 / 4
    d3 := (d2 * 3) / 4
    d4 := (d3 * 3) / 4

    coffset := height / 8
    hoffset := height / (height / 10)
    voffset := -width / 10

    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height, "fill:lightblue")
    dot(width-coffset, height-coffset, d1)
    dot(width/2, height/3, d2)
    dot(width/4, height*2/3, d3)
    dot(width/4+hoffset, height/3+voffset, d4)
    canvas.Grid(0, 0, width, height, width/4, "stroke:red")
    canvas.End()
}
```



```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

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

func main() {
    h2 := height / 2
    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height)
    for x, y := 50, h2; x < 450; x += 75 {
        canvas.Ellipse(x, h2, 25, 25, "fill:white")
        canvas.Ellipse(x, y, 25, 25, "fill:black")
        y += 10
    }
    canvas.End()
}
```

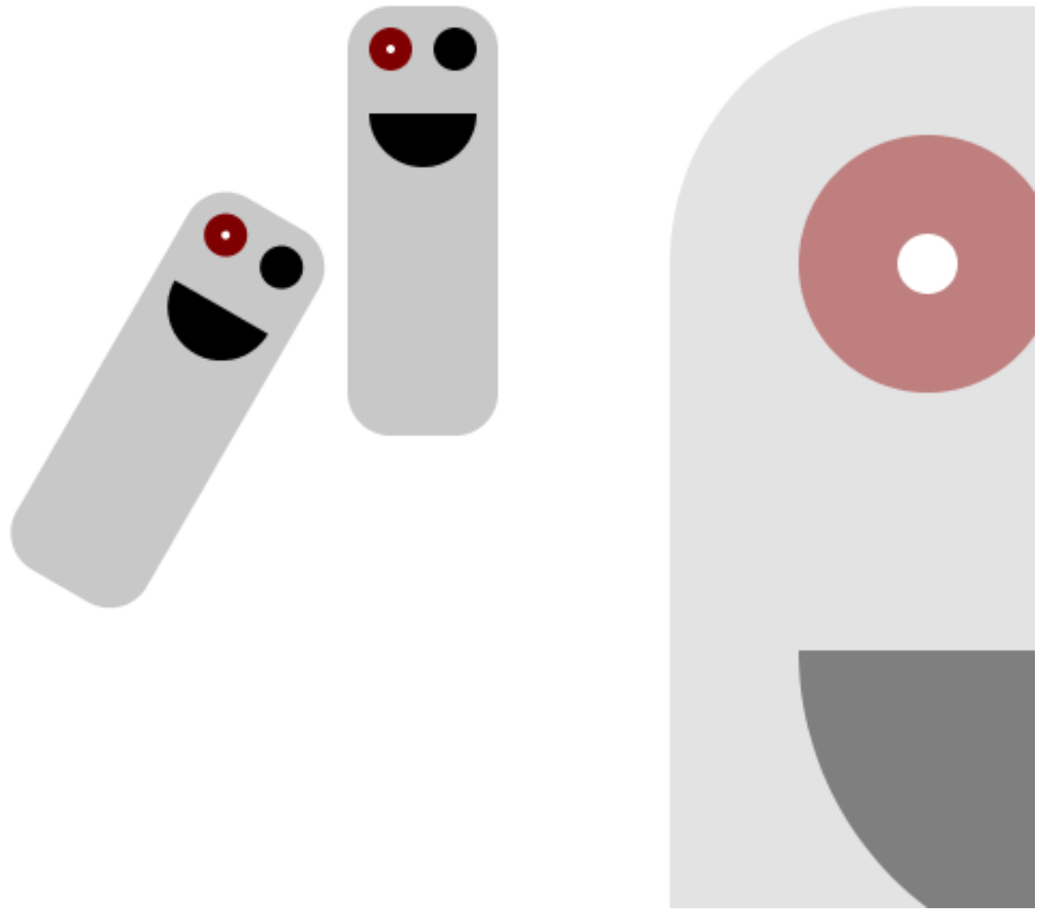


```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

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

func cloud(x, y, r int, style string) {
    small := r / 2
    medium := (r * 6) / 10
    canvas.Gstyle(style)
    canvas.Circle(x, y, r)
    canvas.Circle(x+r, y+small, small)
    canvas.Circle(x-r-small, y+small, small)
    canvas.Circle(x-r, y, medium)
    canvas.Rect(x-r-small, y, r*2+small, r)
    canvas.Gend()
}

func main() {
    canvas.Start(width, height)
    cloud(width/2, height/2, 100, canvas.RGB(127, 127, 127))
    canvas.End()
}
```

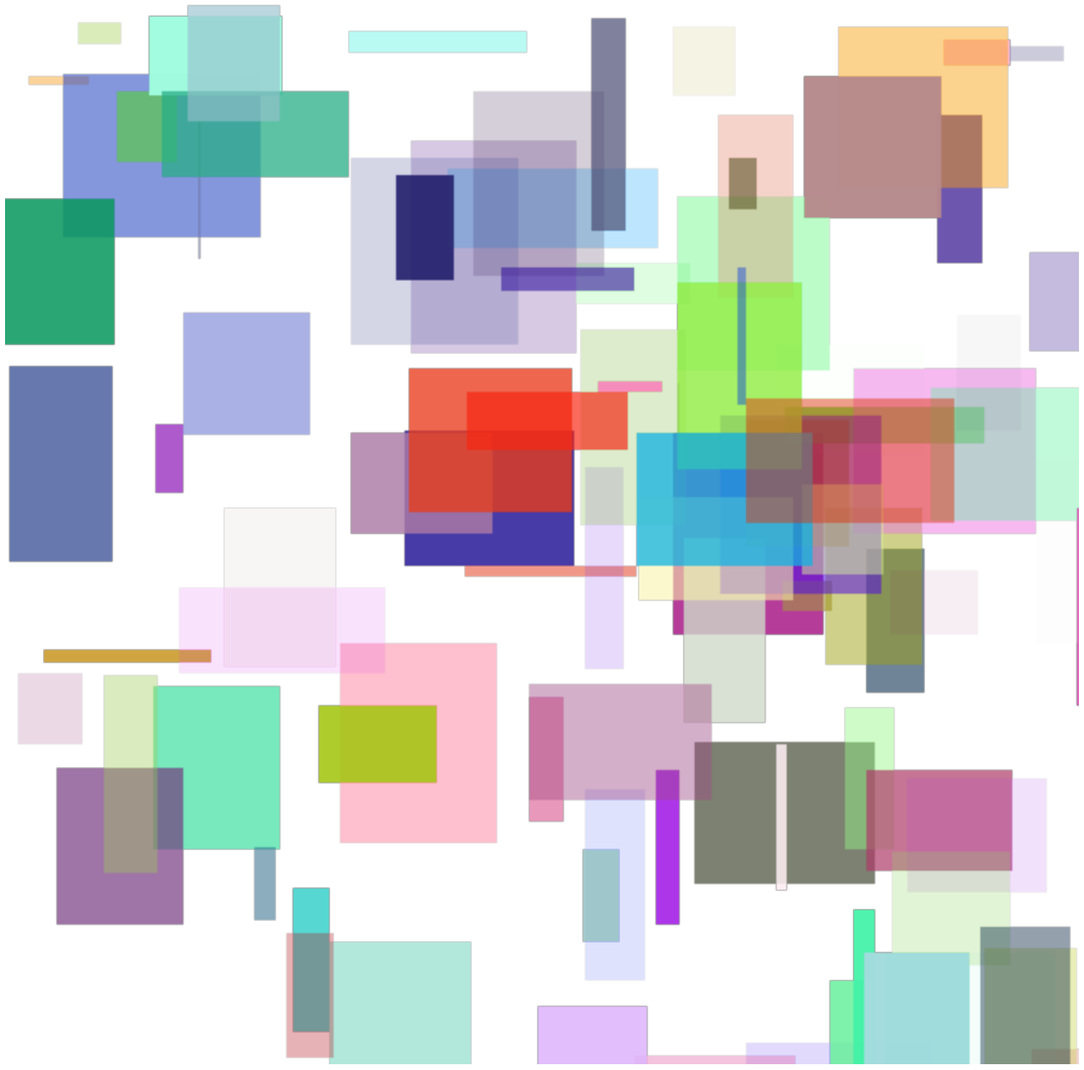
```
package main

import (
    "math/rand"
    "os"
    "time"

    "github.com/ajstarks/svgo"
)

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

func cloud(x, y, r int, style string) {
    small := r / 2
    medium := (r * 6) / 10
    canvas.Gstyle(style)
    canvas.Circle(x, y, r)
    canvas.Circle(x+r, y+small, small)
    canvas.Circle(x-r-small, y+small, small)
    canvas.Circle(x-r, y, medium)
    canvas.Rect(x-r-small, y, r*2+small, r)
    canvas.Gend()
}

func main() {
    rand.Seed(time.Now().Unix())
    canvas.Start(width, height)
    for i := 0; i < 50; i++ {
        red := rand.Intn(255)
        green := rand.Intn(255)
        blue := rand.Intn(255)
        size := rand.Intn(60)
        x := rand.Intn(width)
        y := rand.Intn(height)
        cloud(x, y, size, canvas.RGB(red, green, blue))
    }
    canvas.End()
}
```

color-clouds.go [10]

```
package main

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

var canvas = svg.New(os.Stdout)

func smile(x, y, r int) {
    r2 := r * 2
    r3 := r * 3
    r4 := r * 4
    rq := r / 4
    gray := canvas.RGB(200, 200, 200)
    red := canvas.RGB(127, 0, 0)
    canvas.Roundrect(x-r2, y-r2, r*7, r*20, r2, r2, gray)
    canvas.Circle(x, y, r, red)
    canvas.Circle(x, y, rq, "fill:white")
    canvas.Circle(x+r3, y, r)
    canvas.Arc(x-r, y+r3, rq, rq, 0, true, false, x+r4, y+r3)
}

func main() {
    canvas.Start(500, 500)
    canvas.Rect(0, 0, 500, 500, "fill:white")
    smile(200, 100, 10)
    canvas.Gtransform("rotate(30)")
    smile(200, 100, 10)
    canvas.Gend()
    canvas.Gtransform("translate(50,0) scale(2,2)")
    canvas.Gstyle("opacity:0.5")
    smile(200, 100, 30)
    canvas.Gend()
    canvas.Gend()
    canvas.End()
}
```

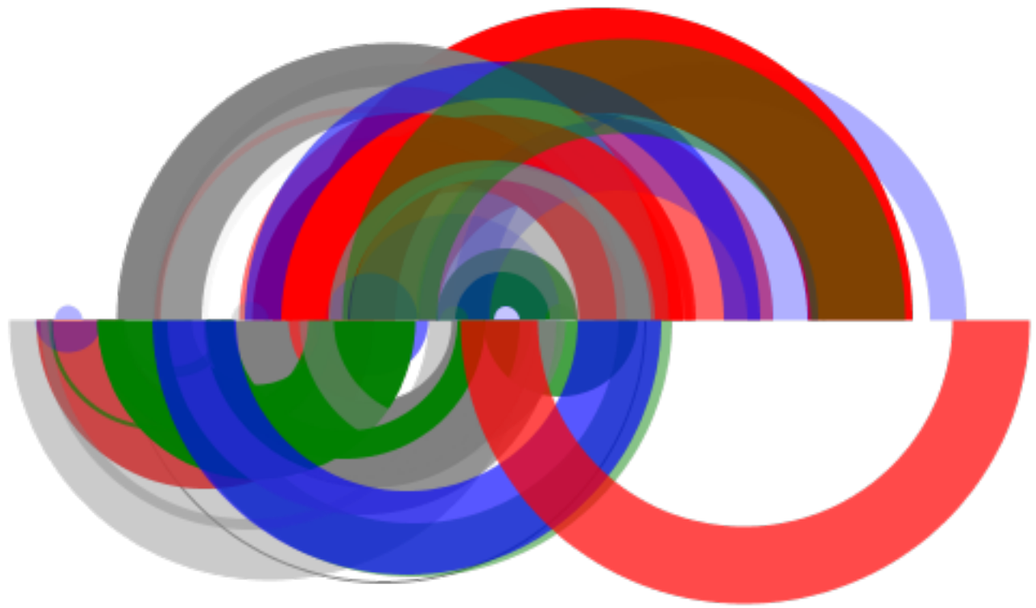


```
package main

import (
    "math/rand"
    "os"
    "time"

    "github.com/ajstarks/svgo"
)

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

func main() {
    canvas.Start(width, height)
    rand.Seed(time.Now().Unix())
    for i := 0; i < 100; i++ {
        fill := canvas.RGBA(
            rand.Intn(255),
            rand.Intn(255),
            rand.Intn(255),
            rand.Float64())
        canvas.Rect(
            rand.Intn(width),
            rand.Intn(height),
            rand.Intn(100),
            rand.Intn(100),
            fill)
    }
    canvas.End()
}
```

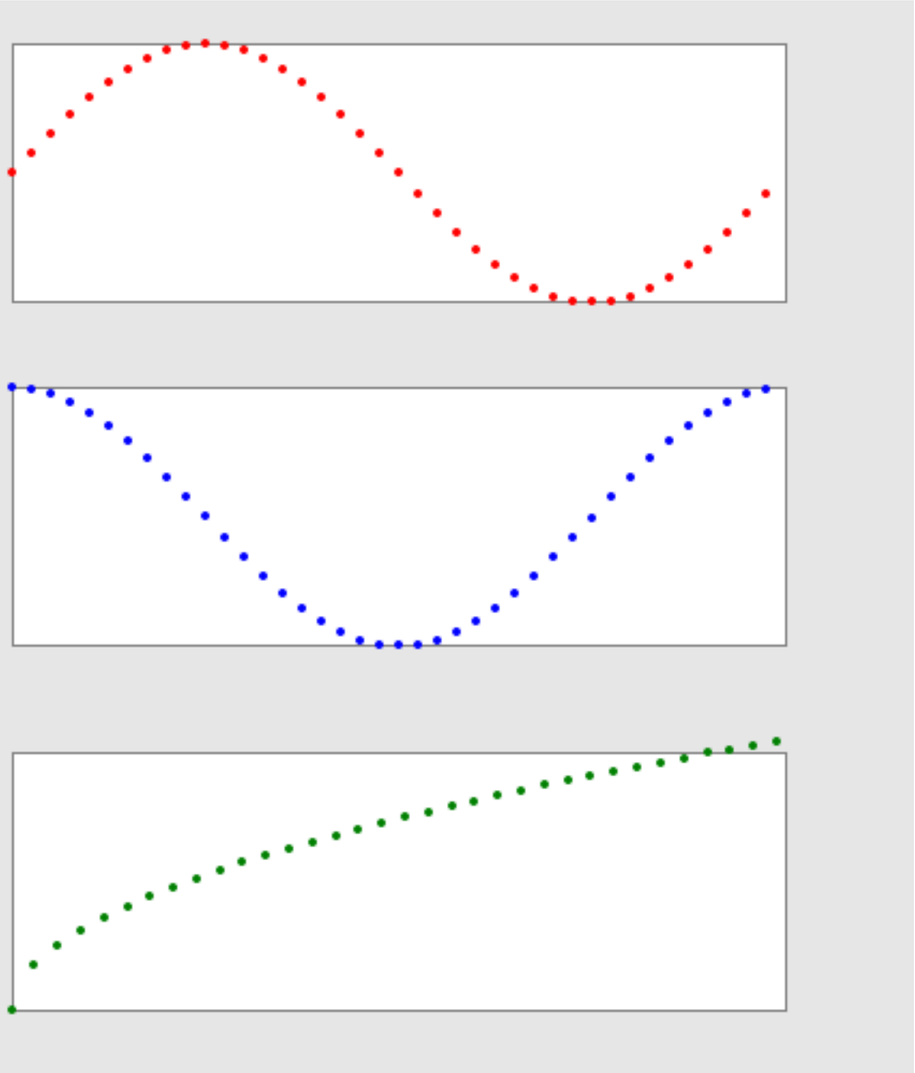


```
package main

import (
    "math/rand"
    "os"
    "time"

    "github.com/ajstarks/svgo"
)

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

func main() {
    canvas.Start(width, height)
    rand.Seed(time.Now().Unix())
    for i := 0; i < 100; i++ {
        fill := canvas.RGBA(
            rand.Intn(255),
            rand.Intn(255),
            rand.Intn(255),
            rand.Float64())
        canvas.Ellipse(
            rand.Intn(width),
            rand.Intn(height),
            rand.Intn(100),
            rand.Intn(100),
            fill)
    }
    canvas.End()
}
```

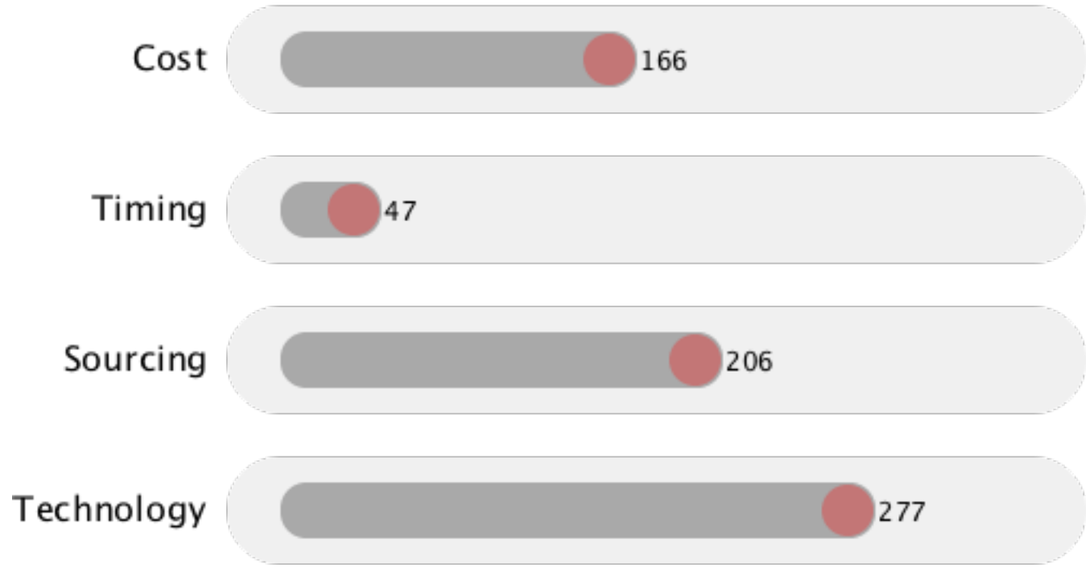
```
package main

import (
    "fmt"
    "math/rand"
    "os"
    "time"

    "github.com/ajstarks/svgo"
)

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

func randarc(aw, ah, sw int, f1, f2 bool) {
    colors := []string{"red", "green", "blue", "gray"}
    afmt := "stroke:%s;stroke-opacity:%.2f;stroke-width:%dpx;fill:none"
    begin, arclength := rand.Intn(aw), rand.Intn(aw)
    end := begin + arclength
    baseline := ah / 2
    al, cl := arclength/2, len(colors)
    canvas.Arc(begin, baseline, al, al, 0, f1, f2, end, baseline,
        fmt.Sprintf(afmt, colors[rand.Intn(cl)], rand.Float64(), rand.Intn(sw)))
}

func main() {
    rand.Seed(time.Now().Unix())
    canvas.Start(width, height)
    aw := width / 2
    maxstroke := height / 10
    for i := 0; i < 20; i++ {
        randarc(aw, height, maxstroke, false, true)
        randarc(aw, height, maxstroke, false, false)
    }
    canvas.End()
}
```



```
package main

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

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

func vmap(value float64, l1 float64, h1 float64,
        l2 float64, h2 float64) float64 {
    return l2 + (h2-l2)*(value-l1)/(h1-l1)
}

func plotfunc(left, top, w, h int, style string,
    min, max, fmin, fmax, interval float64, f func(float64) float64) {
    canvas.Translate(0, top)
    canvas.Rect(left, 0, w, h, "fill:white;stroke:gray")
    for x := min; x < max; x += interval {
        dx := int(vmap(x, min, max, float64(left), float64(w+left)))
        dy := int(vmap(f(x), fmin, fmax, 0, float64(h)))
        canvas.Translate(0, (h - height))
        canvas.Circle(dx, height-dy, 2, style)
        canvas.Gend()
    }
    canvas.Gend()
}

func main() {
    const TwoPi = 2 * math.Pi
    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height, "fill:rgb(230,230,230)")
    plotfunc(80, 20,  360, 120, "fill:red",   0, TwoPi, -1, 1, math.Pi/20, math.Sin)
    plotfunc(80, 180, 360, 120, "fill:blue",  0, TwoPi, -1, 1, math.Pi/20, math.Cos)
    plotfunc(80, 350, 360, 120, "fill:green", 0, 10, 0, 3, 0.3, math.Sqrt)
    canvas.End()
}
```

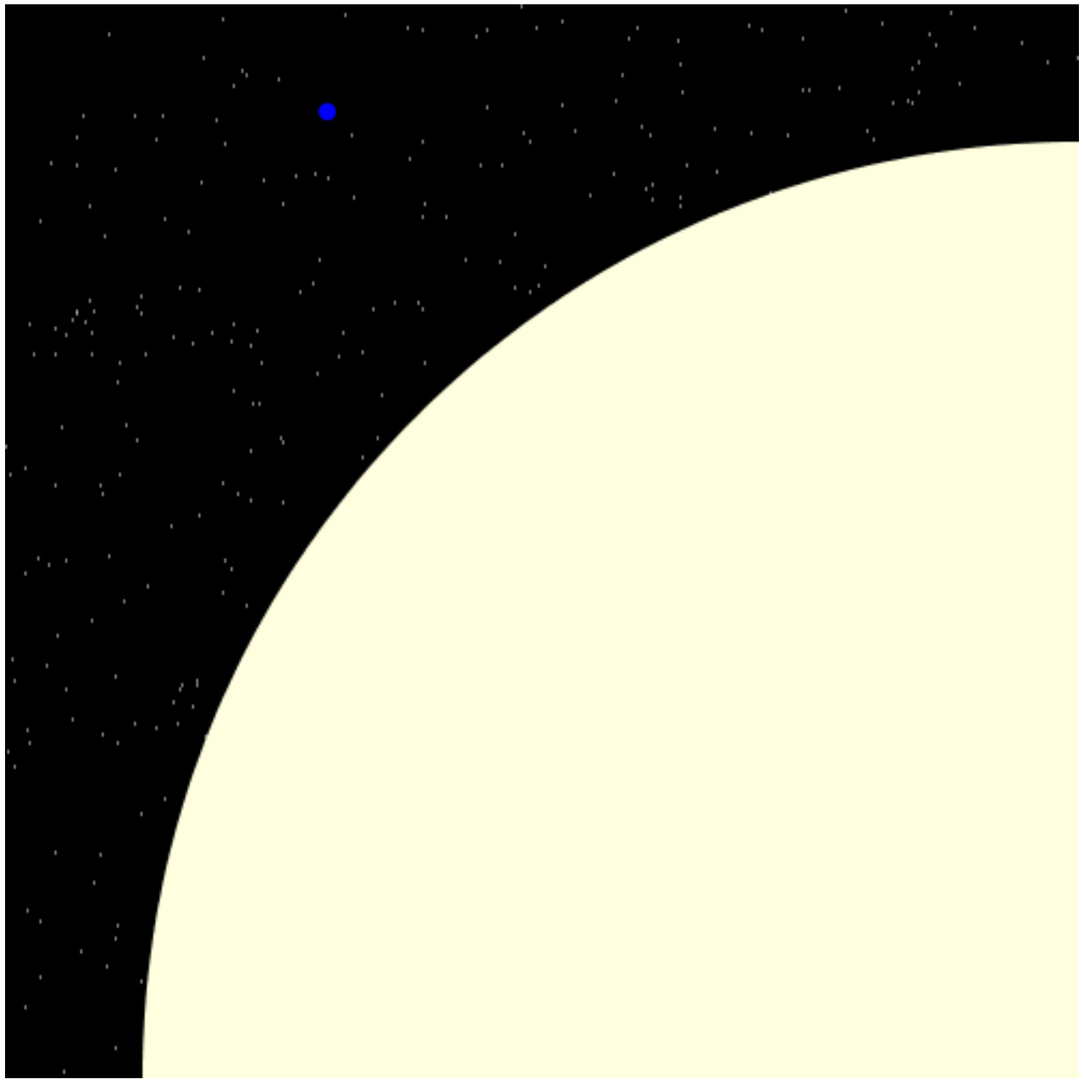


```
package main

import (
    "fmt"
    "math/rand"
    "os"
    "time"

    "github.com/ajstarks/svgo"
)

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

func meter(x, y, w, h, value int, label string) {
    corner := h / 2
    inset := corner / 2
    canvas.Text(x-10, y+h/2, label, "text-anchor:end;baseline-shift:-33%")
    canvas.Roundrect(x, y, w, h, corner, corner, "fill:rgb(240,240,240)")
    canvas.Roundrect(x+corner, y+inset, value, h-(inset*2), inset, inset, "fill:darkgray")
    canvas.Circle(x+inset+value, y+corner, inset, "fill:red;fill-opacity:0.3")
    canvas.Text(x+inset+value+inset+2, y+h/2, fmt.Sprintf("%-3d", value), "font-size:75%;text-anchor:start;baseline-shift:-33%")
}

func main() {
    rand.Seed(time.Now().Unix())
    items := []string{"Cost", "Timing", "Sourcing", "Technology"}
    mh, gutter := 50, 20
    x, y := 100, 50
    canvas.Start(width, height)
    canvas.Gstyle("font-family:sans-serif;font-size:12pt")
    for _, data := range items {
        meter(x, y, width-100, mh, rand.Intn(300), data)
        y += mh + gutter
    }
    canvas.Gend()
    canvas.End()
}
```



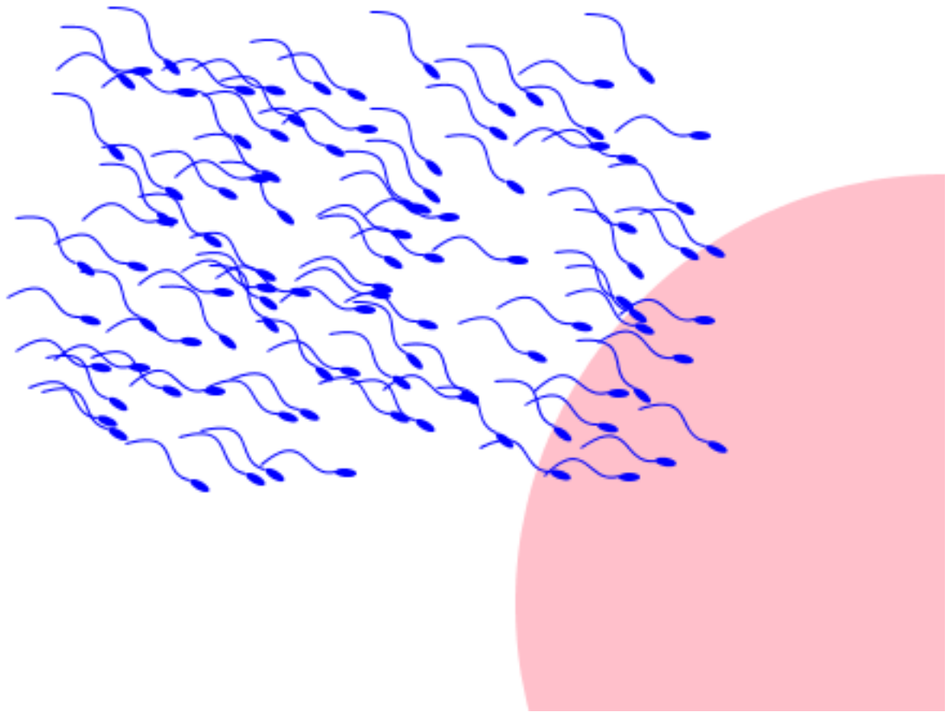
therm.go [16]

```
package main

import (
    "math/rand"
    "os"

    "github.com/ajstarks/svgo"
)

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

func main() {
    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height, "fill:black")
    for i := 0; i < width; i++ {
        x := rand.Intn(width)
        y := rand.Intn(height)
        canvas.Line(x, y, x, y+1, "stroke:white")
    }
    earth := 4
    sun := earth * 109
    canvas.Circle(150, 50, earth, "fill:blue")            // earth
    canvas.Circle(width, height, sun, "fill:lightyellow") // sun
    canvas.End()
}
```

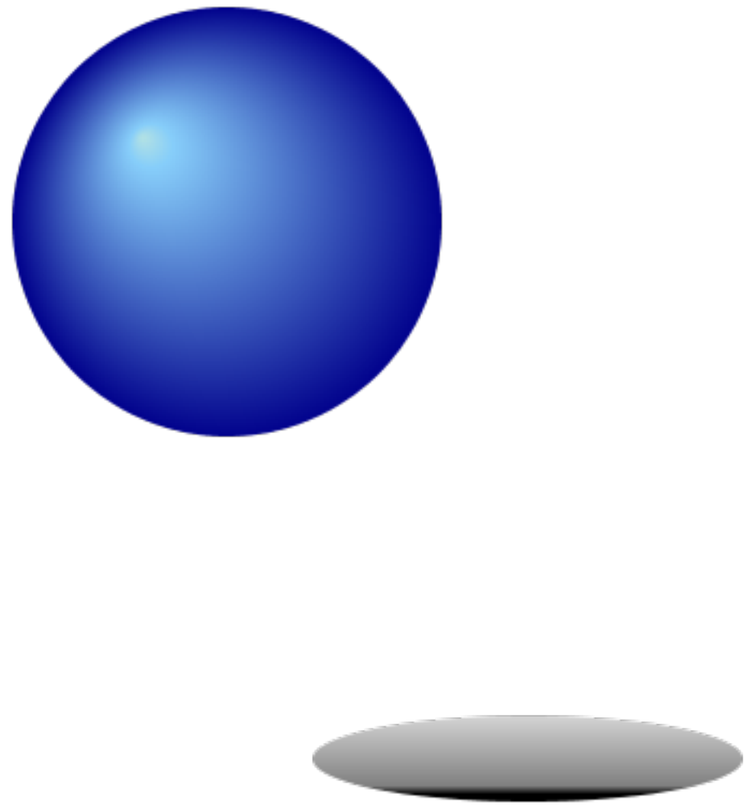


```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

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

func male(x, y, w int) {
    canvas.Ellipse(x, y, w, w/2, "fill:blue")
    canvas.Bezier(
        x-(w*8), y,
        x-(w*4), y-(w*4),
        x-(w*4), y+w,
        x-w, y, "stroke:blue;fill:none")
}

func female(x, y, w int) {
    canvas.Circle(x, y, w, "fill:pink")
}

func main() {
    msize := 5
    fsize := msize * 40
    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height, "fill:white")
    female(width, height-50, fsize)
    male(100, 200, msize)
    canvas.End()
}
```

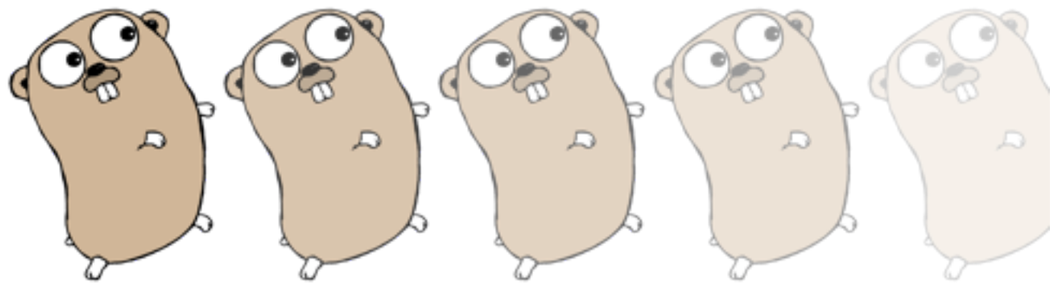
```
package main

import (
    "math/rand"
    "os"

    "github.com/ajstarks/svgo"
)

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

func male(x, y, w int) {
    canvas.Ellipse(x, y, w, w/2, "fill:blue")
    canvas.Bezier(
        x-(w*8), y,
        x-(w*4), y-(w*4),
        x-(w*4), y+w,
        x-w, y, "stroke:blue;fill:none")
}

func female(x, y, w int) {
    canvas.Circle(x, y, w, "fill:pink")
}

func main() {
    msize := 5
    fsize := msize * 40
    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height, "fill:white")
    female(width, height-50, fsize)
    for i := 0; i < 100; i++ {
        canvas.TranslateRotate(rand.Intn(300)+100, rand.Intn(200)+200, rand.Float64()*45)
        male(0, 0, msize)
        canvas.Gend()
    }
    canvas.End()
}
```



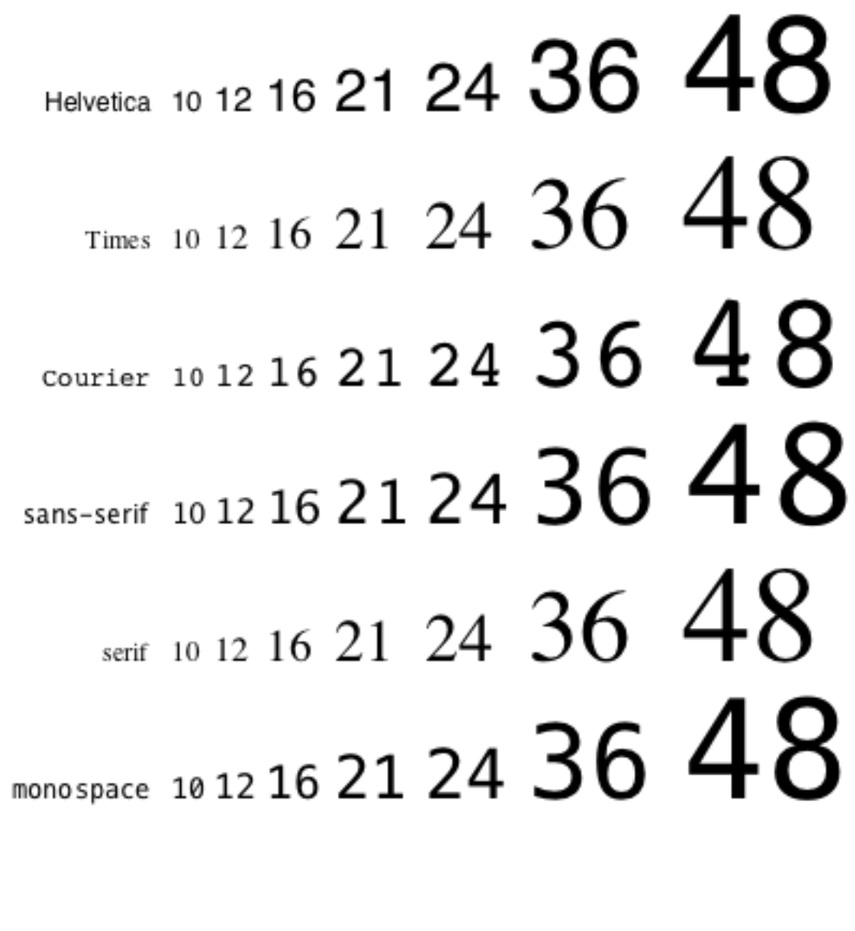
```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

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

func main() {
    rg := []svg.Offcolor{
        {1, "powderblue", 1},
        {10, "lightskyblue", 1},
        {100, "darkblue", 1},
    }
    lg := []svg.Offcolor{
        {10, "black", 1},
        {20, "gray", 1},
        {100, "lightgray", 1},
    }
    canvas.Start(width, height)
    canvas.Def()
    canvas.RadialGradient("rg", 50, 50, 50, 30, 30, rg)
    canvas.LinearGradient("lg", 0, 100, 0, 0, lg)
    canvas.DefEnd()
    canvas.Circle(width/2, height-300, 100, "fill:url(#rg)")
    canvas.Ellipse(width-110, height-50, 100, 20, "fill:url(#lg)")
    canvas.End()
}
```



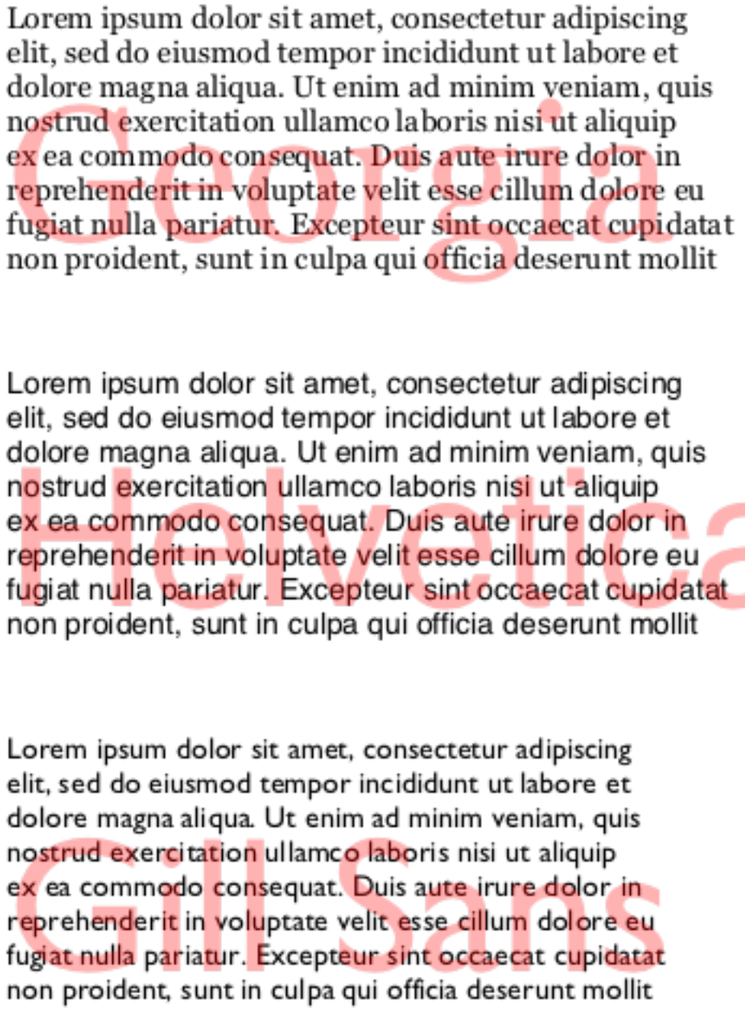
```
package main

import (
    "fmt"
    "os"

    "github.com/ajstarks/svgo"
)

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

func main() {
    canvas.Start(width, height)
    opacity := 1.0
    for x := 0; x < width; x += 100 {
        canvas.Image(x, 100, 128, 128, "gophercolor128x128.png", fmt.Sprintf("opacity:%.2f", opacity))
        opacity -= 0.2
    }
    canvas.End()
}
```



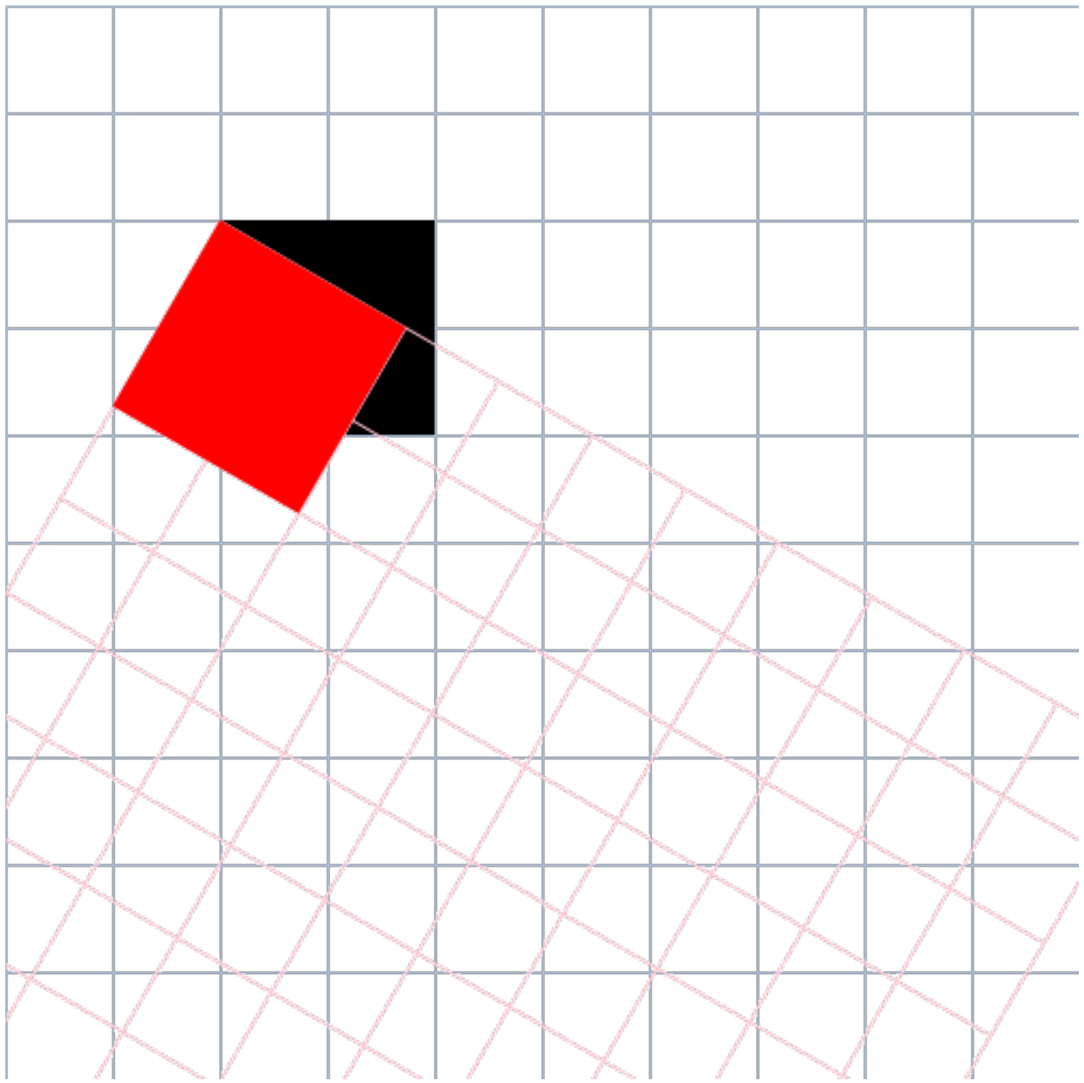
```
package main

import (
    "fmt"
    "os"

    "github.com/ajstarks/svgo"
)

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

func main() {
    fonts := []string{
        "Helvetica", "Times", "Courier",
        "sans-serif", "serif", "monospace",
    }
    sizes := []int{10, 12, 16, 21, 24, 36, 48}
    largest := sizes[len(sizes)-1]
    gutter := largest + (largest / 3)
    margin := gutter * 2
    y := 100

    canvas.Start(width, height)
    for _, f := range fonts {
        x := margin
        canvas.Gstyle("font-family:" + f)
        canvas.Text(x-10, y, f, "text-anchor:end")
        for _, s := range sizes {
            canvas.Text(x, y, fmt.Sprintf("%d", s), fmt.Sprintf("font-size:%dpt", s))
            x += s * 2
        }
        canvas.Gend()
        y += gutter
    }
    canvas.End()
}
```



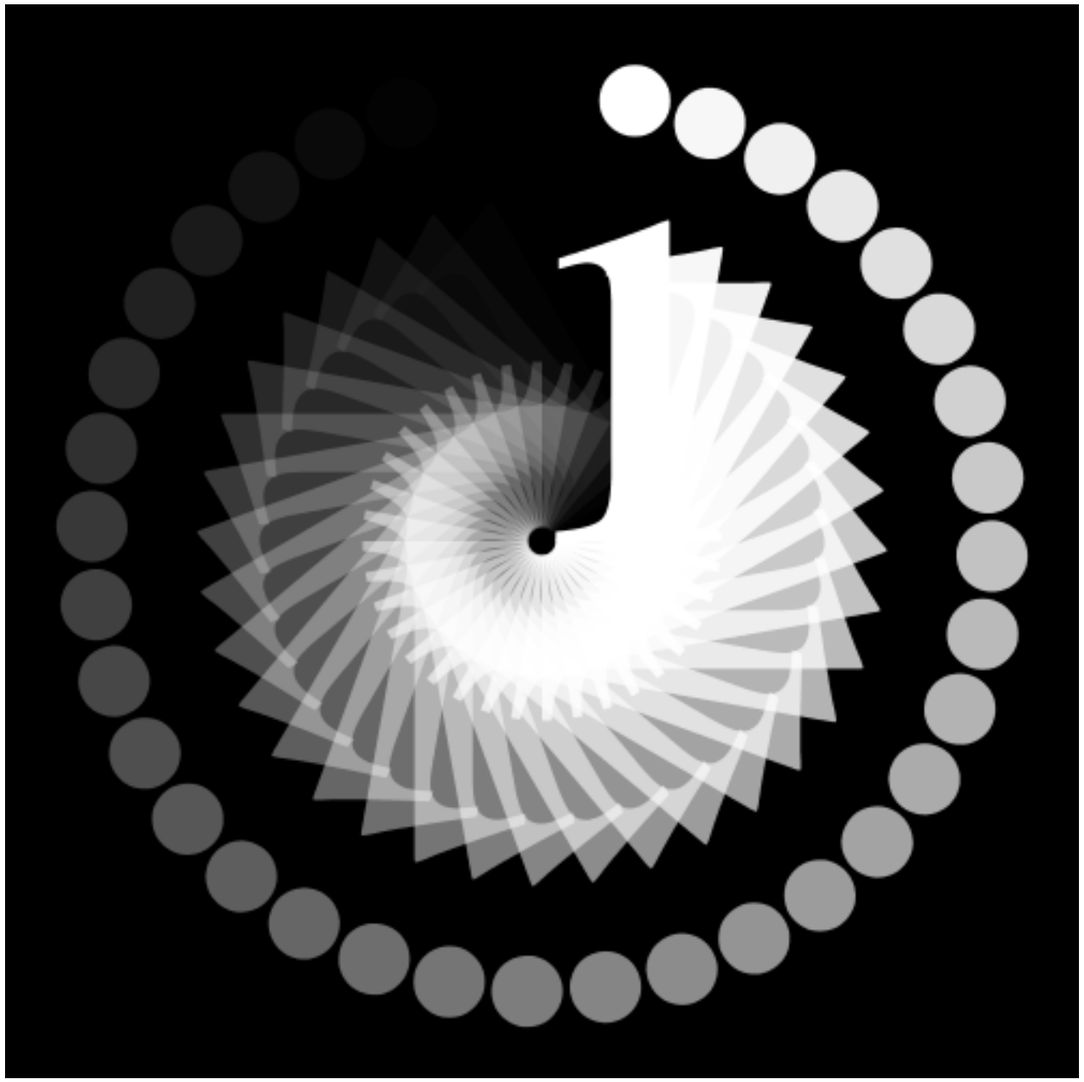
```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

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

func main() {
    lorem := []string{
        "Lorem ipsum dolor sit amet, consectetur adipiscing",
        "elit, sed do eiusmod tempor incididunt ut labore et",
        "dolore magna aliqua. Ut enim ad minim veniam, quis",
        "nostrud exercitation ullamco laboris nisi ut aliquip",
        "ex ea commodo consequat. Duis aute irure dolor in",
        "reprehenderit in voluptate velit esse cillum dolore eu",
        "fugiat nulla pariatur. Excepteur sint occaecat cupidatat",
        "non proident, sunt in culpa qui officia deserunt mollit",
    }
    fontlist := []string{"Georgia", "Helvetica", "Gill Sans"}
    size, leading := 14, 16
    x, y := 50, 20
    tsize := len(lorem)*leading + size*3
    canvas.Start(width, height)
    for _, f := range fontlist {
        canvas.Gstyle("font-family:" + f)
        canvas.Textlines(x, y, lorem, size, leading, "black", "start")
        canvas.Text(x, size+y+tsize/2, f, "fill-opacity:0.3;fill:red;font-size:750%")
        canvas.Gend()
        y += tsize
    }
    canvas.End()
}
```



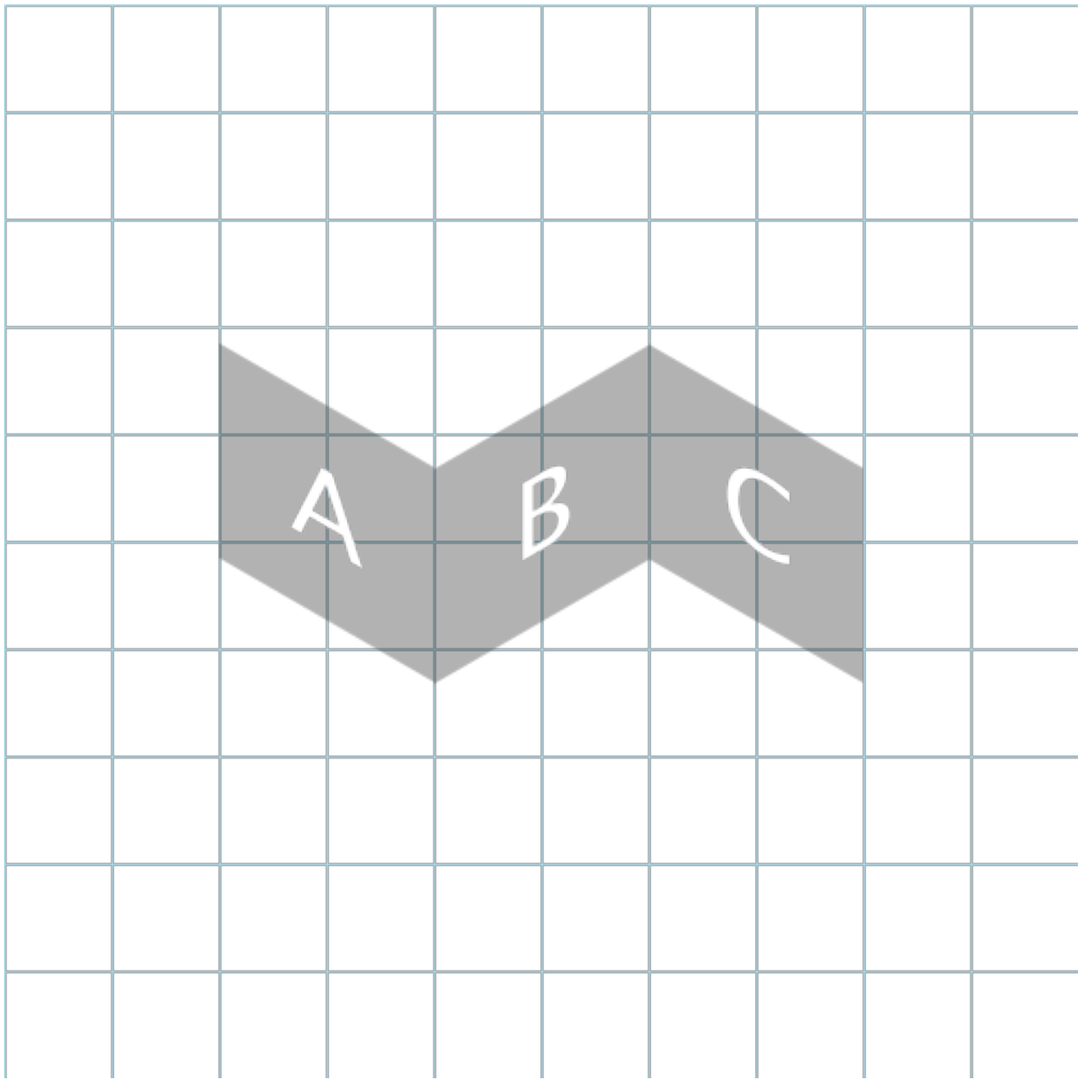
```
package main

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

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

func tro() {
    canvas.Rect(100, 100, 100, 100)
    canvas.TranslateRotate(100, 100, 30)
    canvas.Grid(0, 0, width, height, 50, "stroke:pink")
    canvas.Rect(0, 0, 100, 100, "fill:red")
    canvas.Gend()
}

func main() {
    canvas.Start(width, height)
    canvas.Grid(0, 0, width, height, 50, "stroke:lightsteelblue")
    tro()
    canvas.End()
}
```



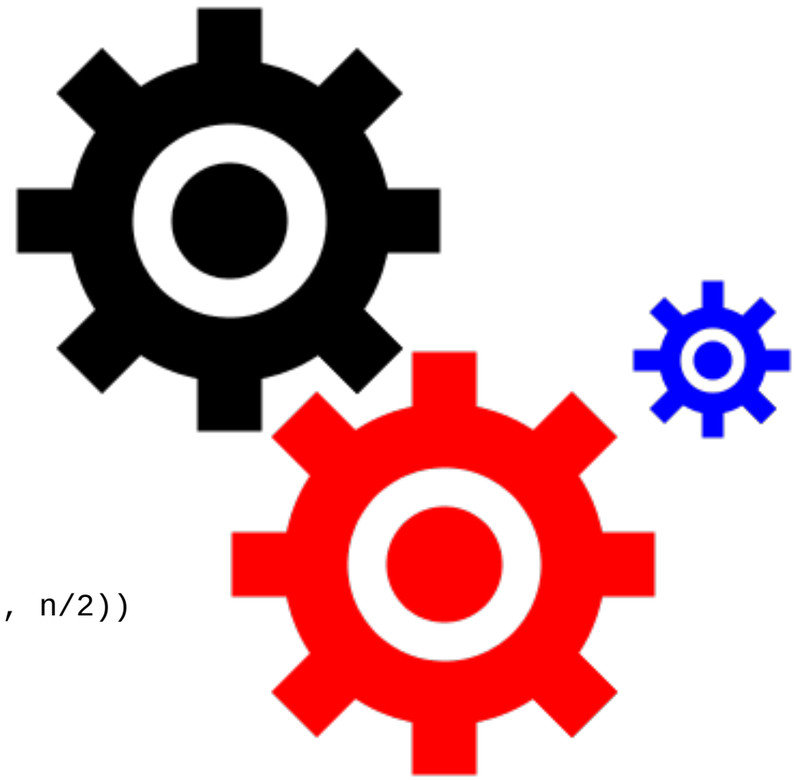
```
package main

import (
    "os"

    "github.com/ajstarks/svgo"
)

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

func main() {
    a := 1.0
    ai := 0.03
    ti := 10.0

    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height)
    canvas.Gstyle("font-family:serif;font-size:244pt")
    for t := 0.0; t <= 360.0; t += ti {
        canvas.TranslateRotate(width/2, height/2, t)
        canvas.Text(0, 0, "i", canvas.RGBA(255, 255, 255, a))
        canvas.Gend()
        a -= ai
    }
    canvas.Gend()
    canvas.End()
}
```



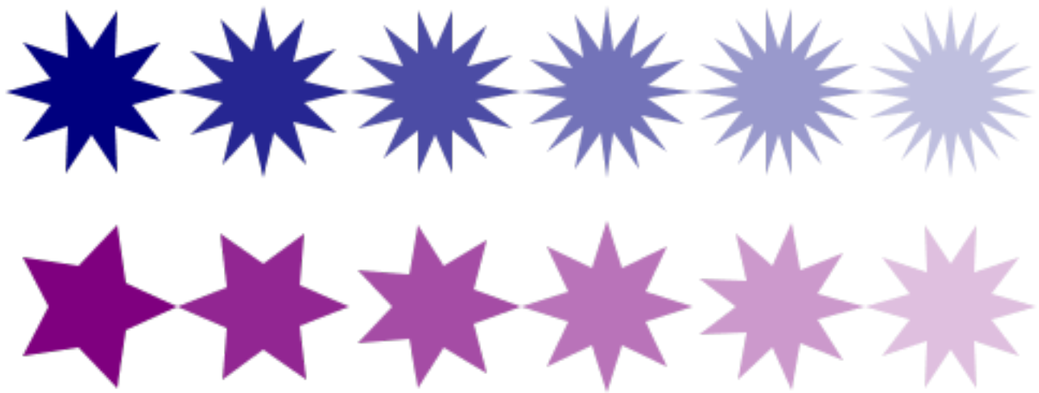
```
package main

import (
    "fmt"
    "os"

    "github.com/ajstarks/svgo"
)

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

func sky(x, y, w, h, a int, s string) {
    tfmt := "font-family:sans-serif;font-size:%dpx;text-anchor:middle"
    canvas.Gstyle(fmt.Sprintf(tfmt, w/2))
    canvas.SkewXY(0, float64(a))
    canvas.Rect(x, y, w, h, "fill:black;fill-opacity:0.3")
    canvas.Text(x+w/2, y+h/2, s, "fill:white;baseline-shift:-33%")
    canvas.Gend()
    canvas.Gend()
}

func main() {
    canvas.Start(width, height)
    canvas.Grid(0, 0, width, height, 50, "stroke:lightblue")
    sky(100, 100, 100, 100, 30, "A")
    sky(200, 332, 100, 100, -30, "B")
    sky(300, -15, 100, 100, 30, "C")
    canvas.End()
}
```



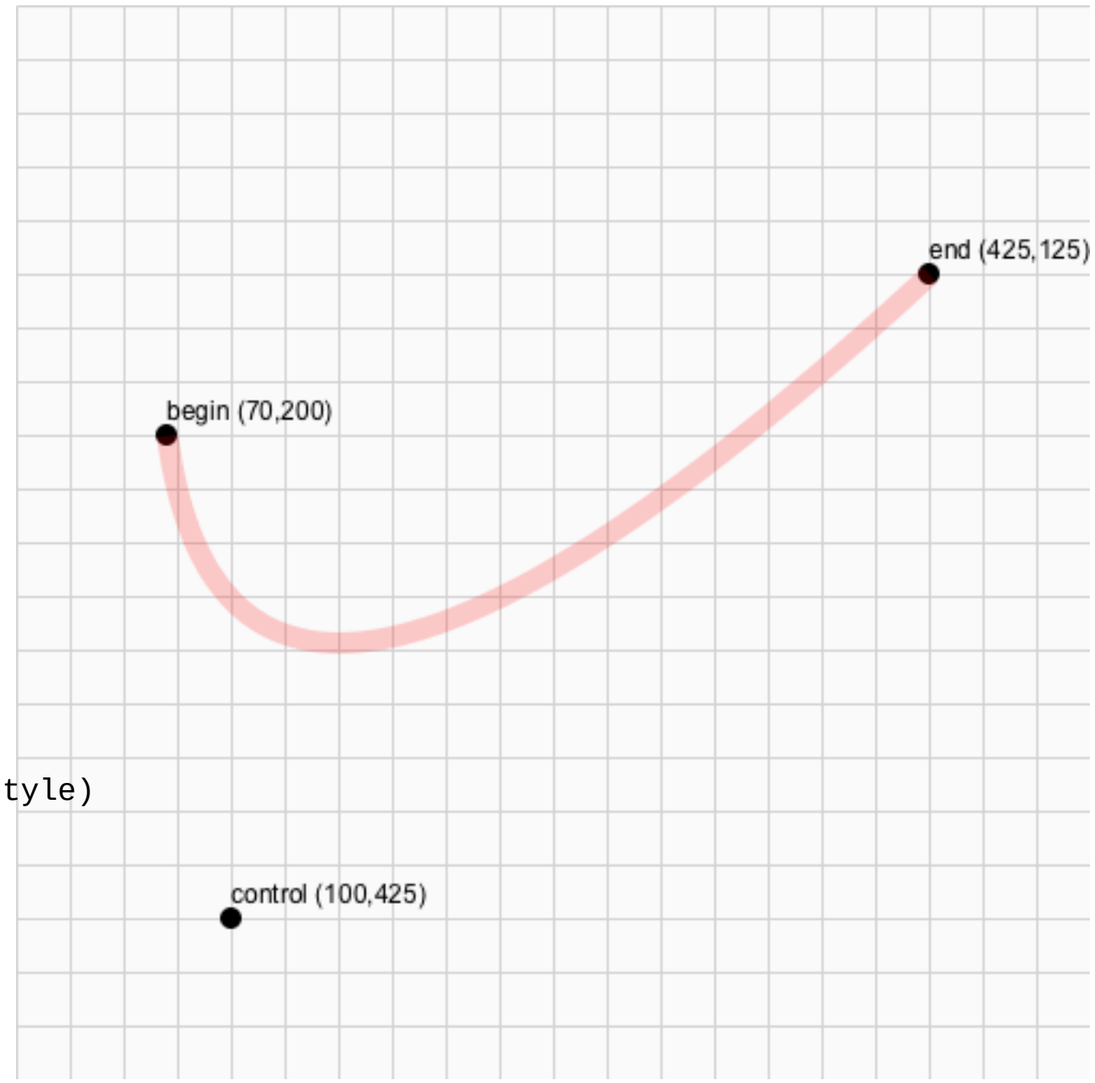
```
package main

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

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

func gear(x, y, w, h, n, l, m int, color string) {
    canvas.Gstyle(fmt.Sprintf("fill:none;stroke:%s;stroke-width:%d", color, n/2))
    canvas.Circle(x+w/2, y+h/2, n)
    canvas.Circle(x+w/2, y+h/2, n/5, "fill:"+color)
    ai := 360 / float64(m)
    for a := 0.0; a <= 360.0; a += ai {
        canvas.TranslateRotate(x+w/2, y+h/2, a)
        canvas.Line(n-l, n-l, n+l, n+l)
        canvas.Gend()
    }
    canvas.Gend()
}

func main() {
    canvas.Start(width, height)
    gear(0, 0, 250, 250, 60, 10, 8, "black")
    gear(100, 160, 250, 250, 60, 10, 8, "red")
    gear(300, 140, 100, 100, 20, 6, 8, "blue")
    canvas.End()
}
```



```
package main

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

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

// See: http://vormplus.be/blog/article/processing-month-day-4-stars
func star(xp, yp, n int, inner, outer float64, style string) {
    xv, yv := make([]int, n*2), make([]int, n*2)
    angle := math.Pi / float64(n)
    for i := 0; i < n*2; i++ {
        fi := float64(i)
        if i%2 == 0 {
            xv[i] = int(math.Cos(angle*fi) * outer)
            yv[i] = int(math.Sin(angle*fi) * outer)
        } else {
            xv[i] = int(math.Cos(angle*fi) * inner)
            yv[i] = int(math.Sin(angle*fi) * inner)
        }
    }
    canvas.Translate(xp, yp)
    canvas.Polygon(xv, yv, style)
    canvas.Gend()
}

func main() {
    canvas.Start(width, height)
    for x, op, i := 50, 1.0, 5; i <= 10; i++ {
        star(x, 200, i*2, 20, 40, canvas.RGBA(0, 0, 127, op))
        star(x, 300, i, 20, 40, canvas.RGBA(127, 0, 127, op))
        x += 80
        op -= 0.15
    }
    canvas.End()
}
```



```
package main

import (
    "fmt"
    "os"

    "github.com/ajstarks/svgo"
)

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

func coord(x, y, size int, label string) {
    tstyle := "text-anchor:middle:font-size:12pt"
    offset := size + (size / 2)
    canvas.Text(x, y-offset, fmt.Sprintf("%s (%d,%d)", label, x, y), tstyle)
    canvas.Circle(x, y, size)
}

func showcurve(bx, by, cx, cy, ex, ey int) {
    dotsize := 5
    sw := dotsize * 2
    cfmt := "stroke:%s;stroke-width:%d;fill:none;stroke-opacity:%.2f"
    style := fmt.Sprintf(cfmt, "red", sw, 0.2)
    coord(bx, by, dotsize, "begin")
    coord(ex, ey, dotsize, "end")
    coord(cx, cy, dotsize, "control")
    canvas.Qbez(bx, by, cx, cy, ex, ey, style)
}

func main() {
    canvas.Start(width, height)
    canvas.Rect(0, 0, width, height, "fill:rgb(250,250,250)")
    canvas.Grid(0, 0, width, height, 25, "stroke:lightgray")
    showcurve(70, 200, 100, 425, 425, 125)
    canvas.End()
}
```



```
package main

import (
    "fmt"
    "os"

    "github.com/ajstarks/svgo"
)

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

func coord(x, y, size int) {
    offset := size * 2
    canvas.Text(x, y-offset, fmt.Sprintf("(%d,%d)", x, y),
        "font-size:50%;text-anchor:middle")
    canvas.Circle(x, y, size, "fill-opacity:0.3")
}

func makepath(x, y, sx, sy, cx, cy, ex, ey int, id, text string) {
    canvas.Def()
    canvas.Qbez(sx, sy, cx, cy, ex, ey, `id="`+id+`"`)
    canvas.DefEnd()
    canvas.Translate(x, y)
    canvas.Textpath(text, "#"+id)
    coord(sx, sy, 5)
    coord(ex, ey, 5)
    coord(cx, cy, 5)
    canvas.Gend()
}

func main() {
    message := `It's fine & "dandy" to have text on a path`
    canvas.Start(width, height)
    canvas.Gstyle("font-family:serif;font-size:21pt")
    makepath(0, 0, 70, 200, 100, 425, 425, 125, "tpath", message)
    canvas.Gend()
    canvas.End()
}
```