Google's Go language

A talk by Elliott Stoneham at SyncIpswich on 31st July 2014

parallel programming comes of age

and why you should care

outline of the evening

- ‘Go & me’ some context
- The story of Go’s birth
- Go as a business tool
- (refreshment break)
- Go code examples
- Q&A, language wars!

- 50 years ago: watched very first Dr Who episode
- 40 years ago: started programming (punch cards)
- 30 years ago: a systems programmer (Metier)
- 20 years ago: became an IT manager (MBA)
- 10 years ago: non-IT (FS,5* service, compliance)

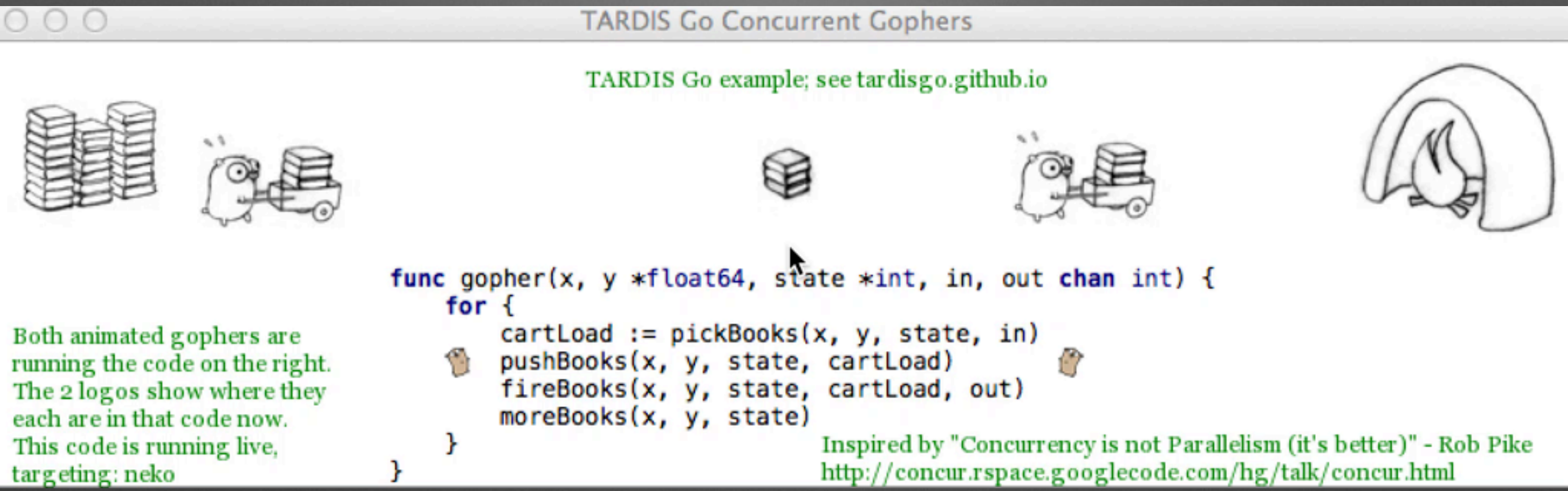
A year ago I discovered Go…

[Gopher logo by Renée French]

Go is like the TARDIS

A small idiosyncratic exterior,
conceals large-scale quality engineering.

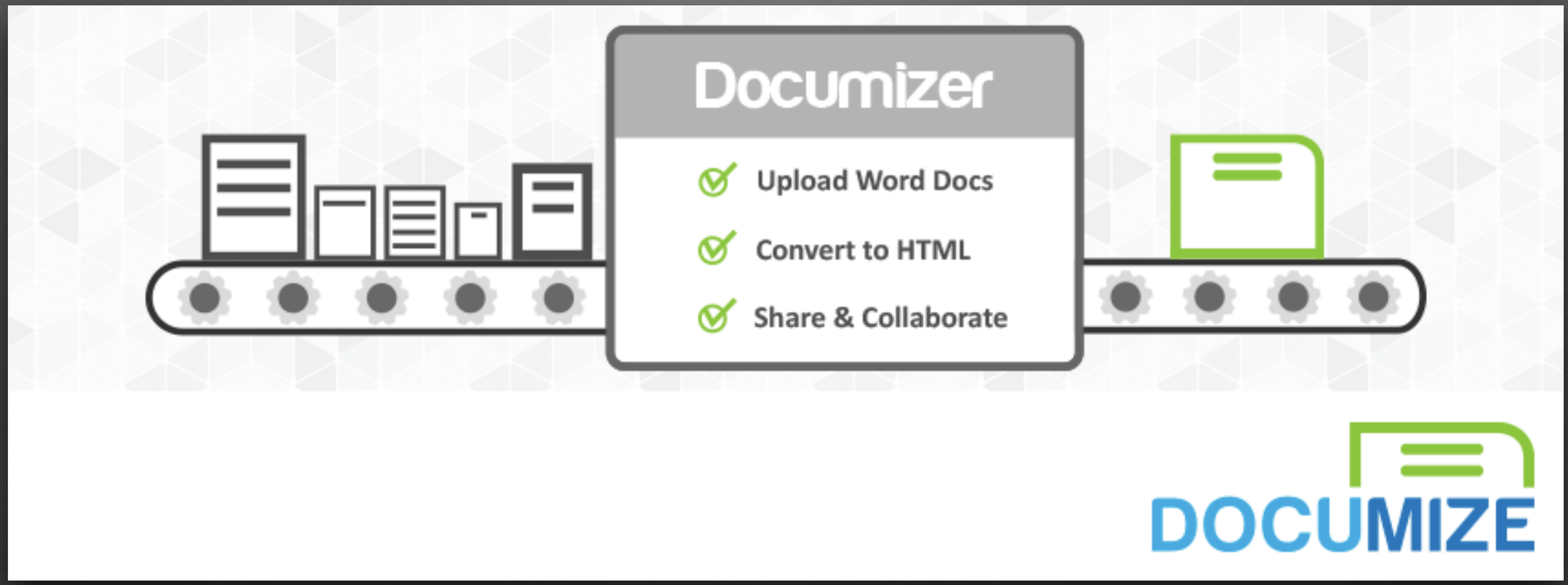
tardisgo.github.io



- Each of the two sprites is a Go “goroutine"
- The pile of books in the middle is a Go “channel”
- This is pipeline concurrency (and parallelism if >1 CPU)

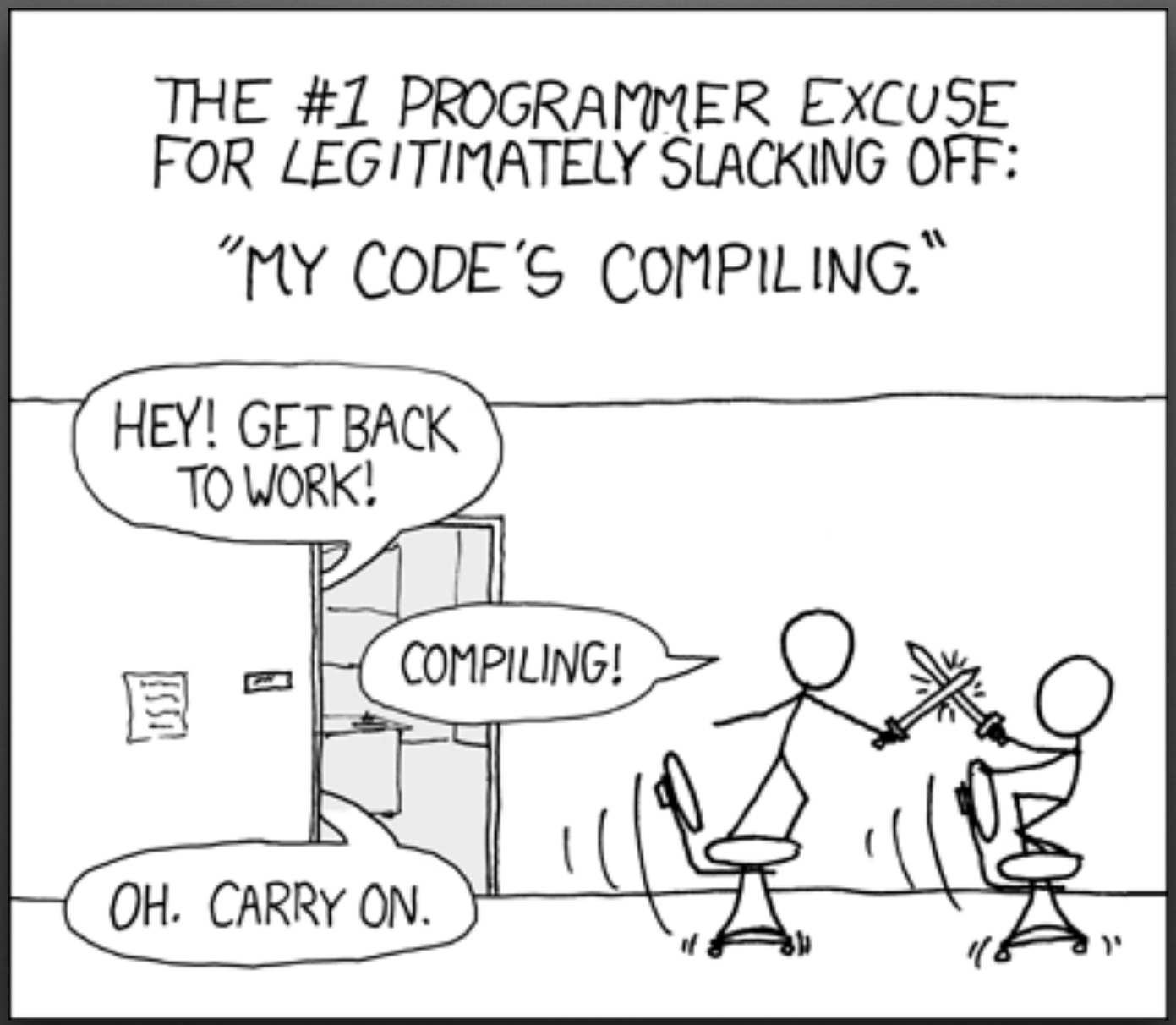
Spoken about TARDIS Go at

- Go London User Group (2x - usually has wait list)
- FOSDEM'14, Brussels (Go room full to bursting)
- GopherCon'14, Denver (800 attended 1st conference)
- FESuffolk
- WWX2014, Paris



www.documize.com

where I am a freelance developer



Waiting for a C++ compilation

[cartoon at http://xkcd.com/303/]

the birth of Go in 2007

“When the three of us got started, it was pure research. The three of us got together and decided that we hated C++.

...we started off with the idea that all three of us had to be talked into every feature in the language, so there was no extraneous garbage put into the language for any reason.”

—Ken Thompson (who designed and implemented the original Unix operating system)

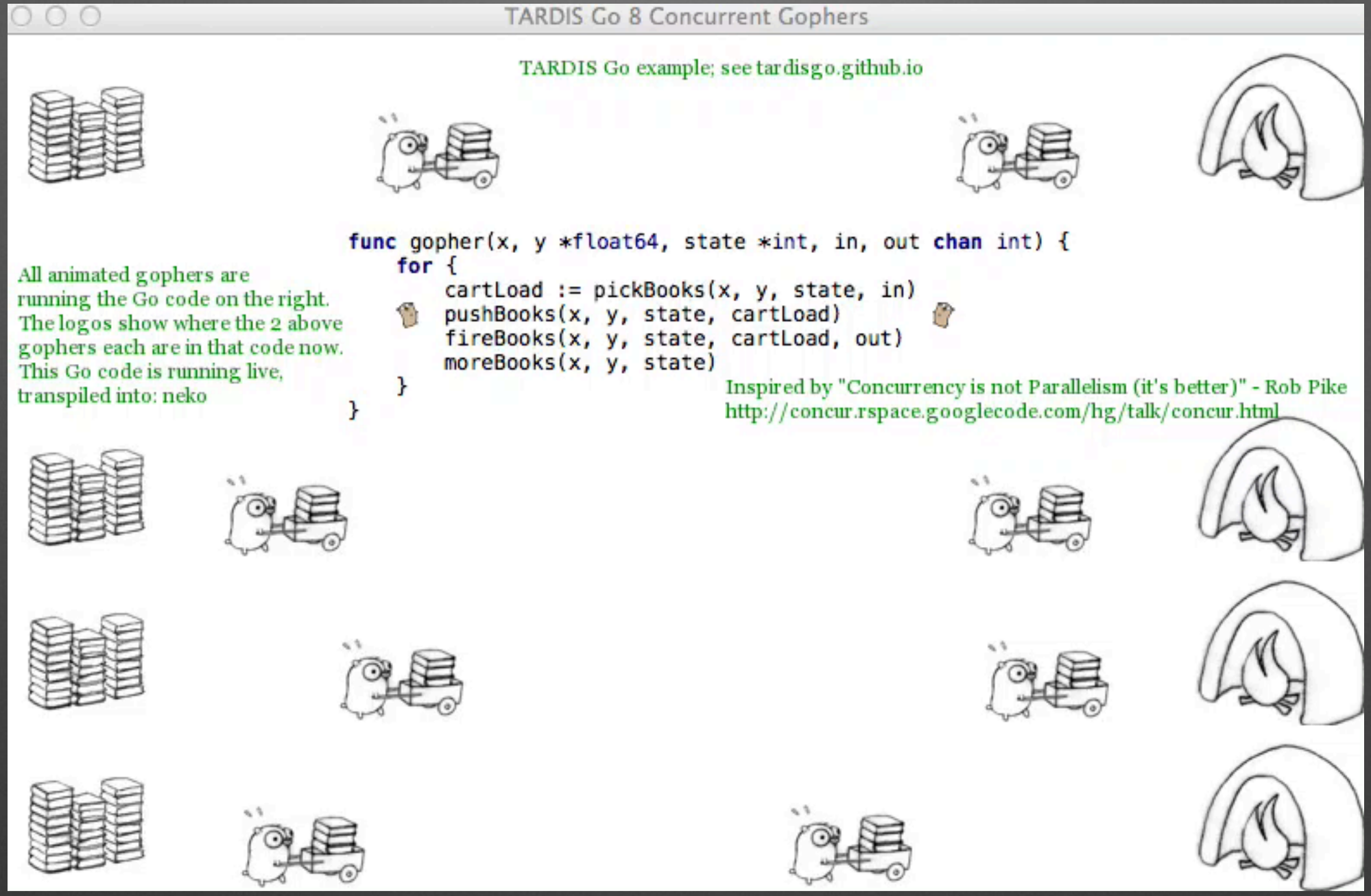
the objectives of Go

- *“Readable, safe, and efficient code.*
- *A build system that scales.*
- *Good concurrency support.*
- *Tools that can operate at Google-scale.*"

— Robert Griesemer (worked on code generation for Google's V8 JavaScript engine)

“Now, Go makes much more sense for the class of problems that C++ was originally intended to solve.”

–Bruce Eckel, author and
founding member of the ANSI/ISO C++ standard committee



burn those C++ manuals!

(TARDIS Go —> Haxe + Haxe / OpenFL; running on the neko target)

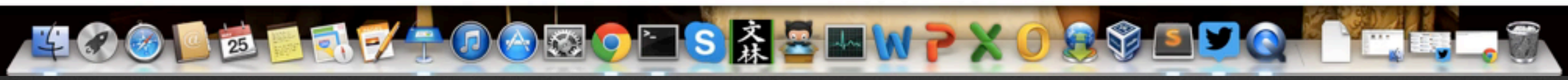
```
}
if *allFlag {
    results := make(chan string)
    for _, cmd := range targets {
        go doTarget(cmd, results)
    }
    for _ = range targets {
        fmt.Println(<-results)
    }
}
```

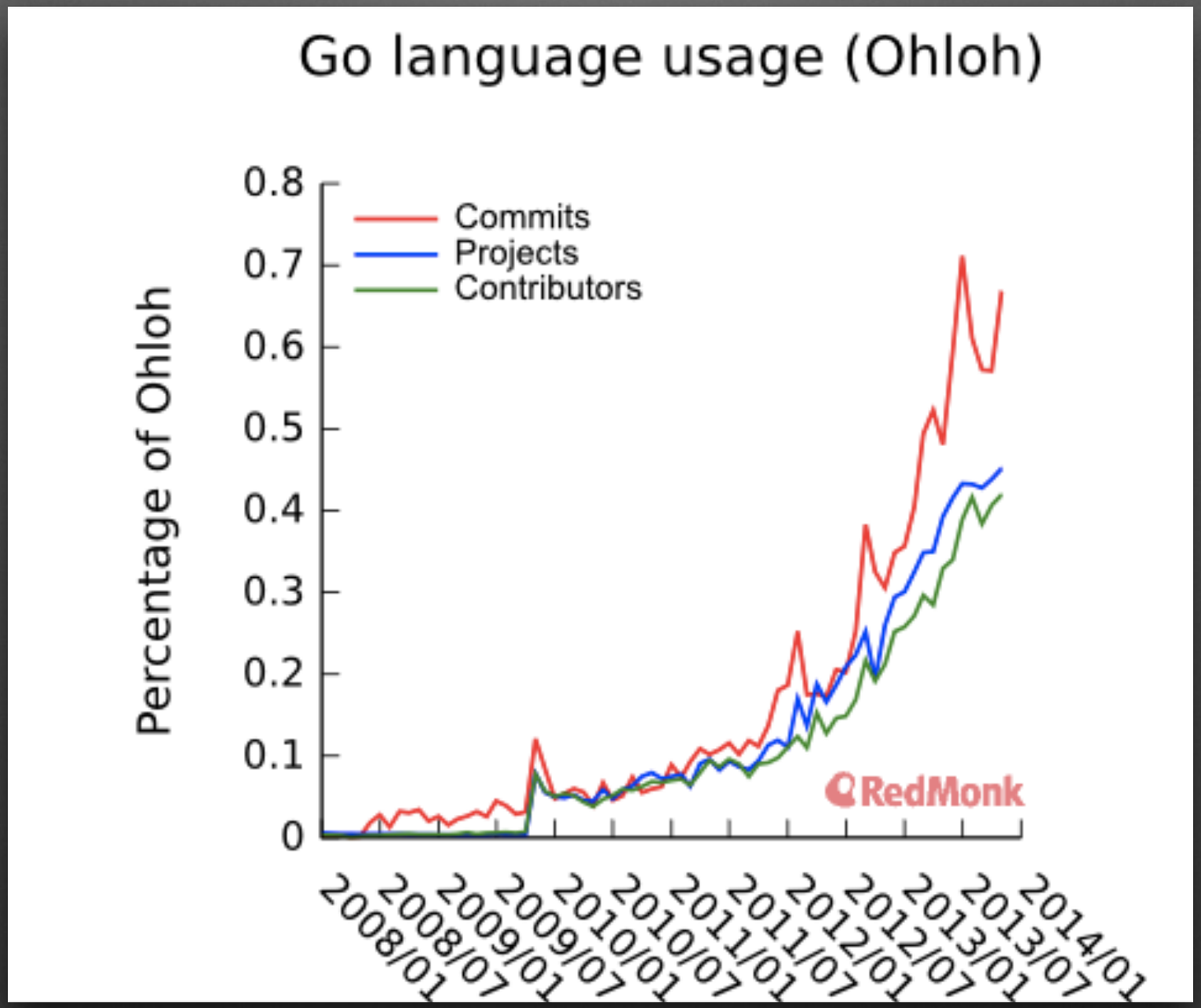
Example: Go to compile and run 8 languages in parallel

QuickTime Player File Edit View Window Help 100% Sun 08:37 Elliott Stoneham

fact10 — bash — 79×20

```
Elliotts-MacBook-Air:fact10 elliott$
```

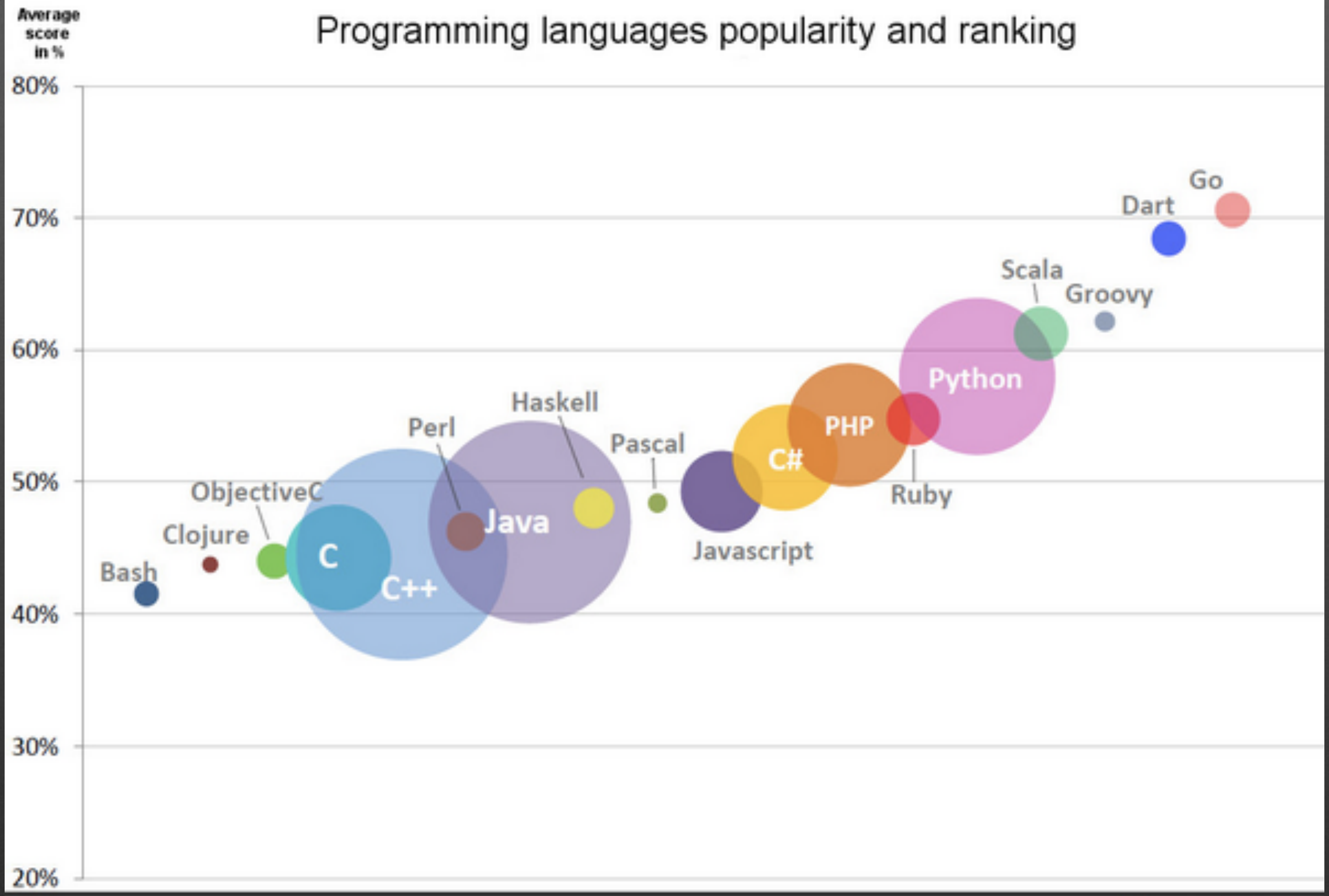




"… Go, a seemingly very minor player, is already used nearly one tenth as much … as the most popular languages in existence. …"

[http://redmonk.com/dberkholz/2014/03/18/go-the-emerging-language-of-cloud-infrastructure/]

Programming languages popularity and ranking



Based on the most used programming languages in the coding contests of the first half of 2014

CodinGame

Matt Butcher's blog: 9 Things I Like About Go

The Toolchain	Clean Code	Go Routines
Channels	Metaphysical Parsimony	Standard Libraries for Today
Core Data Structures	Multiple Returns (and Error Handling)	Interfaces

ThoughtWorks®

Technology Radar

http://www.thoughtworks.com/radar/#/languages-and-frameworks

January to July 2014

January 2014 recommendation: Trial

- The Go language was originally developed by Google as a system programming language to replace C & C++.

- Four years out, Go is gaining traction in other areas.

- The combination of very small, statically linked binaries combined with an excellent HTTP library means Go has been popular with organizations making use of finer-grained, microservice architectures.

July 2014 recommendation: Adopt

- The Go language gradually changed status from "Just Another Language" to a valuable tool in many projects.
- While steadfastly single paradigm in a world of increasingly complex languages, it seems to keep a nice balance between expressiveness, power, and simplicity.

Who really uses it?

Google

docker

apcera

CANONICAL

CoreOS

Iron.io

PACKER

Cloud Foundry

VAGRANT

CLOUDFLARE.

CONSUL

Many, Many More!

source: https://code.google.com/p/go-wiki/wiki/GoUsers

[YouTube: Why We Use Go at Apcera - Derek Collison]

Some other big Go projects:

- github.com/youtube/vitess - YouTube's MySQL serving infrastructure
- soundcloud.com - more than 10 million registered users
- www.gov.uk - front-end for all UK government services
- bitly.github.io/nsq - handling billions of messages per day

Software engineering in the LARGE

Complex distributed systems, running at scale; and written by big teams.

TL;DR

COMPLEX

SIMPLE

[A Recovering Java Developer Learns to Go]

[http://www.slideshare.net/mstine/java-devlearnstogooscon]

Complexity is time, and time is money

Using Go you do not defer complexity cost, you take the hit up-front
… but it turns out that is usually more efficient overall

Product Life Cycle

Sales

Product is launched.
Sales grow slowly as people are not aware of the product.
Informative advertising is used
Usually no profit

Introduction

Sales start to grow rapidly.
Persuasive advertising may be used.
Prices may be reduced as new competitors enter the market.
Profits start coming.

Growth

Maturity

Sales now increase slowly.
Intense competition in the market.
Competitive or promotional pricing may be used.
Advertising expenditure at its highest to sustain growth.
Profits may soon start to fall as the product enters the saturation stage.

Decline

Sales will fall.
Product loses its appeal.
Stiff competition in the market.
Advertising is reduced and then stopped.
Production may be stopped in the future.

Time

Introduction: Developer cost

- Fast compilation, saving developer time
- (attracting NodeJS developers, rubyists & pythonistas)
- Speed of reading code (Go is small & DEWISOTT)
- Speed of refactoring (Go is very pragmatic)
- Speed of training new recruits

Growth: scaling costs

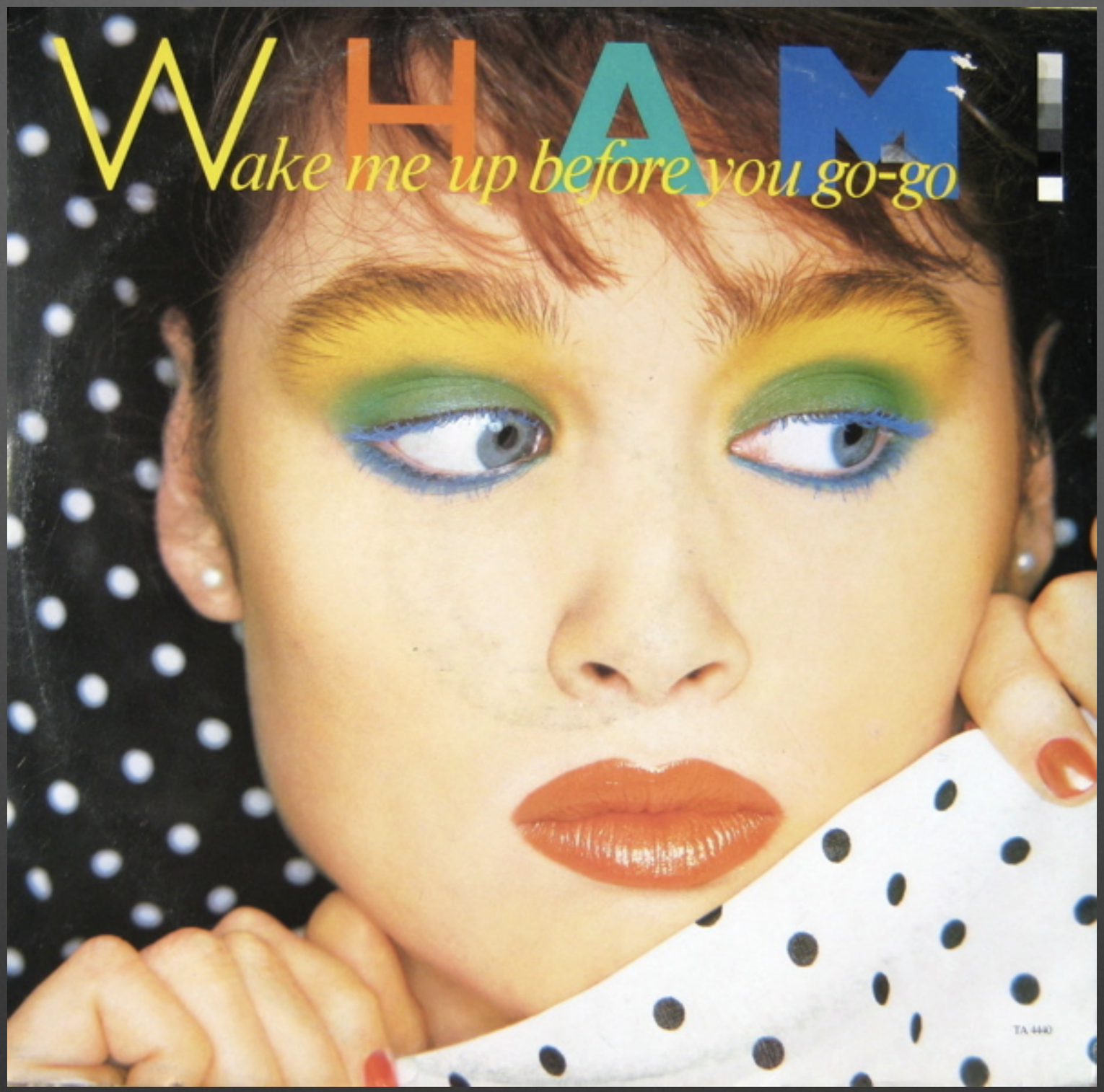
- Static binaries => very simple individual deployment
- Fast spin-up of additional servers, compared with VMs
- Cross-compilation => multi-platform deployment
- Robust networking library and built-in web server
- Easy to use Go as a dev-ops “scripting language”
- Many scaling services are written in Go

Maturity: cloud costs

- Go serves webpages roughly 100x faster than PHP
  [http://www.cyb.co.uk/blog/development/2014/07/why-you-should-consider-golang-for-your-next-web-application]
- Easy parallelism enables elapsed-time speed-ups on a single multi-core machine
- Low memory footprint compared with VMs
- Efficient distributed processing enables “big data” applications across networks of machines

Decline: reflections

- Go is cheap to maintain and simple to re-use
- The “cool” nature of Go makes it very easy to attract top-quality talent
- Go subtly changes the “art of the possible”, making more ambitious systems possible to build, operate and maintain
- Judging by the level of take-up of Go both within Google and externally, the language is here to stay



“Wake me up before you go-Go…”

– Wham!