

ORCHESTRATION

---

**GO: ENABLING DEVOPS TO GO FASTER**

## LICENSE AND MATERIALS

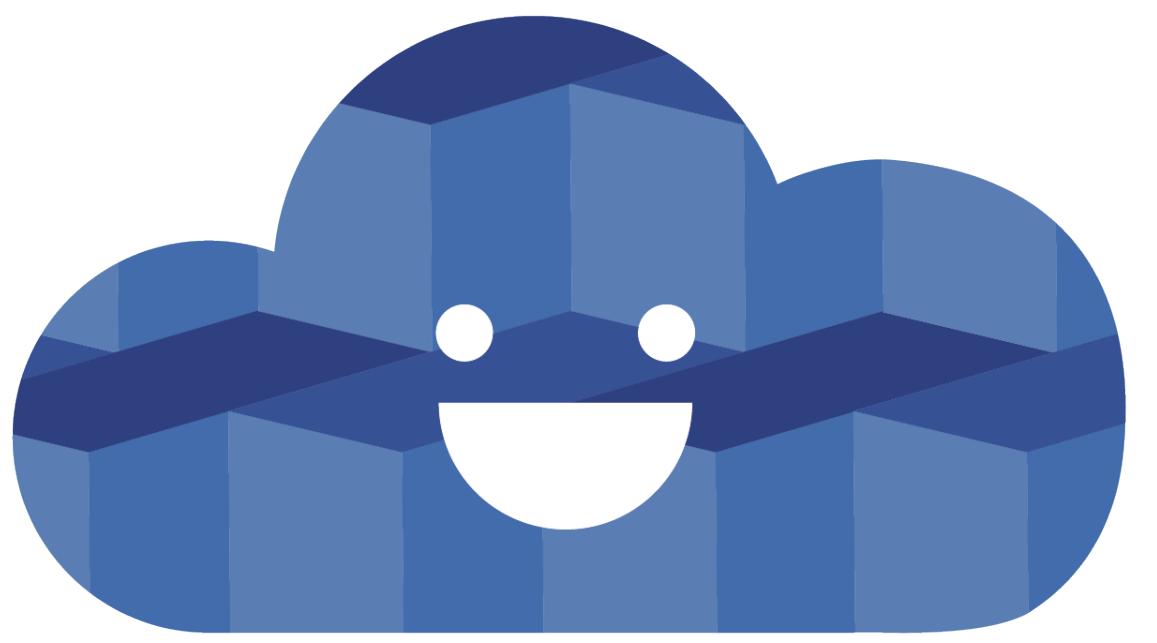
- ▶ Gopher Artwork from Ashley McNamara: <https://github.com/ashleymcnamara/gophers>
- ▶ All product names, logos, and brands are property of their respective owners. All company, product and service names used in this work are for identification purposes only. Use of these names, logos, and brands does not imply endorsement.
- ▶ This presentation is licensed under the [Creative Commons Attribution-ShareAlike 4.0 International](#) license.
- ▶ You are encouraged to remix, transform, or build upon the material, providing you distribute your contributions under the same license.
- ▶ This presentation will be available on [chrissshort.net](#) on or after 31 Jan 2018.

# INTRODUCTION



# chrissshort.net

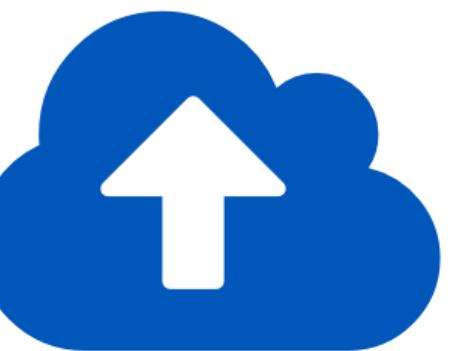
open  
source  
.com





# CLOUD NATIVE COMPUTING FOUNDATION

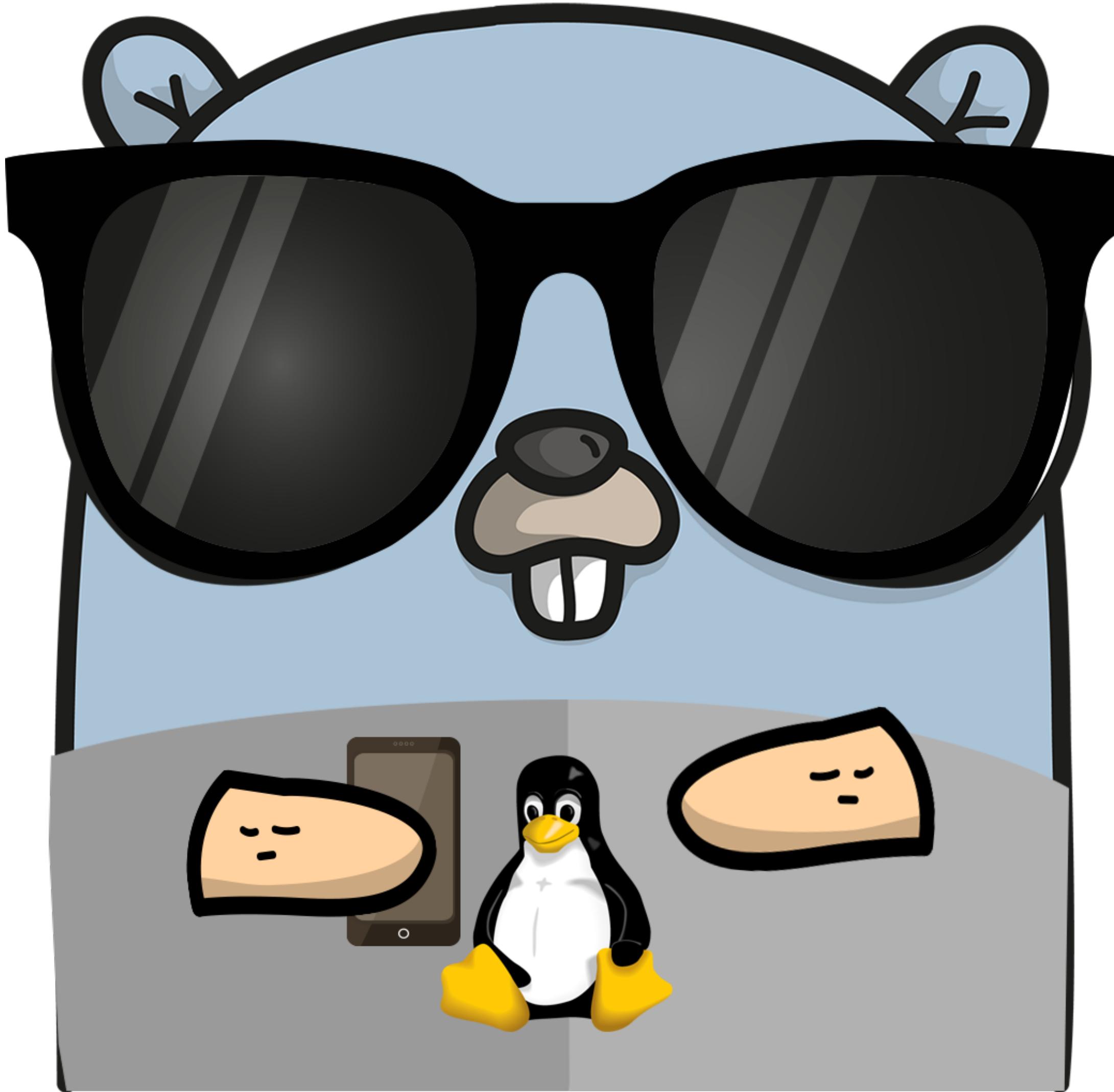
---



# DevOps'ish



I'M ALSO A GOPHER



Chris Short in Gopher Form by [Gopherize.me](http://Gopherize.me)

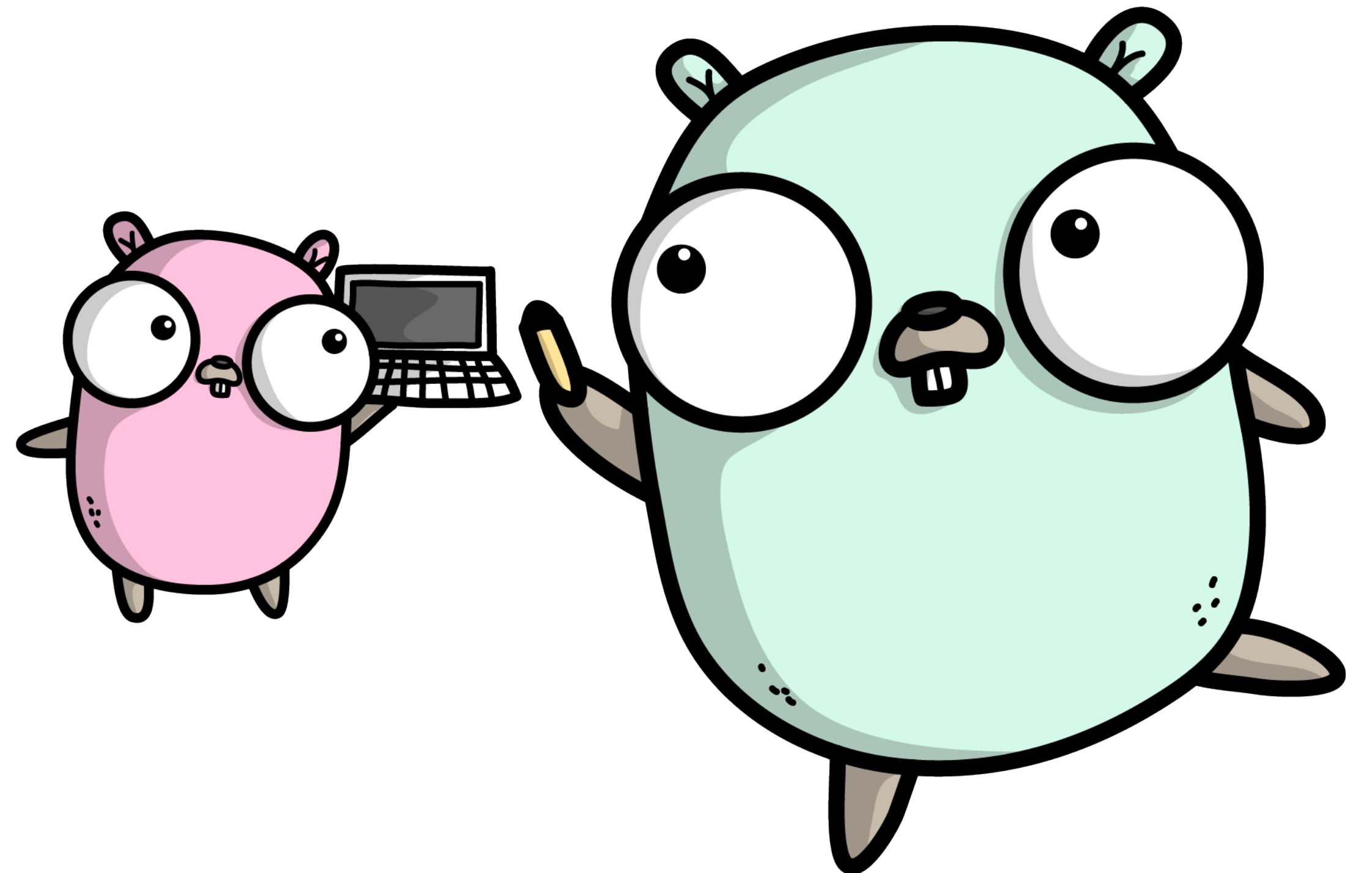
GO: ENABLING DEVOPS TO GO FASTER

---

**WHAT IS GO?**

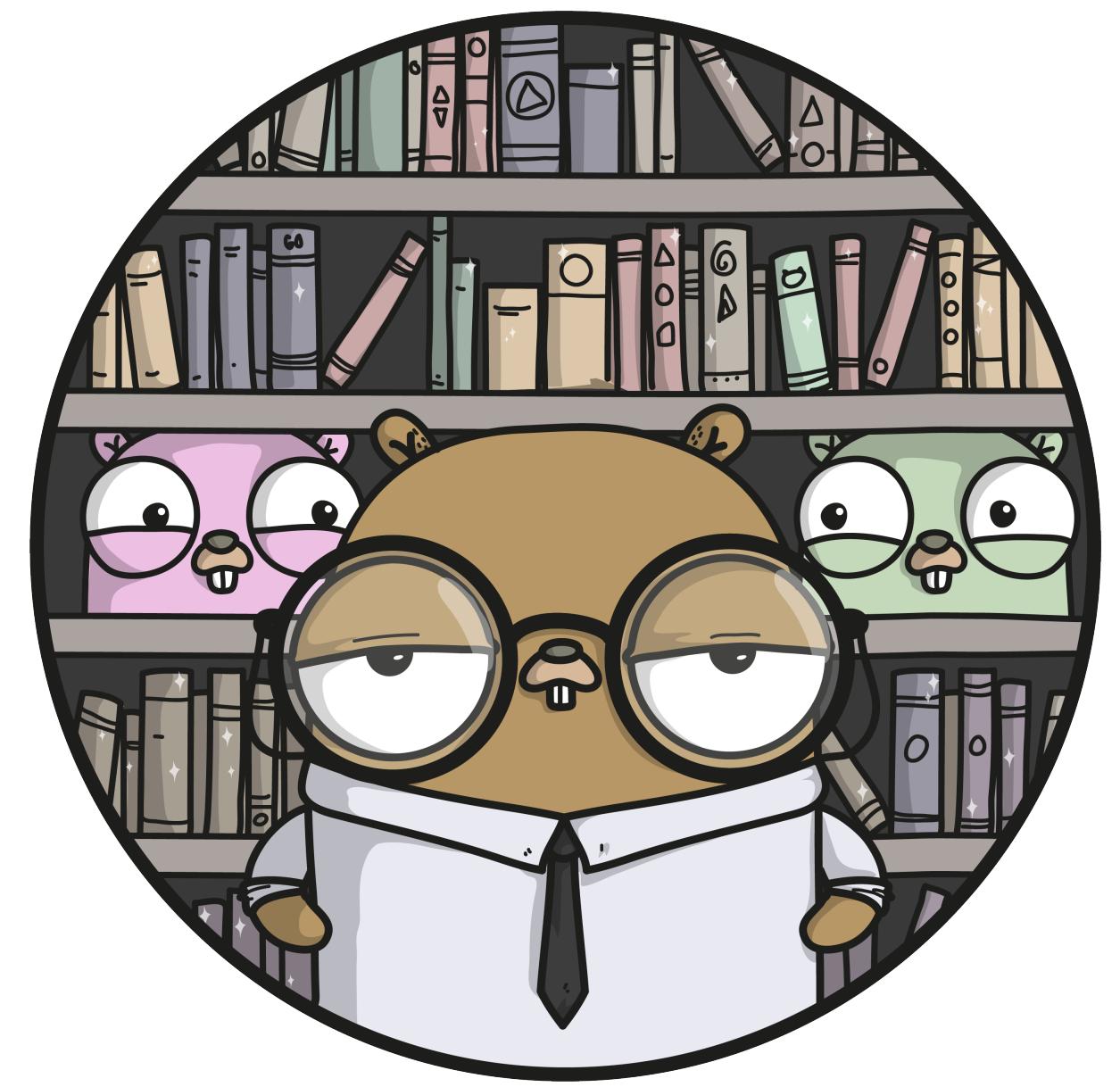
## GOVERVIEW

- ▶ "Go is an open source programming language that makes it easy to build simple, reliable, and efficient software."
- ▶ Development started in 2007
- ▶ Public release in 2009
- ▶ Go 1.0 released in 2012
- ▶ A lot of thought went into Go



## WHO MADE GO?

- ▶ Programming language created at **Google**
- ▶ Created by Robert Griesemer, Rob Pike, Ken Thompson
- ▶ Later adding Ian Lance Taylor and Russ Cox
- ▶ These cats have done some things:
  - ▶ Sawzall (Hadoop), first window system for Unix in 1981, Google's V8 Engine, Plan 9 from Bell Labs, UTF-8, B programming language (C predecessor), regular expressions, GCC, the gold linker, and more



## WHY MAKE GO?

- ▶ "No new major systems language in a decade." –[Rob Pike](#)
- ▶ Designed with the following advances in technology in mind:
  - ▶ Modern Networking
  - ▶ Multi-core CPUs
  - ▶ Slowing of Moore's Law
  - ▶ Improved safety, high speed compilation, and communications



## WHAT IS GO?

### GO VS. OTHER LANGUAGES

Go

Clean/minimalist

No header files

Efficient Garbage Collection

Fast compilation

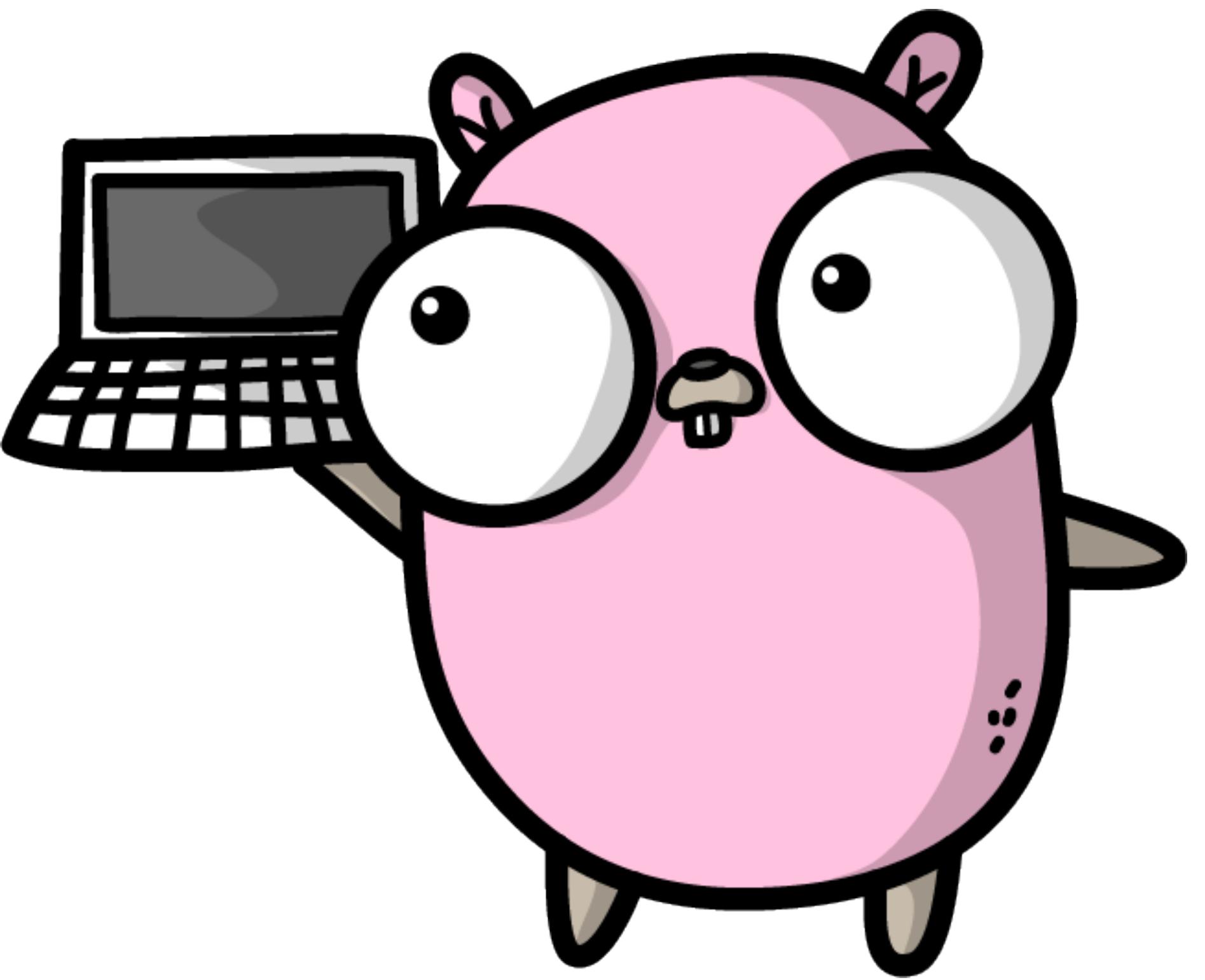
Others

Java? 😂 😂 😂

C/C++ 🙌 🙌 🙌

😱 😱 😱

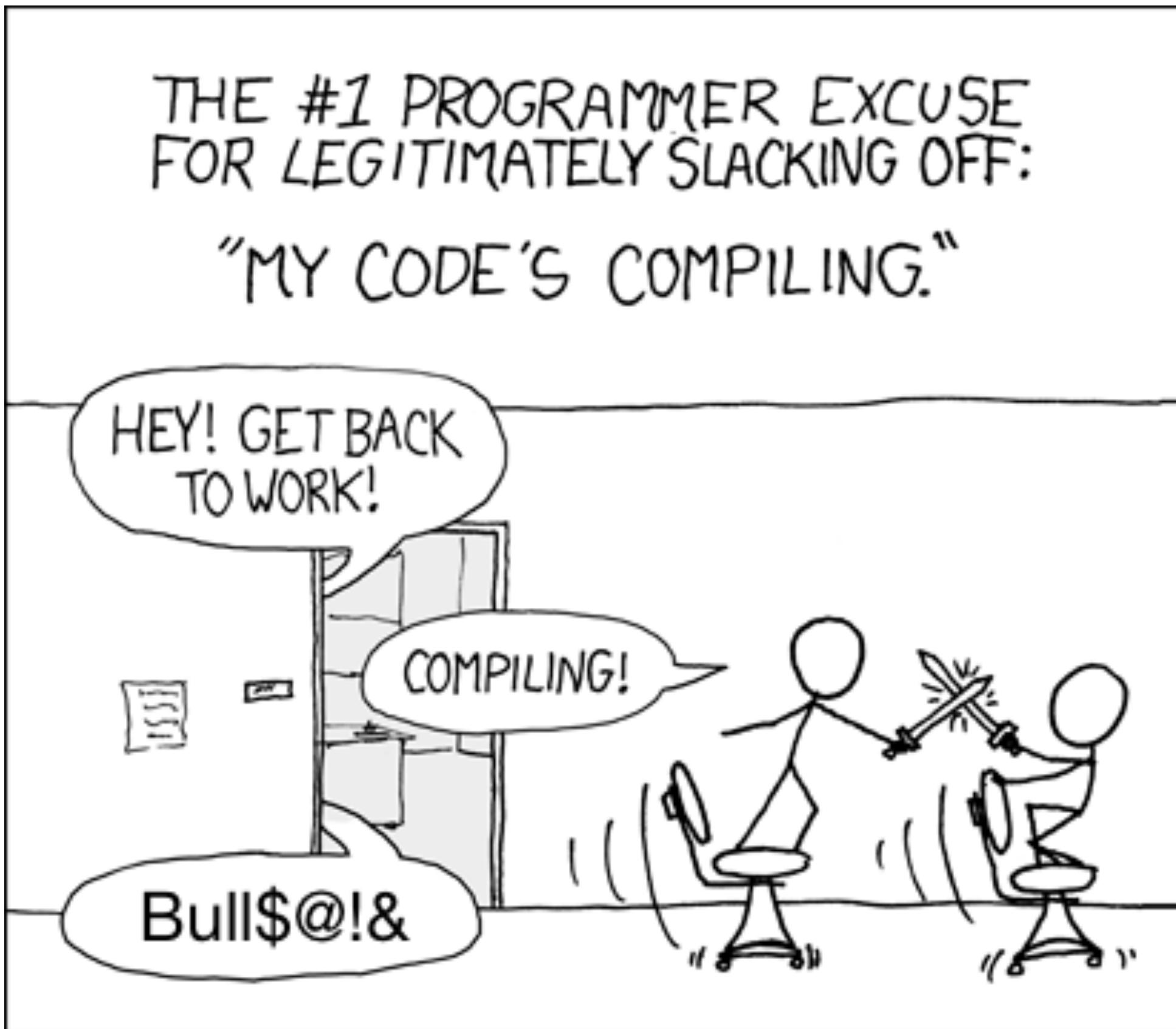
😑 😑 😑



WHAT IS GO?

---

## SORRY, DEVELOPERS



<https://xkcd.com/303/>

## GO TOOLING

- ▶ Standard Library is AMAZING!
- ▶ Intuitive packages:
  - ▶ fmt
  - ▶ crypto
  - ▶ log
  - ▶ net and net/http
  - ▶ os
  - ▶ syscall





"IT JUST WORKS."

Dave Chaney

## WHAT IS GO?

# WHO CONTROLS GO?

- ▶ It's open source! The community!
- ▶ Go was developed at Google by Google Folks
- ▶ But, look who is writing Go code
  - ▶ #2: Microsoft
  - ▶ #4: Apache
  - ▶ #6 Alibaba

	Repositories	Developers	Trending: this month ▾
1	 google (Google) grumpy Grumpy is a Pyth...		
2	 Microsoft (Microsoft) docker Docker - the ope...		
3	 ethereum go-ethereum Official Go imple...		
4	 apache (The Apache Software Foundation) incubator-service... A standalone ser...		
5	 tensorflow k8s Tools for ML/Ten...		
6	 alibaba (Alibaba) pouch Pouch is an open...		
7	 kubernetes (Kubernetes) kubernetes Production-Grad...		
8	 shadowsocks (shadowsocks) shadowsocks-go go port of shado...		
9	 golang (Go) go The Go program...		
10	 aws (Amazon Web Services) aws-sdk-go AWS SDK for the...		

GO: ENABLING DEVOPS TO GO FASTER

---

**WHAT IS GO GOOD AT?**



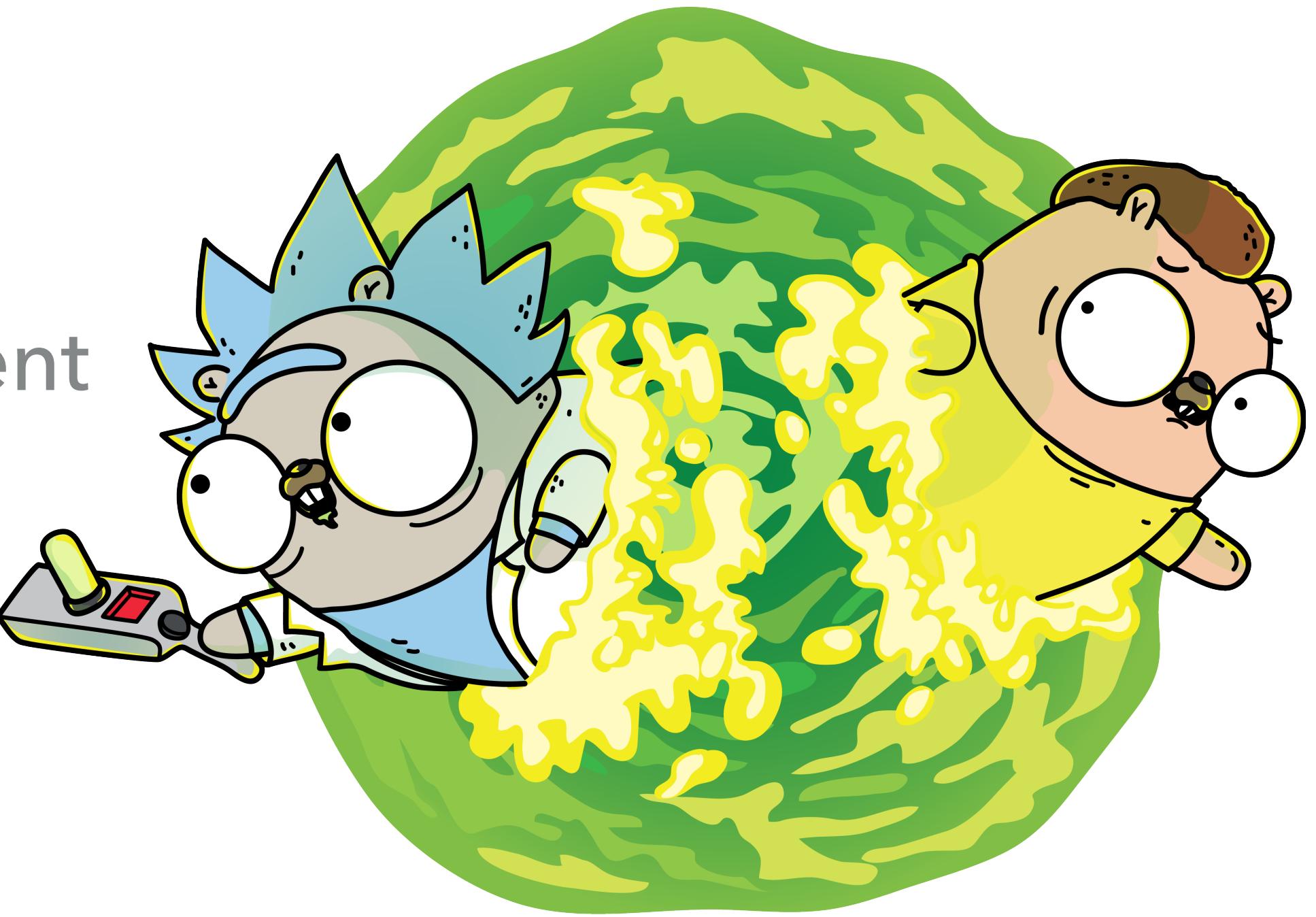
## CONTAINER RUNTIMES

- ▶ Go is a lower-level language (like C and C++)
- ▶ Interacts with kernel directly; not through a VM (like Java)
- ▶ Go easily manages processes, syscalls, etc.
- ▶ Go's concurrency model makes for efficient core/thread use
- ▶ Multi-architecture builds
- ▶ Static compilation



## CRYPTOCURRENCIES

- ▶ Ethereum has the #3 GitHub project for Go
- ▶ geth is the Go implementation of Ethereum client
- ▶ geth is the *default* Ethereum client
- ▶ geth became the "reference client"



"THE TRUE POWER OF ... GO WAS THE  
EASE OF USE AND THE POWER OF  
COMMUNICATING CONCEPTS..."

Jeffrey Wilcke

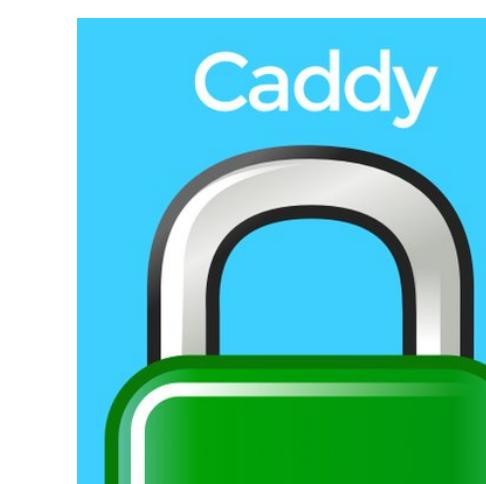
## STORAGE SYSTEMS

- ▶ **Dropbox's Magic Pocket** is a multi-exabyte storage system written in (mostly) Go
- ▶ Rewrite of prototype was necessary
- ▶ Go addresses the need for massively distributed systems
- ▶ 100K LOC written by 4 people in only

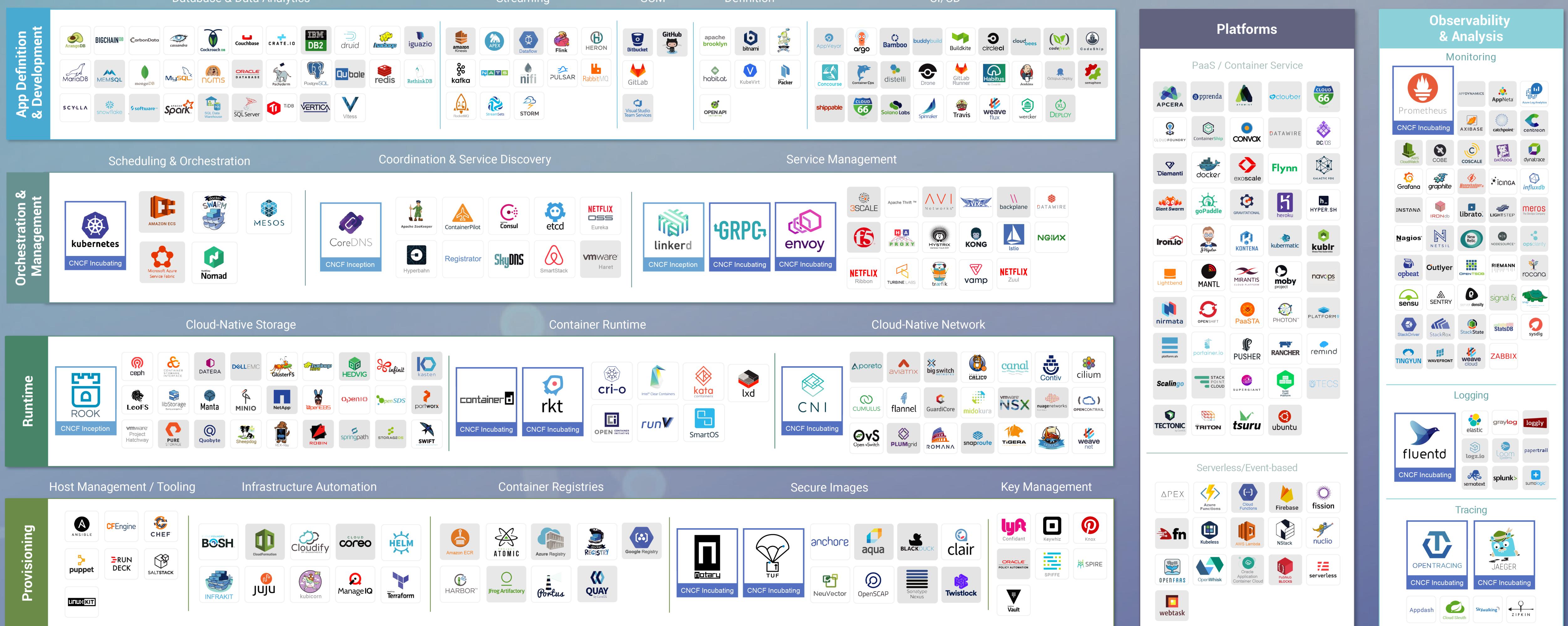


WHAT IS GO GOOD AT?

## PROJECTS UTILIZING GO



## Cloud Native Landscape



*This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.*



[github.com/cncf/landscape](https://github.com/cncf/landscape)



Greyed logos  
are not  
open source

## GOPINIONS

- ▶ When asked, "Why does Go make you happy?" Go devs responded with:
  - ▶ "Less is more." –Kris Nova, Heptio
  - ▶ "Go does a really awesome job at making the easy things really easy, and the complicated things easy to understand while not abstracting them away."  
–Julia Ferraioli, Google
  - ▶ "Go makes me happy because it's so cool it has its own set of proverbs! [go-proverbs.github.io](https://go-proverbs.github.io)" –Carlisia Pinto, Fastly
  - ▶ "Comprehensible parallelism that won't shoot you in the foot is Go's most winsome feature." –Liz Fong-Jones, Google Cloud

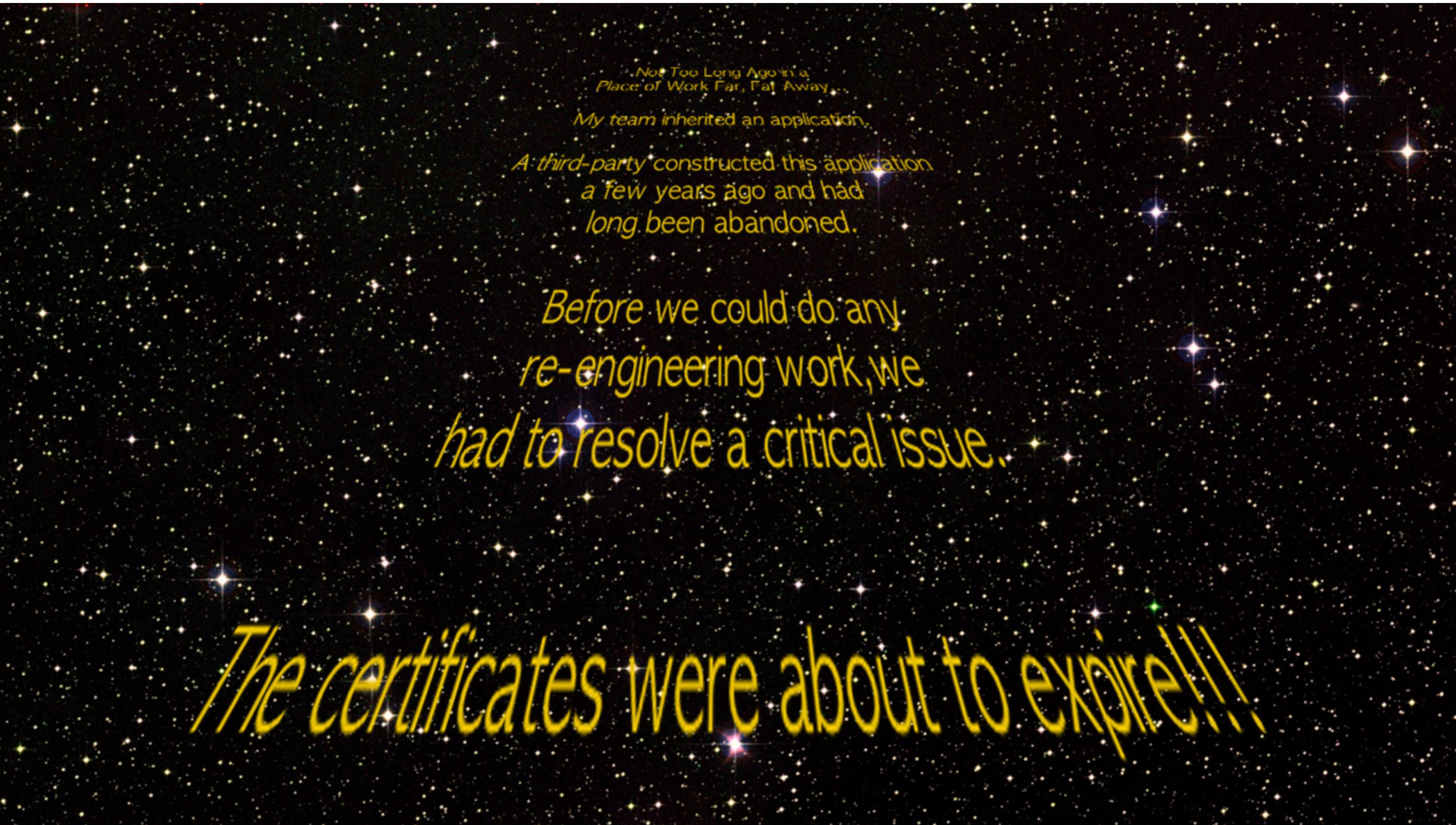
GO: ENABLING DEVOPS TO GO FASTER

---

# HOW GO BAILED ME OUT

## NOT TOO LONG AGO IN A PLACE OF WORK FAR, FAR AWAY...

---



*Not Too Long Ago in a  
Place of Work Far, Far Away...*

*My team inherited an application.*

*A third-party constructed this application  
a few years ago and had  
long been abandoned.*

*Before we could do any  
re-engineering work, we  
had to resolve a critical issue.*

*The certificates were about to expire!!!*

AND OF COURSE PRODUCTION

---



## LET'S TALK CERTIFICATE CHAINS

---



# THIS IS THE GOAL

You are here: [Home](#) > [Projects](#) > [SSL Server Test](#) > [chrissshort.net](#)

## SSL Report: chrissshort.net (104.198.106.181)

Assessed on: Mon, 08 Jan 2018 21:03:12 UTC | [Hide](#) | [Clear cache](#)

[Scan Another »](#)

### Summary

**Overall Rating**

**A+**

	Certificate	Protocol Support	Key Exchange	Cipher Strength
Certificate	95	92	88	85

Visit our [documentation page](#) for more information, configuration guides, and books. Known issues are documented [here](#).

This site works only in browsers with SNI support.

HTTP Strict Transport Security (HSTS) with long duration deployed on this server. [MORE INFO »](#)

### Certificate #1: RSA 2048 bits (SHA256withRSA)

**Server Key and Certificate #1**

Subject	chrissshort.net
Fingerprint SHA256:	aa9309af84b0a3b78eeece18102a9bc9963acb6bdd15f60a1fde4f6d857a75b1
Pin SHA256:	4wPHDis4wFxaphEtm+0fMnFqoPMxHnE0Qj31Dc56Lbg=
Common names	chrissshort.net
Alternative names	chrissshort.net www.chrissshort.net



# RapidSSL

Simple site security for less.

Regions: US/Canada Europe UK Australia

buy

switch

resell

learn

support

Home > Support > General Information Details

## RapidSSL Intermediate CAs

[Printable version](#)

General Information ID: INFO1548

Updated: 06/02/2016

### CONTACT SUPPORT

US Support:

Order Processing  
[Email Form](#)

Technical Support  
[Email Form](#)

European Support:

Order Processing



SSL

Solutions

Partner

Company

Support

1.801.701.9600



### Description

RapidSSL uses an Intermediate CA to enhance the security of SSL certificates. When installing a RapidSSL certificate, it is essential to install the Intermediate CA at the same time as the SSL certificate, this ensures that the SSL certificate is fully trusted by all browsers and prevents SSL errors from appearing when users visit the website.

The RapidSSL and Wildcard certificates Intermediate CA can be downloaded from the table below. Once your SSL certificate is installed, please use the [SSL Certificate Installation Checker](#) to ensure a problem-free SSL certificate experience for you and your users!

### RSA SHA-1 SSL Certificates

Product Name	Intermediate CA
RapidSSL Wildcard FreeSSL	<a href="#">SO26462</a>

### RSA SHA-2 (under SHA-1 Root) SSL Certificates

Product Name	Intermediate CA
RapidSSL Wildcard FreeSSL	<a href="#">SO28616</a>

### RSA SHA-2 (under SHA-2 Root) SSL Certificates

<b>Compatibility Intermediate CA</b>	Issuer: GTE CyberTrust Global Root Valid until: 10/Aug/2018 Serial #: 0E:E0:68:2D:BB:98:2D:92:C6:85:6A:DA:DE:48:19:80 Thumbprint: F08B49D0EBE7975062CD19C731B141DF4D11DF52 <a href="#">Download</a>
<b>Cybertrust Japan Issuing CA-1</b>	Issuer: Verizon Global Root CA Valid until: 01/Sep/2026 Serial #: 0C:5B:12:0D:AC:42:A1:CB:7B:20:89:DB:17:6E:04:78 Thumbprint: 4B8FE3B160D85B627F660C6A425059C2A420A774 <a href="#">Download</a>
<b>DigiCert Assured ID CA-1</b>	Issuer: DigiCert Assured ID Root CA Valid until: 10/Nov/2021 Serial #: 06:FD:F9:03:96:03:AD:EA:00:0A:EB:3F:27:BB:BA:1B Thumbprint: 19A09B5A36F4DD99727DF783C17A51231A56C117 <a href="#">Download</a>
<b>DigiCert Assured ID CA G2</b>	Issuer: DigiCert Assured ID Root G2 Valid until: 01/Aug/2028 Serial #: 0F:5F:CC:FC:AB:20:F3:DF:8E:6D:A3:D8:47:67:C2:93 Thumbprint: 28E96CDB1DBA273FD1A6151BE15F088F26046273 <a href="#">Download</a>
<b>DigiCert Assured ID CA G3</b>	Issuer: DigiCert Assured ID Root G3 Valid until: 01/Aug/2028 Serial #: 01:05:DA:E2:55:AA:B2:95:4A:0D:B2:C9:E6:B5:32:2C Thumbprint: C619BE4F415453F46D020ED79F5D5CA5C37E14AD <a href="#">Download</a>
<b>DigiCert Assured ID Code Signing CA-1</b>	Issuer: DigiCert Assured ID Root CA Valid until: 10/Feb/2026 Serial #: 0F:A8:49:06:15:D7:00:A0:BE:21:76:FD:C5:EC:6D:BD Thumbprint: 409AA4A74A0CDA7C0FEE6BD0BB8823D16B5F1875 <a href="#">Download</a>

ESPN

CHIGA

MICHIGAN

Mich St

21



OUCHDOWN

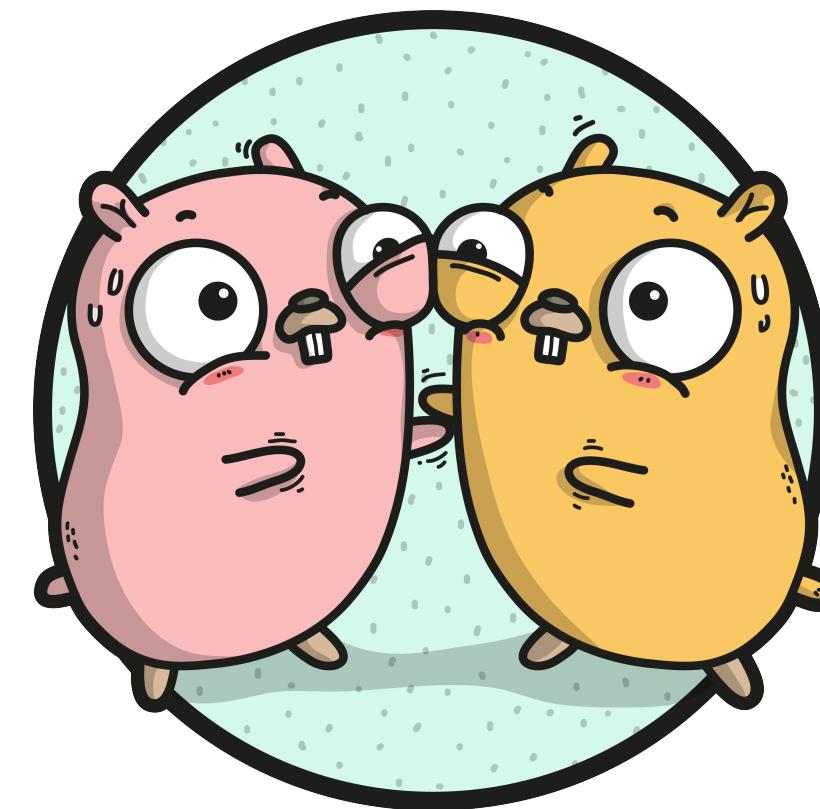
## SO WHAT DOES ANY GOOD ENGINEER DO?

---



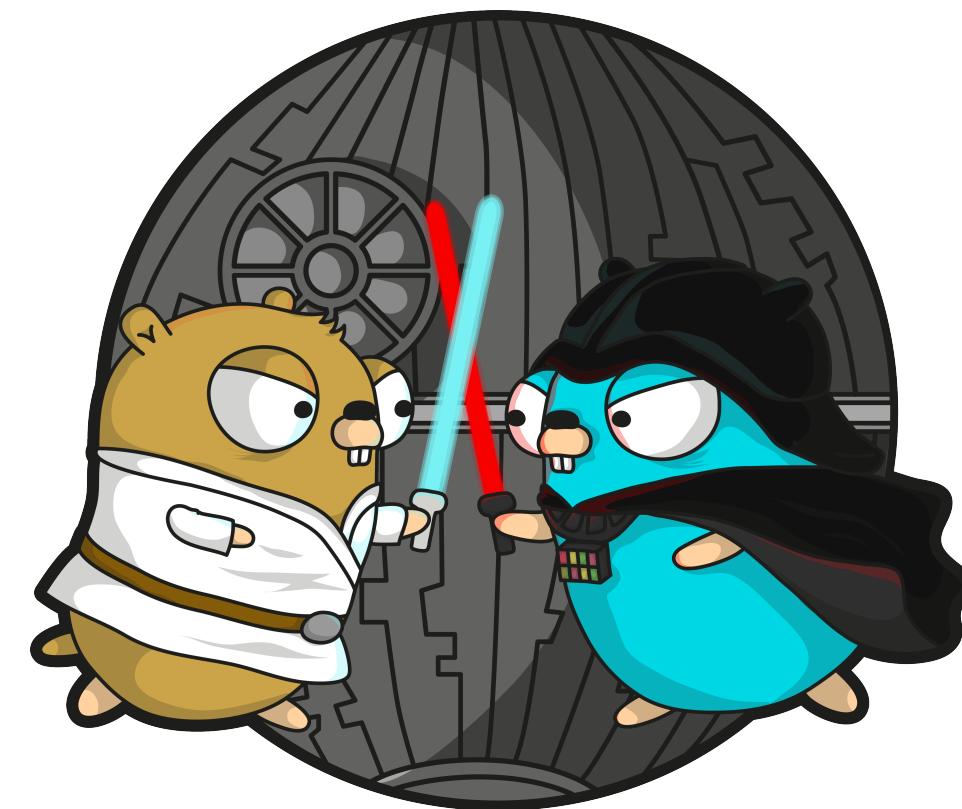
### LOG

- ▶ The Go [log](#) package is pretty self explanatory
- ▶ Package that enables logging
- ▶ Needed a spectacular failure at the sign of trouble
- ▶ log has three helper functions: print, fatal, and panic



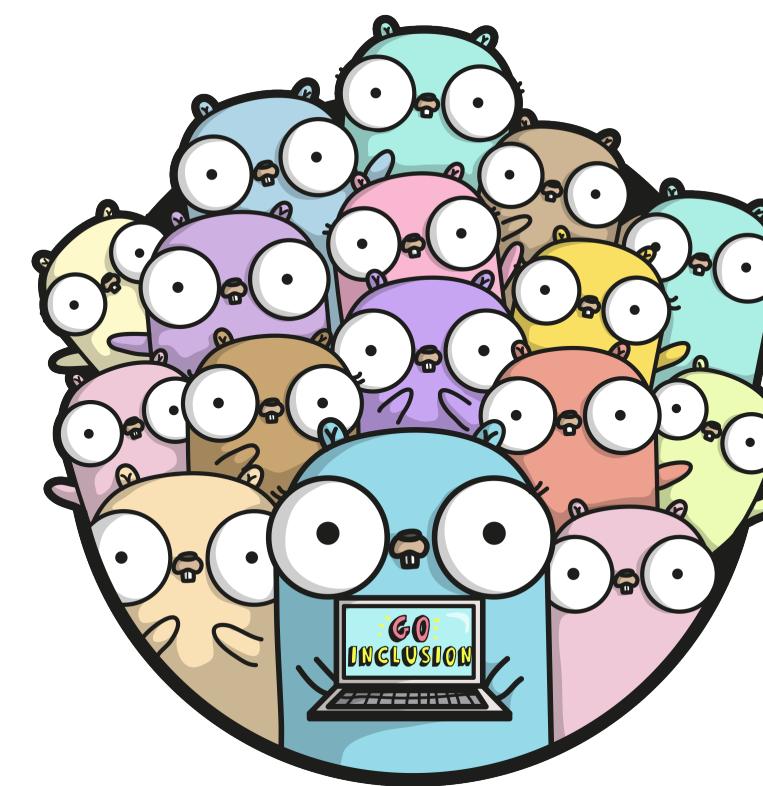
### CRYPTO/TLS

- ▶ The Go [`crypto/tls`](#) package partially implements TLS 1.2, as specified in [RFC-5246](#)
- ▶ Package configures usable SSL/TLS versions
- ▶ Identifies preferred cipher suites and elliptic curves used during handshakes
- ▶ This is the package that handles connections securely



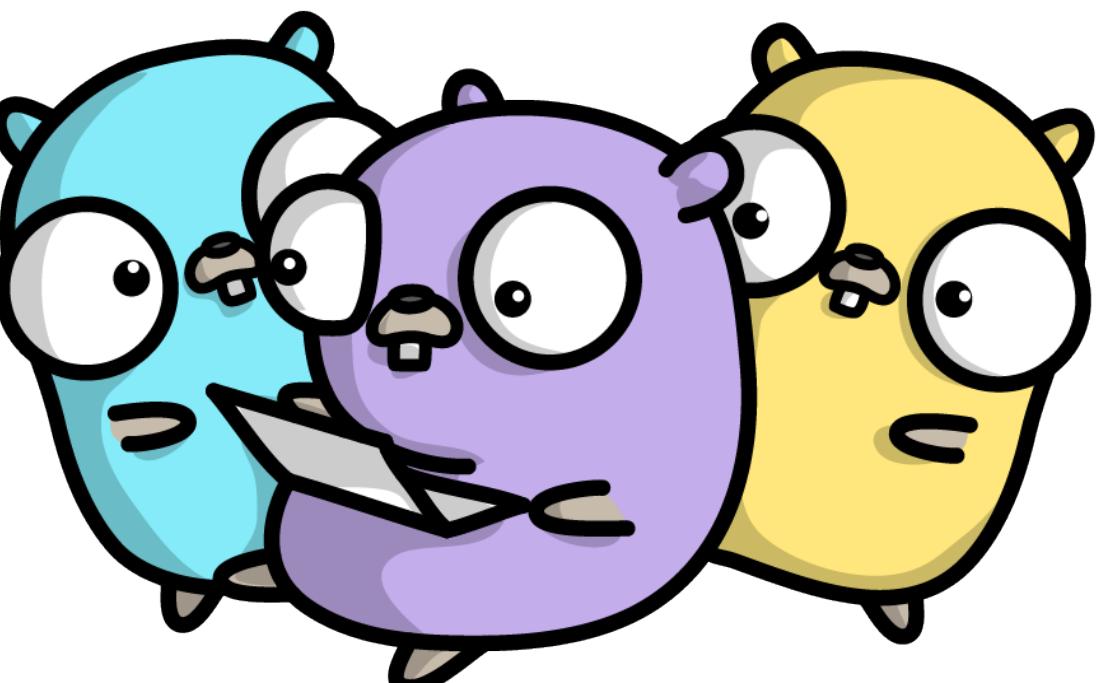
### NET/HTTP

- ▶ Go implementation of HTTP
- ▶ [net/http](#) has a function called `ListenAndServeTLS`
- ▶ `ListenAndServeTLS` provides the desired certificate checking functionality
- ▶ "If the certificate is signed by a certificate authority, the certFile should be the concatenation of the server's certificate, any intermediates, and the CA's certificate."



### MAIN: MUX, CFG, SRV

- ▶ Code creates a `mux`, short for HTTP request multiplexer
- ▶ I ❤️ multiplexers (it's a long story that involves analog signals)
- ▶ `mux` has a function that creates an HTTP server with headers and content (Hello World!)
- ▶ `cfg` brings in all the TLS bits seen in a solid web server config
- ▶ `srv` puts the pieces together and defines what port to listen on



# FAIL SPECTACULARLY

- ▶ I ❤️ DevOps and I embrace failure
- ▶ `log.Fatal(srv.ListenAndServeTLS("/etc/ssl-tester/tls.crt", "/etc/ssl-tester/tls.key"))`
- ▶ Defines path of certificate files to use
- ▶ Logs a fatal error if certificate is not valid
- ▶ Fails Fast



# IT'S OPEN SOURCE!

41 lines (38 sloc) | 1.28 KB

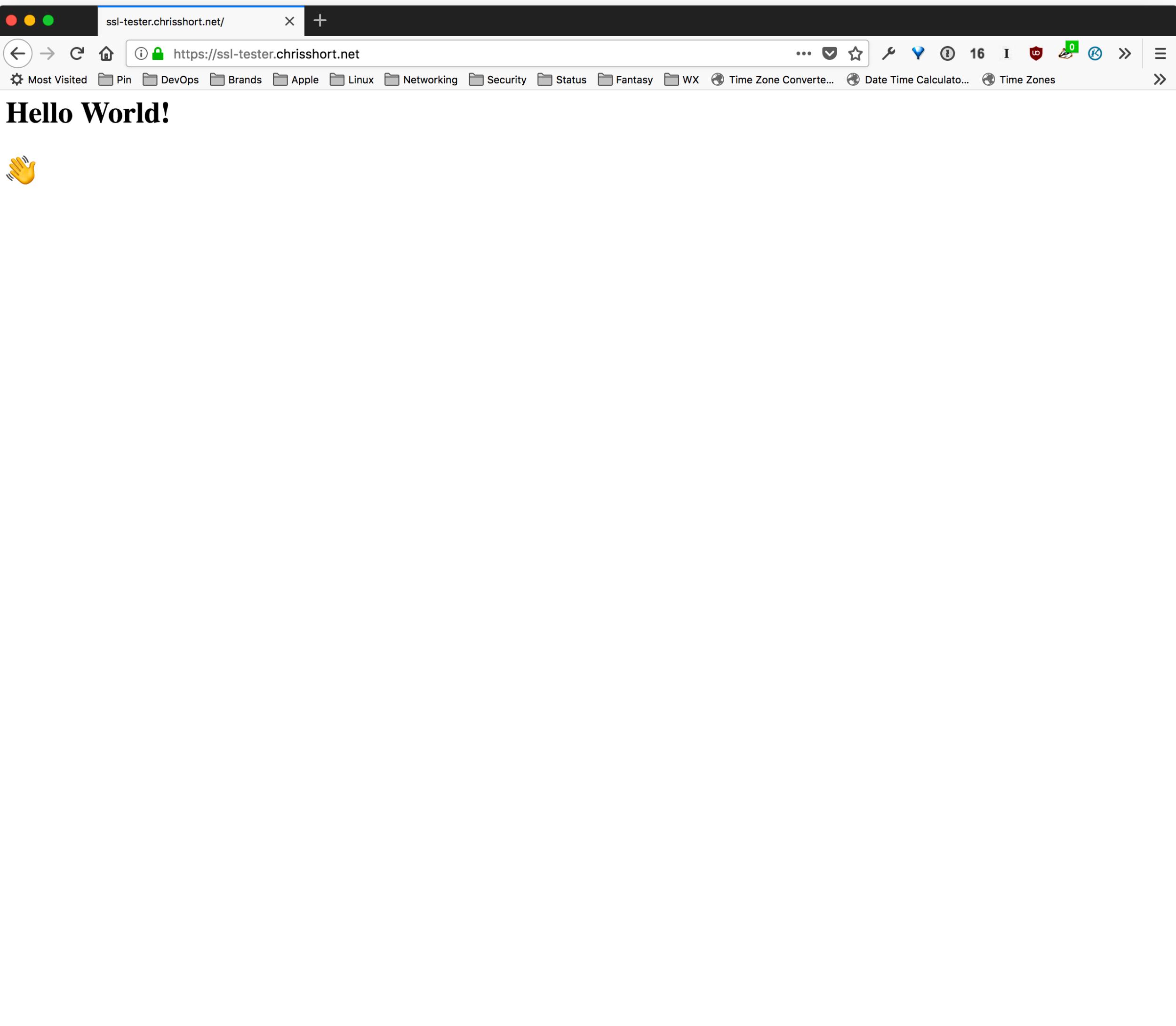
Sourcegraph | Raw | Blame | History |

```
1 package main
2
3 import (
4     "crypto/tls"
5     "log"
6     "net/http"
7 )
8
9 func main() {
10    mux := http.NewServeMux()
11    mux.HandleFunc("/", func(w http.ResponseWriter, req *http.Request) {
12        w.Header().Add("Strict-Transport-Security", "max-age=63072000;")
13        w.Write([]byte("<h1>Hello World!</h1>\n<h1>👋</h1>"))
14    })
15    cfg := &tls.Config{
16        MinVersion:          tls.VersionTLS12,
17        CurvePreferences:   []tls.CurveID{tls.CurveP521, tls.CurveP384, tls.CurveP256},
18        PreferServerCipherSuites: true,
19        CipherSuites:         []uint16{
20            tls.TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA,
21            tls.TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA,
22            tls.TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,
23            tls.TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,
24            // POLY1305 ciphers are not in Go 1.6 and 1.7
25            //           tls.TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305,
26            //           tls.TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305,
27            tls.TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,
28            tls.TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,
29            tls.TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,
30            tls.TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,
31        },
32    }
33    srv := &http.Server{
34        Addr:      ":443",
35        Handler:   mux,
36        TLSConfig: cfg,
37        TLSNextProto: make(map[string]func(*http.Server, *tls.Conn, http.Handler), 0),
38    }
39    log.Fatal(srv.ListenAndServeTLS("/etc/ssl-tester/tls.crt", "/etc/ssl-tester/tls.key"))
40 }
```

<https://github.com/chris-short/ssl-tester>

# IT WORKS!

---



# NO. IT REALLY WORKS!

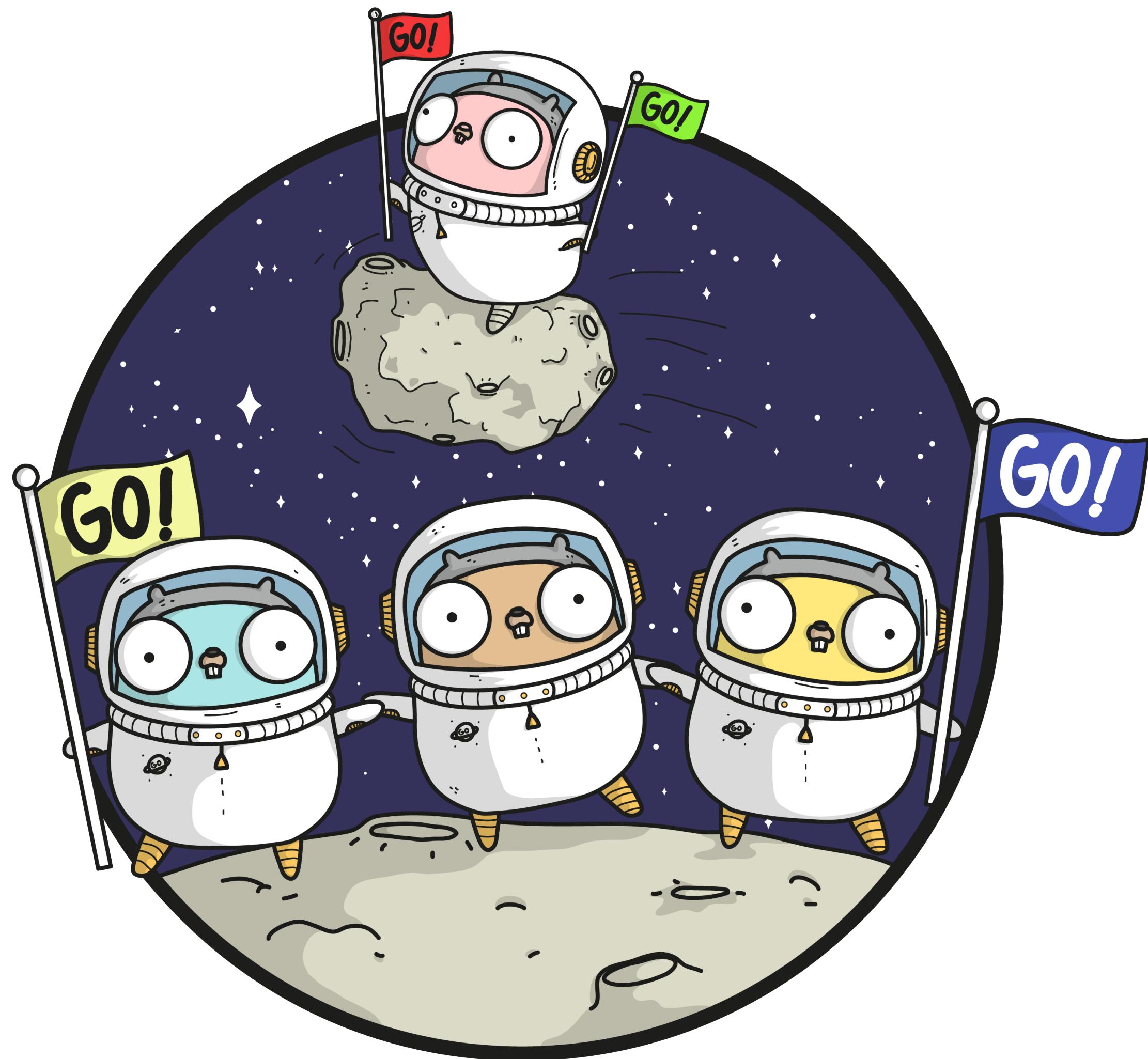
The screenshot shows a browser window displaying the Qualys SSL Labs SSL Report for the domain `ssl-tester.chrisshort.net`. The report is dated Sun, 14 Jan 2018 20:23:35 UTC. The overall rating is **A+**. The summary section includes four horizontal bars representing different security metrics: Certificate (green), Protocol Support (green), Key Exchange (green), and Cipher Strength (green). Below the summary, there is a yellow box containing a link to the documentation page and a green box indicating HSTS deployment. The detailed certificate information for Certificate #1 is listed below.

**Certificate #1: RSA 2048 bits (SHA256withRSA)**

Server Key and Certificate #1	
Subject	ssl-tester.chrisshort.net Fingerprint SHA256: 148b7927f91ea33e84771d833898b2516e76df8a6e2b83a455ce6111646c963e Pin SHA256: 28P6HGeFpvkOeNsZhg7Q5vLPC+kLJam8pewXNlysEN0=
Common names	ssl-tester.chrisshort.net
Alternative names	ssl-tester.chrisshort.net
Serial Number	03755ed61b6bcc6fd7a78a589661848e3122
Valid from	Sun, 10 Dec 2017 14:05:23 UTC
Valid until	Sat, 10 Mar 2018 14:05:23 UTC (expires in 1 month and 23 days)
Key	RSA 2048 bits (e 65537)
Weak key (Debian)	No

## RECAP

- ▶ The Go code does exactly what I need it to do
- ▶ About 40 lines of code!!! I ❤️ Go!
- ▶ Binary is a self contained web server
- ▶ Compiles to less than 6MB!!! I ❤️ Go!
- ▶ Can be safely deployed to any public server
- ▶ External testing run against it for extra vetting





KEEP  
CALM  
AND  
LEARN  
GO

CLEAR IS BETTER THAN CLEVER.

Rob Pike

## CONCLUSION

---

