When You are Ready to GO Beyond PYTHON

Frank Seesink, UNC Chapel Hill

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

First, a message from our sponsor…

What I picture in my head…

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

What it ends up looking like…

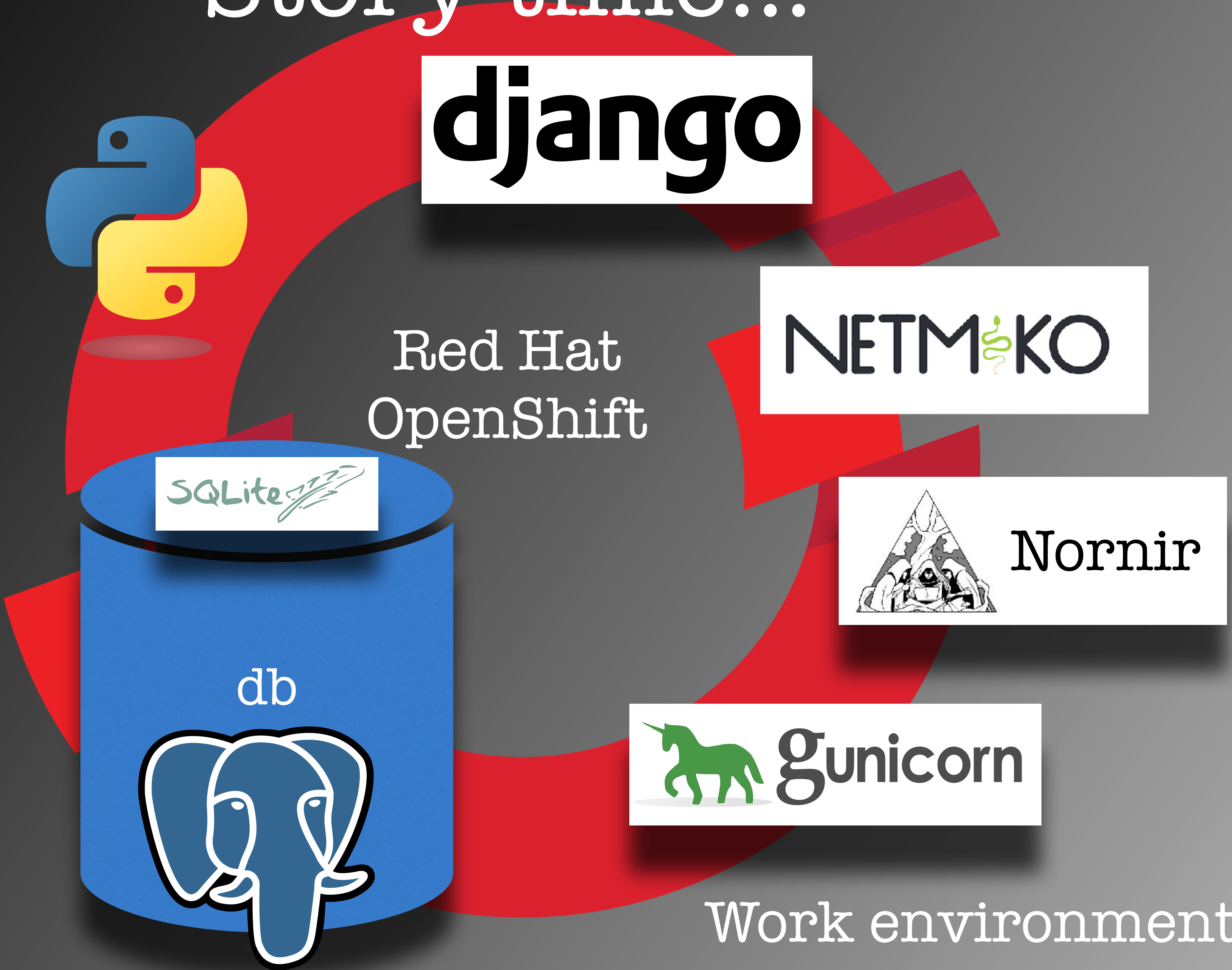
Actually that's not quite right. The guy who made this is **clearly** more talented.

Who am I?

Frank Seesink

- Senior Network Engineer, UNC Chapel Hill
- Part of network DevOps group
- Involved in network automation for years
- Love languages, both human & computer
- Programming since I was 12 years old
- Formally B.S. in Computer Science with all coursework for an M.S. in same
- JOAT - databases, OSes, networking,...

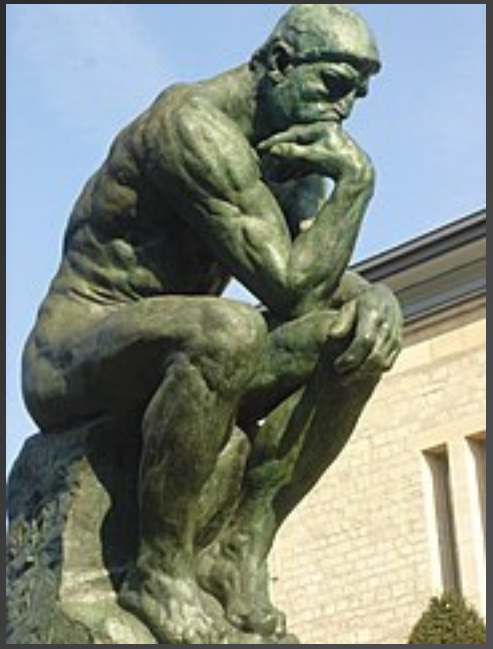
Story time...



Work environment

Story time…

In January 2022, I was in a rut…



Why Go?

- Python's creator, Guido van Rossum, worked at Google from 2005-2012.
- For years Google heavily used Python internally and even offered Python classes to its employees.
    - https://developers.google.com/edu/python
- Google had also hired Rob Pike and Ken Thompson of Bell Labs (UNIX, C) fame. They, along with Robert Griesemer, created Go.
- In 2013 Guido van Rossum went to work at Dropbox. (Dropbox was known to use Python.) That seemed odd.
- In 2014 Google publicly released Kubernetes, which is written in Go.

The writing was on the wall?

Why Go?

containerd

HELM

docker

GO

“Language of the cloud”

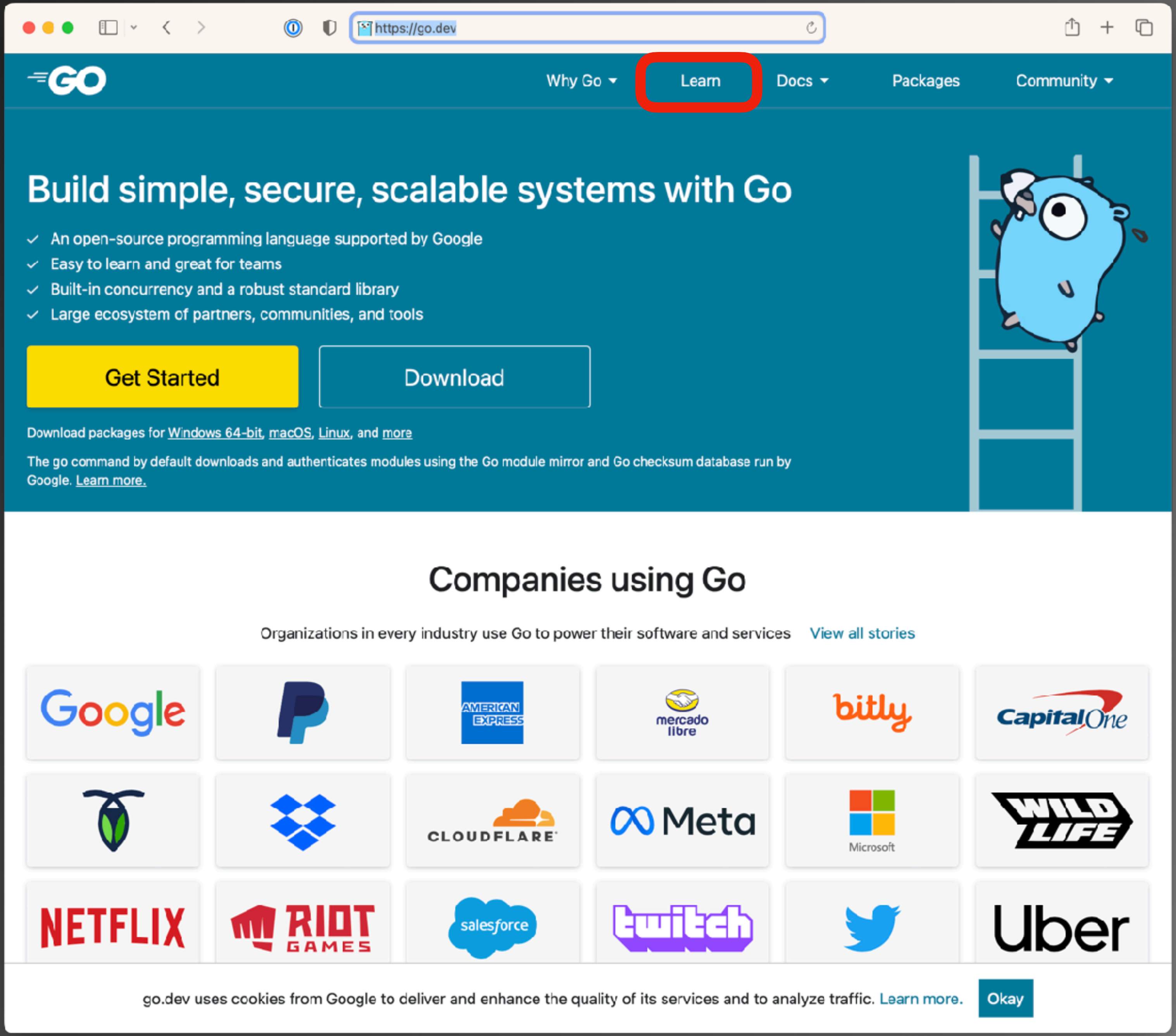
etcd

CoreDNS

influxdb

HashiCorp Terraform

Go (Golang)



Build simple, secure, scalable systems with Go

- ✓ An open-source programming language supported by Google
- ✓ Easy to learn and great for teams
- ✓ Built-in concurrency and a robust standard library
- ✓ Large ecosystem of partners, communities, and tools

Download packages for Windows 64-bit, macOS, Linux, and more

The go command by default downloads and authenticates modules using the Go module mirror and Go checksum database run by Google. Learn more.

Companies using Go

Organizations in every industry use Go to power their software and services View all stories

go.dev uses cookies from Google to deliver and enhance the quality of its services and to analyze traffic. Learn more. Okay

https://go.dev/

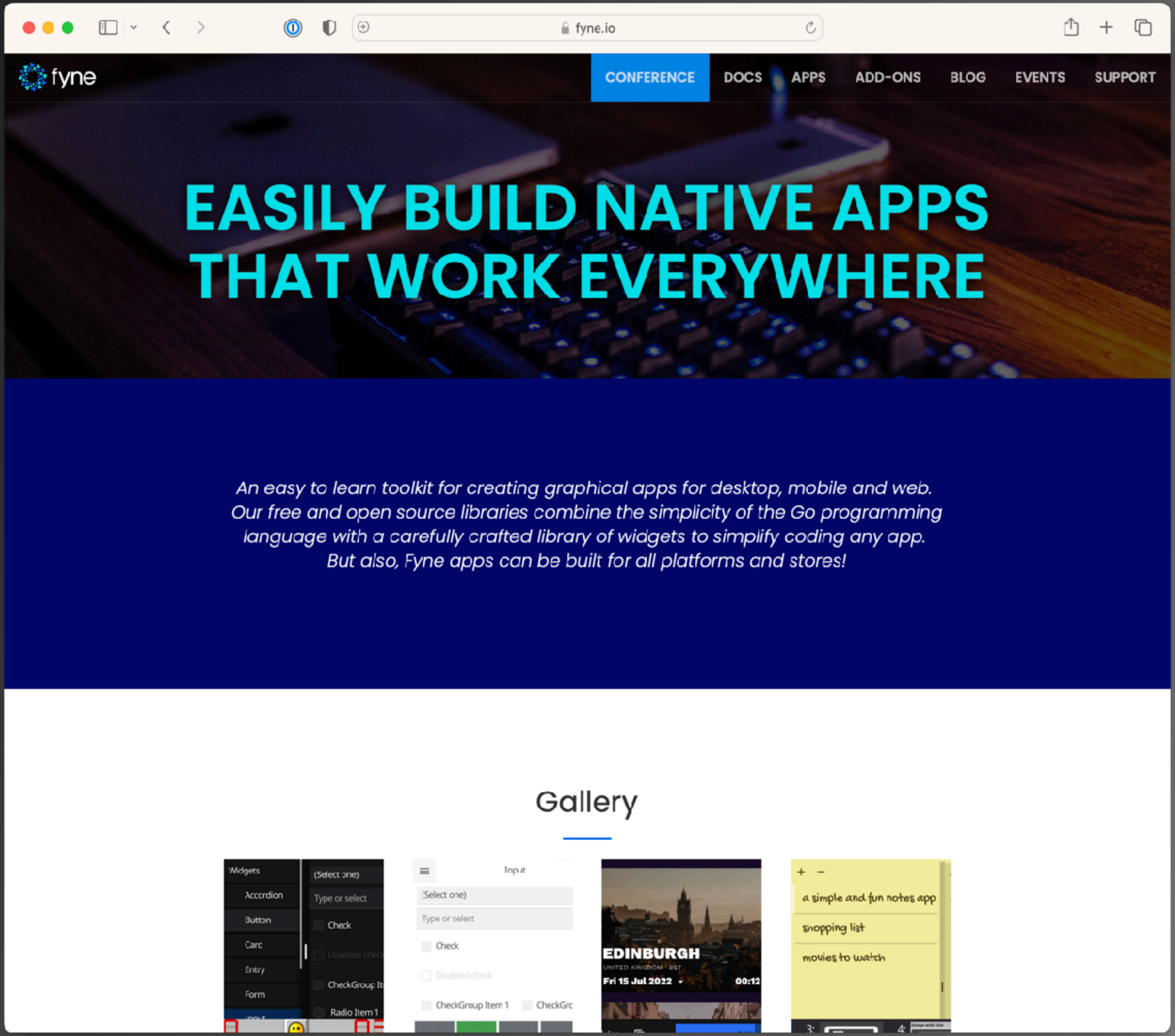
Go (Golang)

LEARNING

- **Learning Go** https://www.linkedin.com/learning/learning-go
- **Go for Python Developers** https://www.linkedin.com/learning/go-for-python-developers
- https://**learnxinyminutes.com**/docs/go/

Fyne

GTK



EASILY BUILD NATIVE APPS THAT WORK EVERYWHERE

An easy to learn toolkit for creating graphical apps for desktop, mobile and web. Our free and open source libraries combine the simplicity of the Go programming language with a carefully crafted library of widgets to simplify coding any app. But also, Fyne apps can be built for all platforms and stores!

Qt Group

tcl tk UNIVERSAL SCRIPTING

wxWidgets

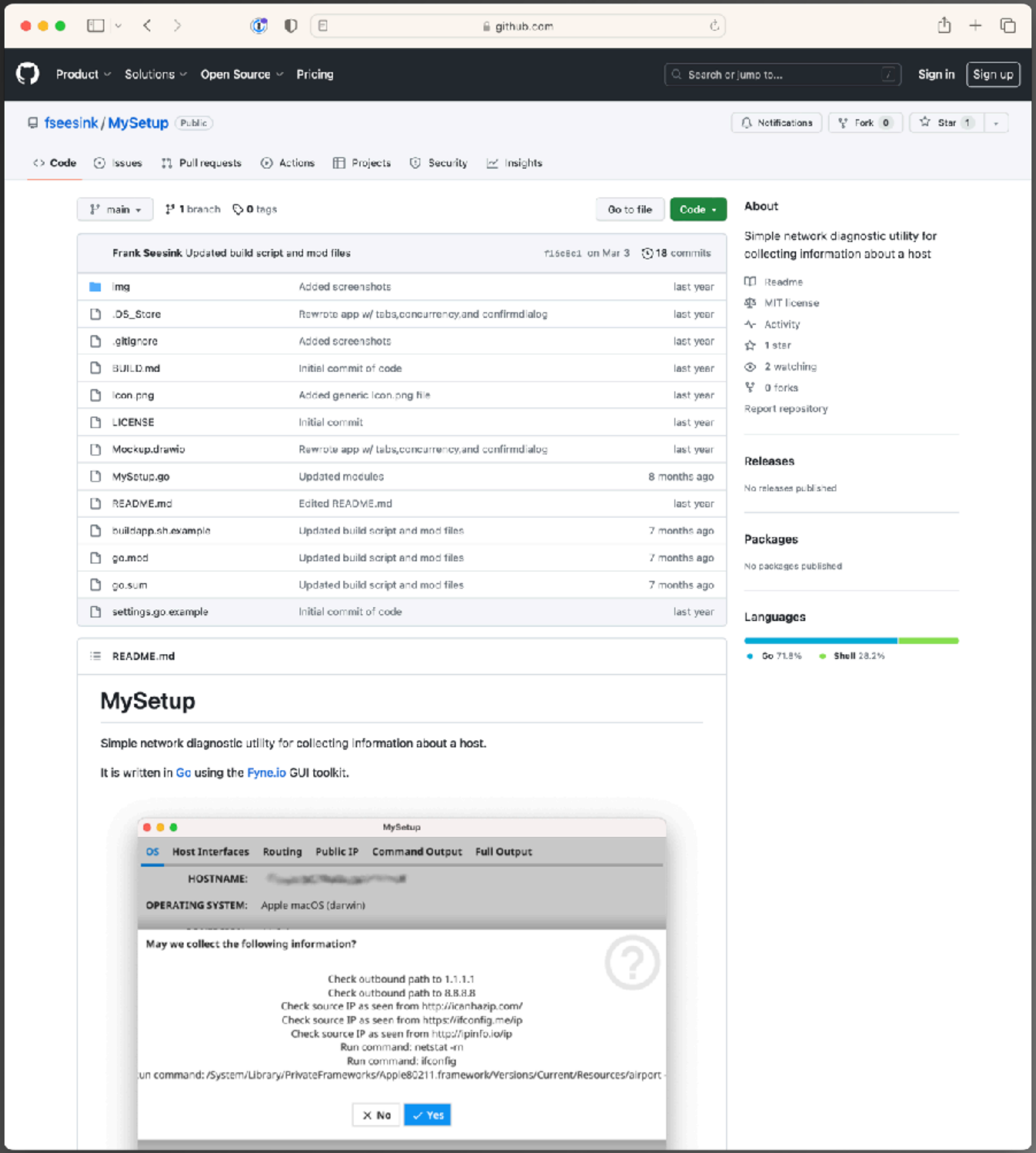
https://fyne.io/

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

To Learn a Programming Language...

1. You need to program in it
2. You need to program in it
3. You need to program in it
4. You need to have a project/goal

Initial Go Test Project



fseesink / MySetup Public

Frank Seesink Updated build script and mod files f15e8c1 on Mar 3 18 commits

img	Added screenshots	last year
.DS_Store	Rewrote app w/ tabs,concurrency,and confirmdialog	last year
.gitignore	Added screenshots	last year
BUILD.md	Initial commit of code	last year
Icon.png	Added generic Icon.png file	last year
LICENSE	Initial commit	last year
Mockup.drawio	Rewrote app w/ tabs,concurrency,and confirmdialog	last year
MySetup.go	Updated modules	8 months ago
README.md	Edited README.md	last year
buildapp.sh.example	Updated build script and mod files	7 months ago
go.mod	Updated build script and mod files	7 months ago
go.sum	Updated build script and mod files	7 months ago
settings.go.example	Initial commit of code	last year

About

Simple network diagnostic utility for collecting information about a host

Readme
MIT license
Activity
1 star
2 watching
0 forks
Report repository

Releases

No releases published

Packages

No packages published

Languages

● Go 71.8% ● Shell 28.2%

README.md

MySetup

Simple network diagnostic utility for collecting information about a host.

It is written in Go using the Fyne.io GUI toolkit.

HOSTNAME: [redacted]

OPERATING SYSTEM: Apple macOS (darwin)

May we collect the following information?

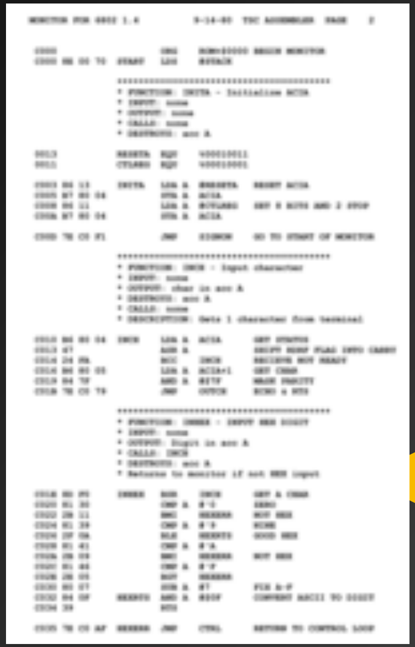
Check outbound path to 1.1.1.1
Check outbound path to 8.8.8.8
Check source IP as seen from http://icanhazip.com/
Check source IP as seen from https://ifconfig.me/ip
Check source IP as seen from http://ipinfo.io/ip
Run command: netstat -rn
Run command: ifconfig
un command: /System/Library/PrivateFrameworks/Apple80211.framework/Versions/Current/Resources/airport -

fyne.io

https://github.com/fseesink/mysetup

History of Programming Languages

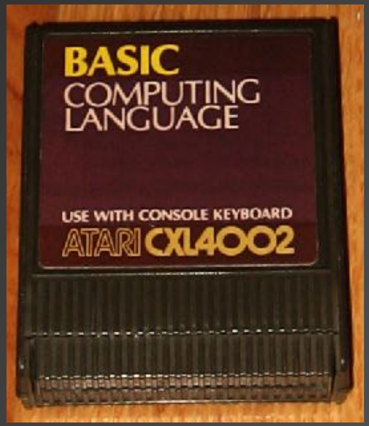
0/1



```
[illegible]
```

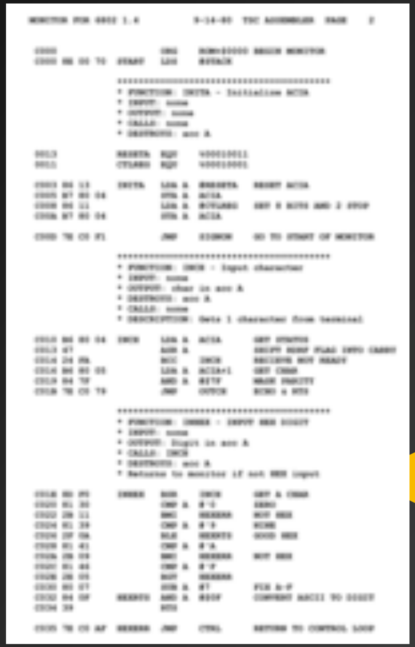
1940s

Present



History of Programming Languages

0/1

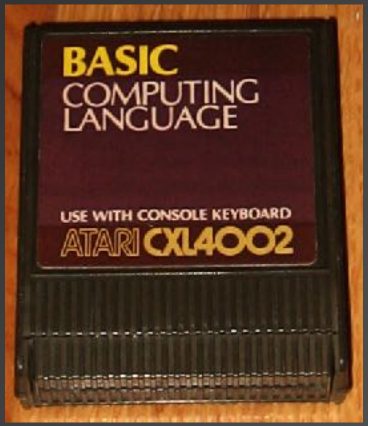


```
MONITOR FOR 6802 1.4          9-14-80  TSC ASSEMBLER  PAGE   2

C000                    ORG    ROM+$0000 BEGIN MONITOR
C000 8E 00 70  START    LDS    #STACK

               ****************************************
               * FUNCTION: INITA - Initialize ACIA
               * INPUT: none
               * OUTPUT: none
               * CALLS: none
               * DESTROYS: acc A

0013           RESETA   EQU    %00010011
0011           CTLREG   EQU    %00010001

C003 86 13     INITA    LDA A  #RESETA   RESET ACIA
C005 B7 80 04           STA A  ACIA
C008 86 11              LDA A  #CTLREG   SET 8 BITS AND 2 STOP
C00A B7 80 04           STA A  ACIA

C00D 7E C0 F1           JMP    SIGNON    GO TO START OF MONITOR

               ****************************************
               * FUNCTION: INCH - Input character
               * INPUT: none
               * OUTPUT: char in acc A
               * DESTROYS: acc A
               * CALLS: none
               * DESCRIPTION: Gets 1 character from terminal

C010 B6 80 04  INCH     LDA A  ACIA      GET STATUS
C013 47                 ASR A            SHIFT RDRF FLAG INTO CARRY
C014 24 FA              BCC    INCH      RECIEVE NOT READY
C016 B6 80 05           LDA A  ACIA+1    GET CHAR
C019 84 7F              AND A  #$7F      MASK PARITY
C01B 7E C0 79           JMP    OUTCH     ECHO & RTS

               ****************************************
               * FUNCTION: INHEX - INPUT HEX DIGIT
               * INPUT: none
               * OUTPUT: Digit in acc A
               * CALLS: INCH
               * DESTROYS: acc A
               * Returns to monitor if not HEX input

C01E 8D F0     INHEX    BSR    INCH      GET A CHAR
C020 81 30              CMP A  #'0       ZERO
C022 2B 11              BMI    HEXERR    NOT HEX
C024 81 39              CMP A  #'9       NINE
C026 2F 0A              BLE    HEXRTS    GOOD HEX
C028 81 41              CMP A  #'A
C02A 2B 09              BMI    HEXERR    NOT HEX
C02C 81 46              CMP A  #'F
C02E 2E 05              BGT    HEXERR
C030 80 07              SUB A  #7        FIX A-F
C032 84 0F     HEXRTS   AND A  #$0F      CONVERT ASCII TO DIGIT
C034 39                 RTS

C035 7E C0 AF  HEXERR   JMP    CTRL      RETURN TO CONTROL LOOP
```

PROGRAMMING REPUBLIC OF PERL

php

1940s

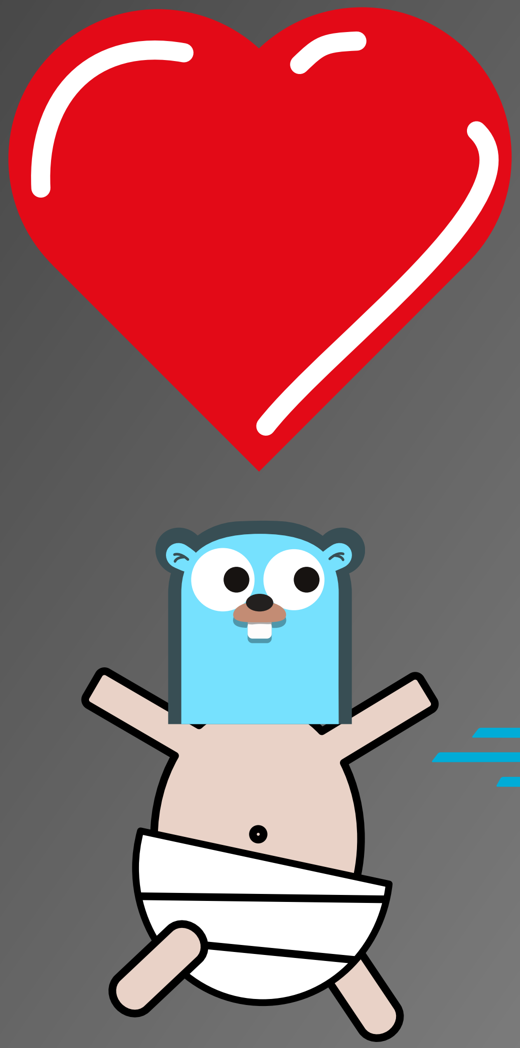


&

GO

Present

History of Programming Languages



History/Comparison

	C	Python	Go
First appeared	1972	1992	2009
Designed by	Dennis Ritchie	Guido van Rossum	Robert Griesemer Rob Pike Ken Thompson
Typing	Static, weak, manifest, nominal	Duck, dynamic, strong typing	Inferred, static, strong, structural, nominal
Keywords	32	35	25

Features

	C	Python	Go
Built-in concurrency	N/A	N/A	Go routines
Concurrency via libraries	fork() *provides access to underlying OS concurrency features	multiprocessing concurrent.futures asyncio	⬆
Native multi-core support	N/A	N/A due to GIL	⬆
Memory Management	malloc()/free() *developer responsible for all memory mgmt	Garbage Collection	Garbage Collection

Libraries/Modules

	C	Python	Go
Standard Library	✅	✅	✅
Package ecosystem	N/A	PyPI.org	via VCS such as Git
Example package import	#include <stdio.h>	import netmiko	import (“github.com/ nornir-automation/ gornir/pkg/gornir")
Largest library (e.g., AI/ML, data analysis)		✅	

Workflow

Python	Go
`#!//usr/local/bin/python3` `print("Hello world")`	`package main` `import "fmt"` `func main() { fmt.Println("Hello world") }`
`$ python3 helloworld.py` or if permissions set, simply `$ helloworld.py`	`$ go run helloworld.go` or `$ go run .` to run interactively. Compile and run executable with `$ go build .` `$ helloworld`

Python

```python
#!/usr/local/bin/python3

print("Hello world")
```

```
$ python3 helloworld.py
```

or if permissions set, simply

```
$ helloworld.py
```

Go

```go
package main

import "fmt"

func main() {
    fmt.Println("Hello world")
}
```

```
$ go run helloworld.go
```

or

```
$ go run .
```

to run interactively.

Compile and run executable with

```
$ go build .
$ helloworld
```

Workflow Performance

Python

```python
#!/usr/local/bin/python3

print("Hello world")
```

```
$ time python3 helloworld.py
Hello world
python3 helloworld.py  0.02s
user 0.02s system 36% cpu
0.111 total
```

Time to run executable binary

Go

```go
package main

import "fmt"

func main() {
    fmt.Println("Hello world")
}
```

Time to compile AND run the program (when developing)

```
$ time go run helloworld.go
Hello world
go run helloworld.go  0.14s
user 0.29s system 49% cpu
0.860 total
$ go build helloworld.go
$ time ./helloworld
Hello world
./helloworld  0.00s user 0.00s
system 2% cpu 0.135 total
```

Final Program Size

Python

```python
#!/usr/local/bin/python3

print("Hello world")
```

C version
TOTAL == 32 KB
or 0.032 MB

```
 46 bytes:  helloworld.py
310 MB:     Python install (*)
```

To run a Python script, you need Python installed.

TOTAL == ~310 MB

(*) v3.11.5 macOS installation on disk

Go

```go
package main

import "fmt"

func main() {
    fmt.Println("Hello world")
}
```

```
72 bytes:  helloworld.go
238 MB:    Go install (*)
1.8 MB:    helloworld binary
```

To run a Go compiled app, you just need the binary.

TOTAL == 1.8 MB

(*) v1.21.0 macOS installation on disk

Language Similarities

Python

```python
import os


def itsvalid():
    print("Valid day of the month")
    cwd = os.getcwd()
    print(cwd)


def main():
    # Variable assignment
    name = "Frank"
    day = 19

    if day >= 1 and day < 31:
        itsvalid()


if __name__ == "__main__":
    main()
```

Go

```go
package main

import (
    "fmt"
    "os"
)

func itsvalid() {
    fmt.Println("Valid day of the month")
    cwd, _ := os.Getwd()
    fmt.Println(cwd)
}

func main() {
    // Variable assignment
    name := "Frank"
    day := 19

    if day >= 1 && day < 31 {
        itsvalid()
    }

    fmt.Println(name)
}
```

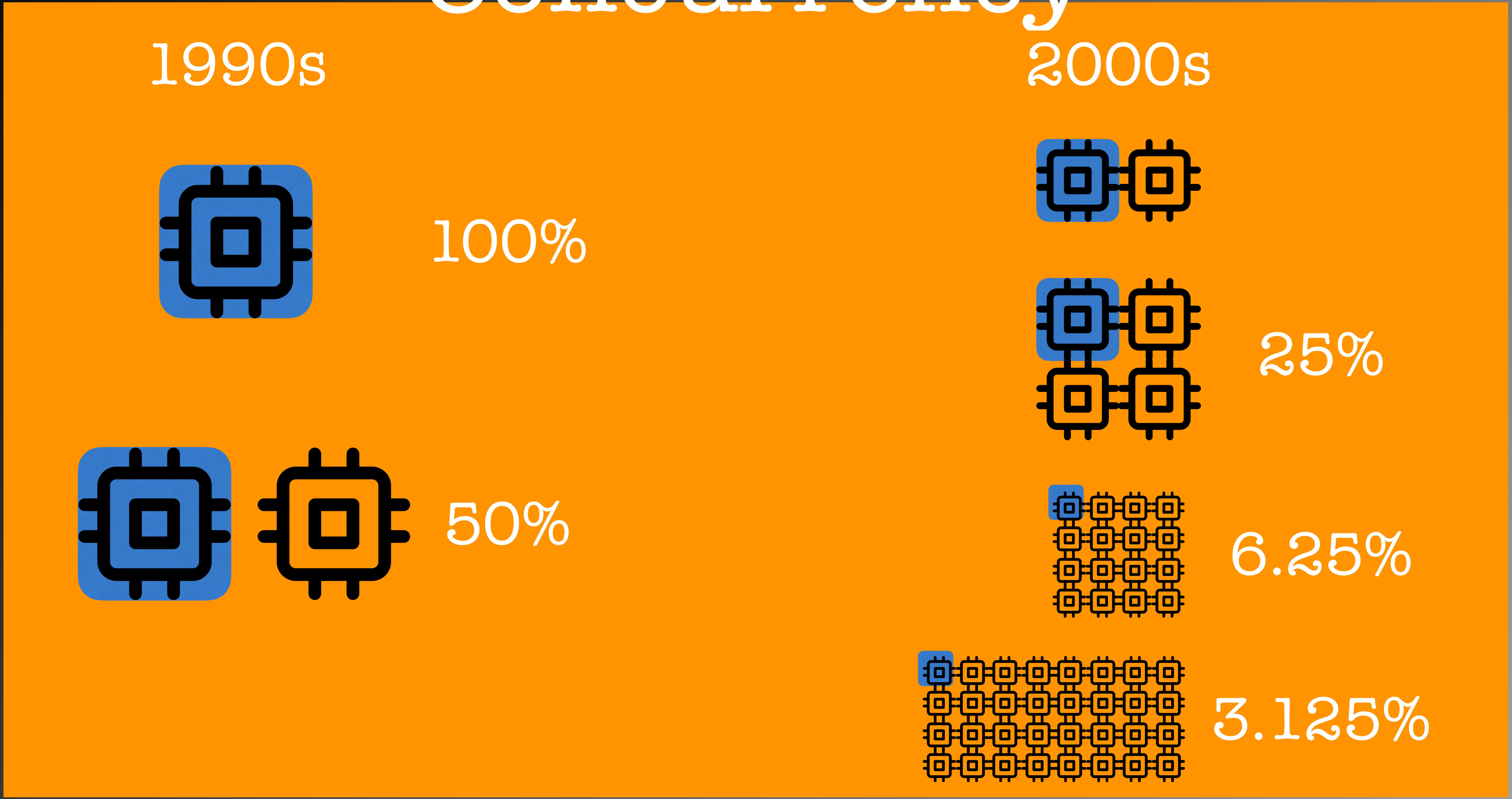
Global Interpreter Lock (GIL)

GIL

“In CPython, the global interpreter lock, or GIL, is a mutex that protects access to Python objects, preventing multiple threads from executing Python bytecodes at once. The GIL prevents race conditions and ensures thread safety. A nice explanation of how the Python GIL helps in these areas can be found here. In short, this mutex is necessary mainly because CPython's memory management is not thread-safe.”

- https://wiki.python.org/moin/GlobalInterpreterLock

Python's Lack of Concurrency



== CPU core

🟦 == Python

To bypass the GIL

To use multiple threads/cores, you must take action. This requires extra effort.

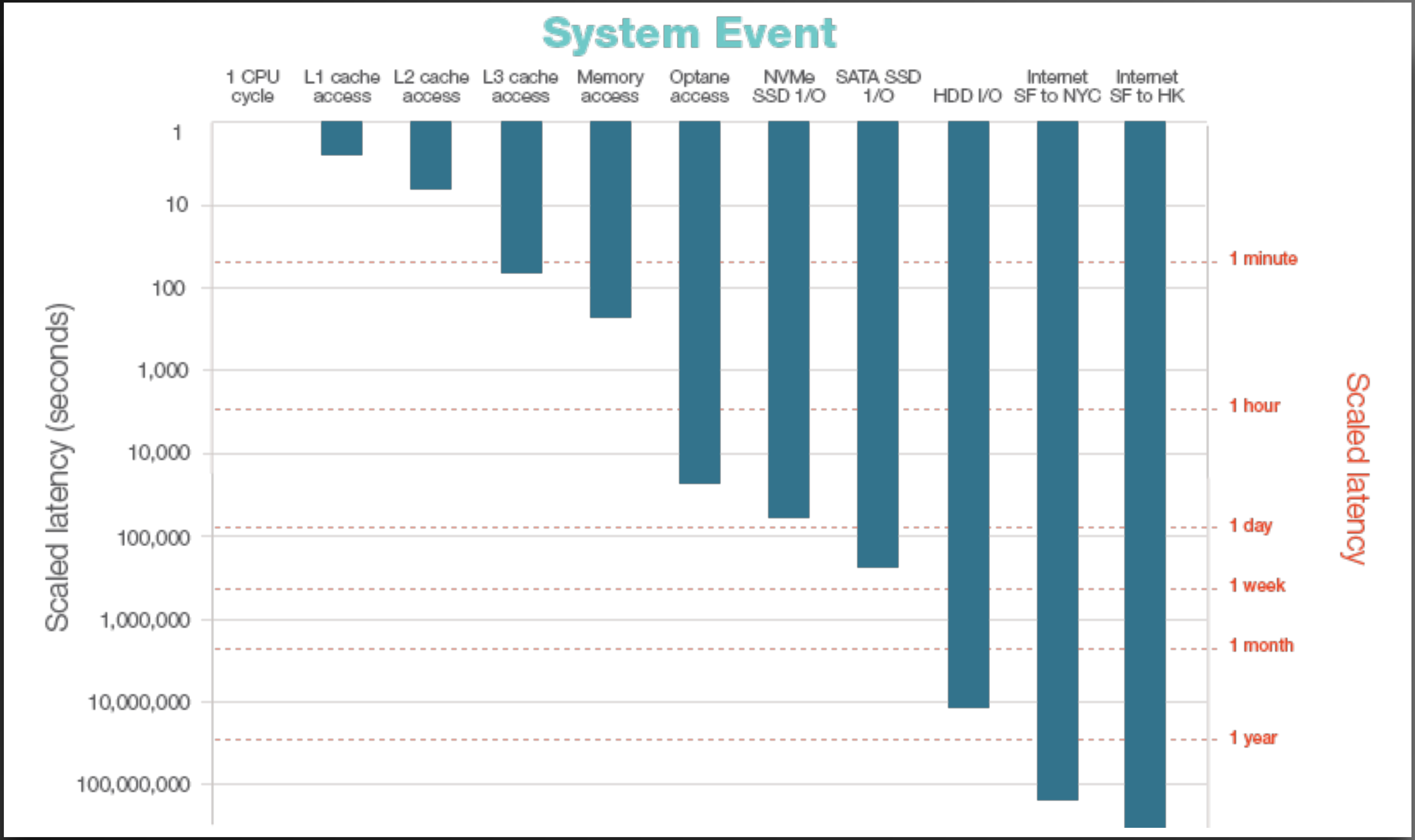
For example,

- multiprocessing or concurrent.futures module in Python standard library. (You must use processes and not threads in latter. Otherwise it stays within a single core.)

- Use modules like Nornir (which use `concurrent futures`)

`asyncio` does NOT help here. That is cooperative multi-threading. Again, single core.

Disclaimer



Network Automation tends to be
I/O-bound vs. CPU-bound

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Possible Python Future

PEP 703 – Making the Global Interpreter Lock Optional in CPython

> CPython's global interpreter lock ("GIL") prevents multiple threads from executing Python code at the same time. The GIL is an obstacle to using multi-core CPUs from Python efficiently. This PEP proposes adding a build configuration (--disable-gil) to CPython to let it run Python code without the global interpreter lock and with the necessary changes needed to make the interpreter thread-safe.

https://peps.python.org/pep-0703/

Go routines

1. Put 'go' in front of a function call.
2. …
3. Profit!

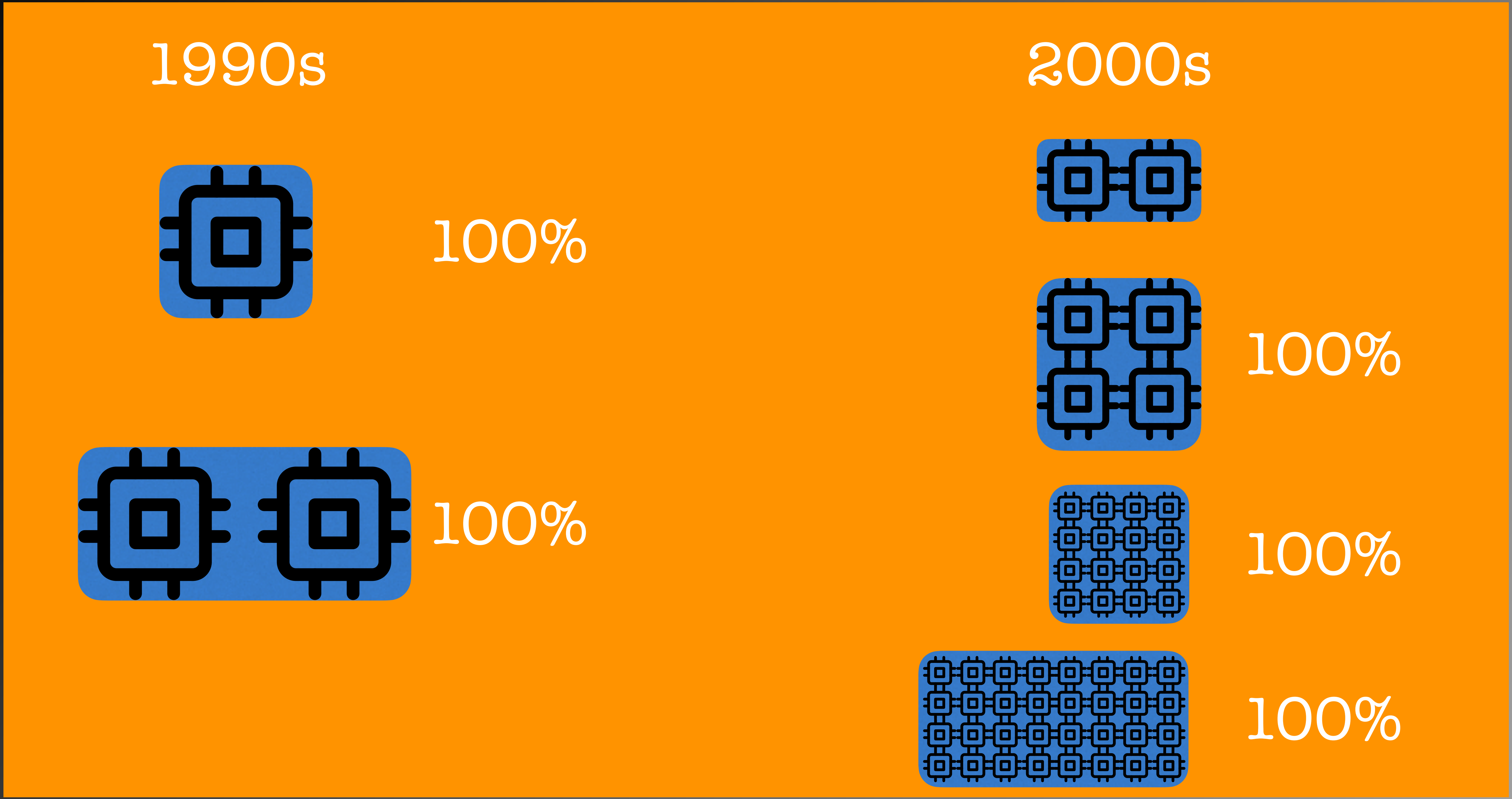
Main routine waits for function

```
func main() {
    // Variable assignment
…
   dosomething()
…
}
```

Main routine keeps going

```
func main() {
    // Variable assignment
…
   go dosomething()
…
}
```

Go routines

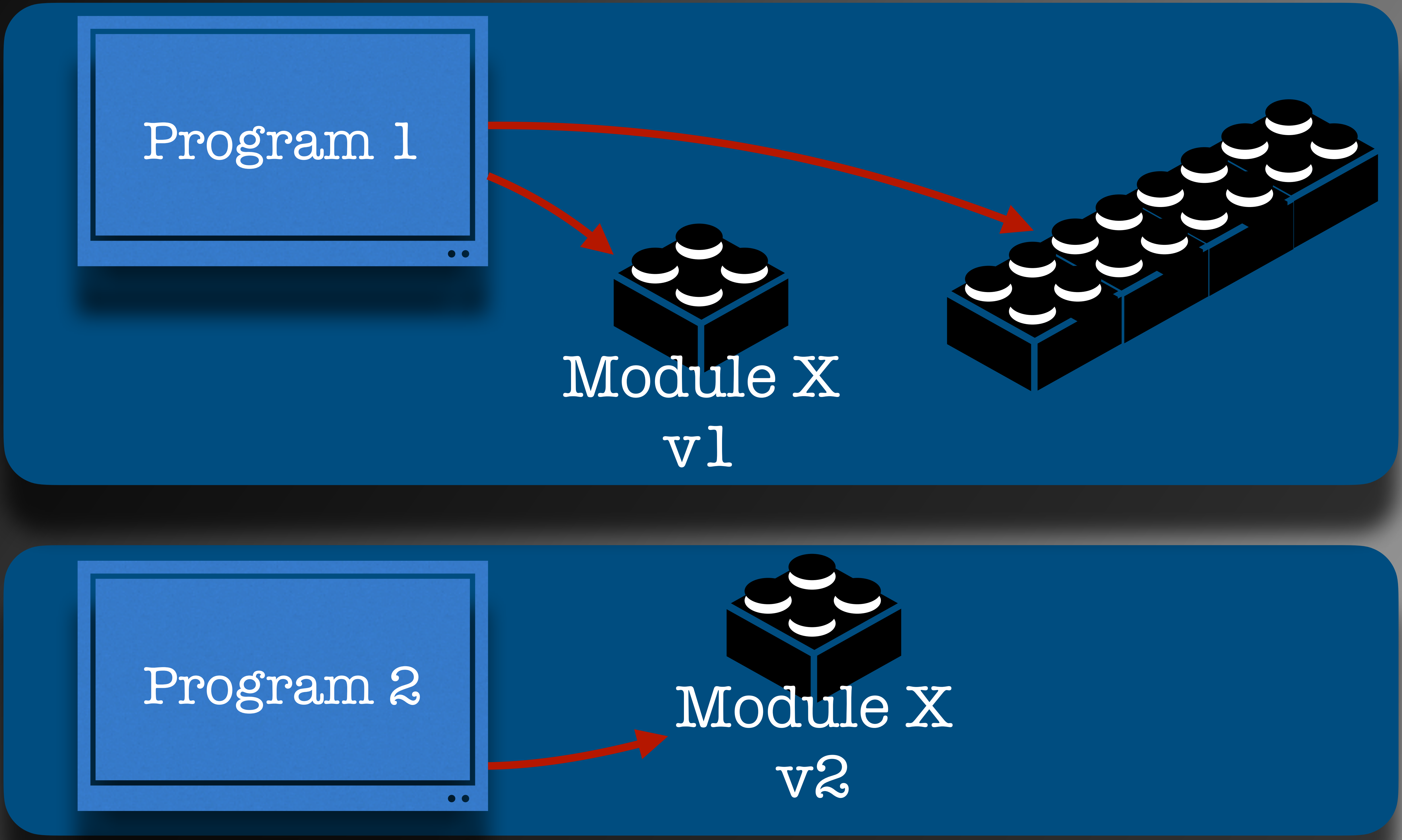


== CPU core

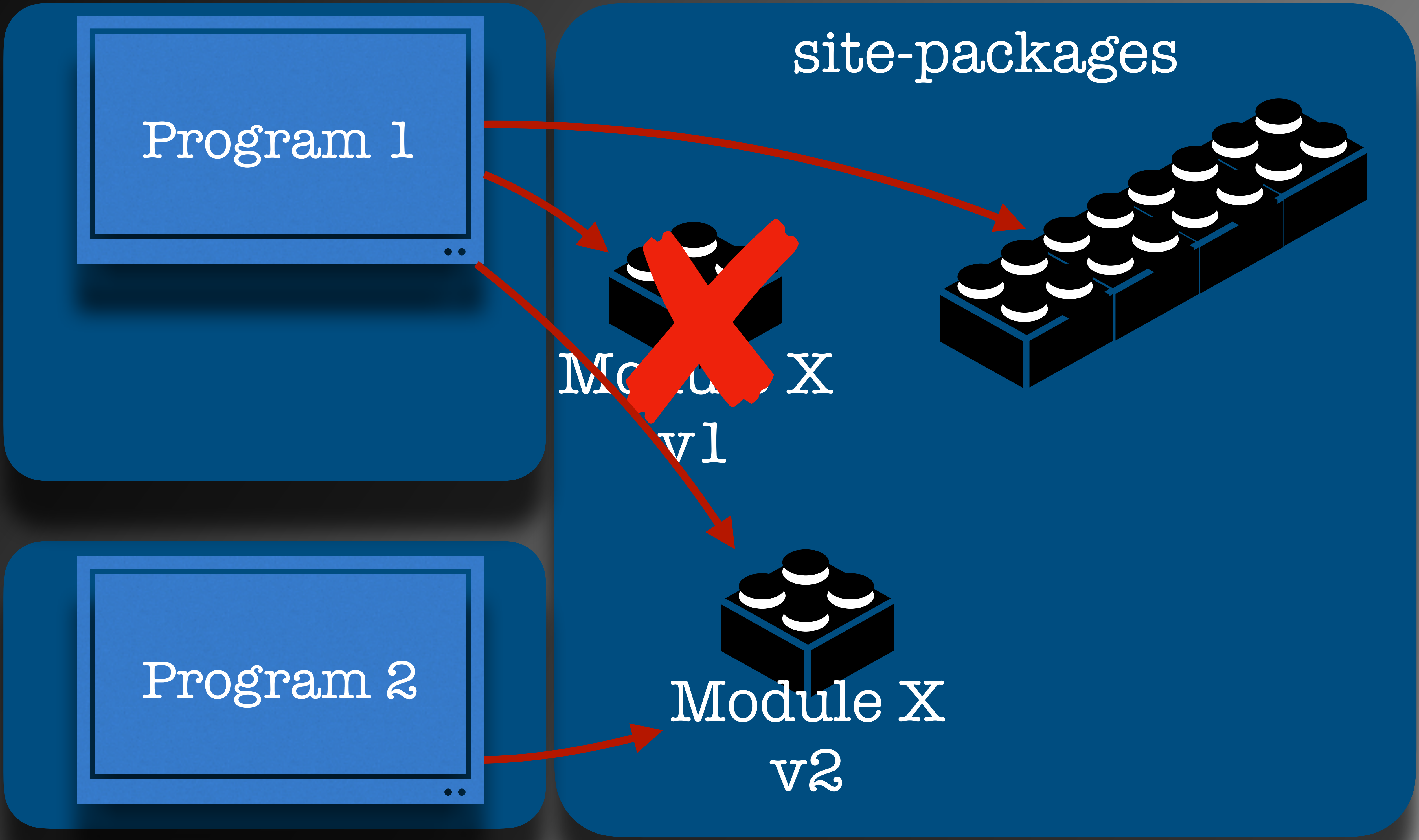
🟦 == Go routines

Dependency Hell

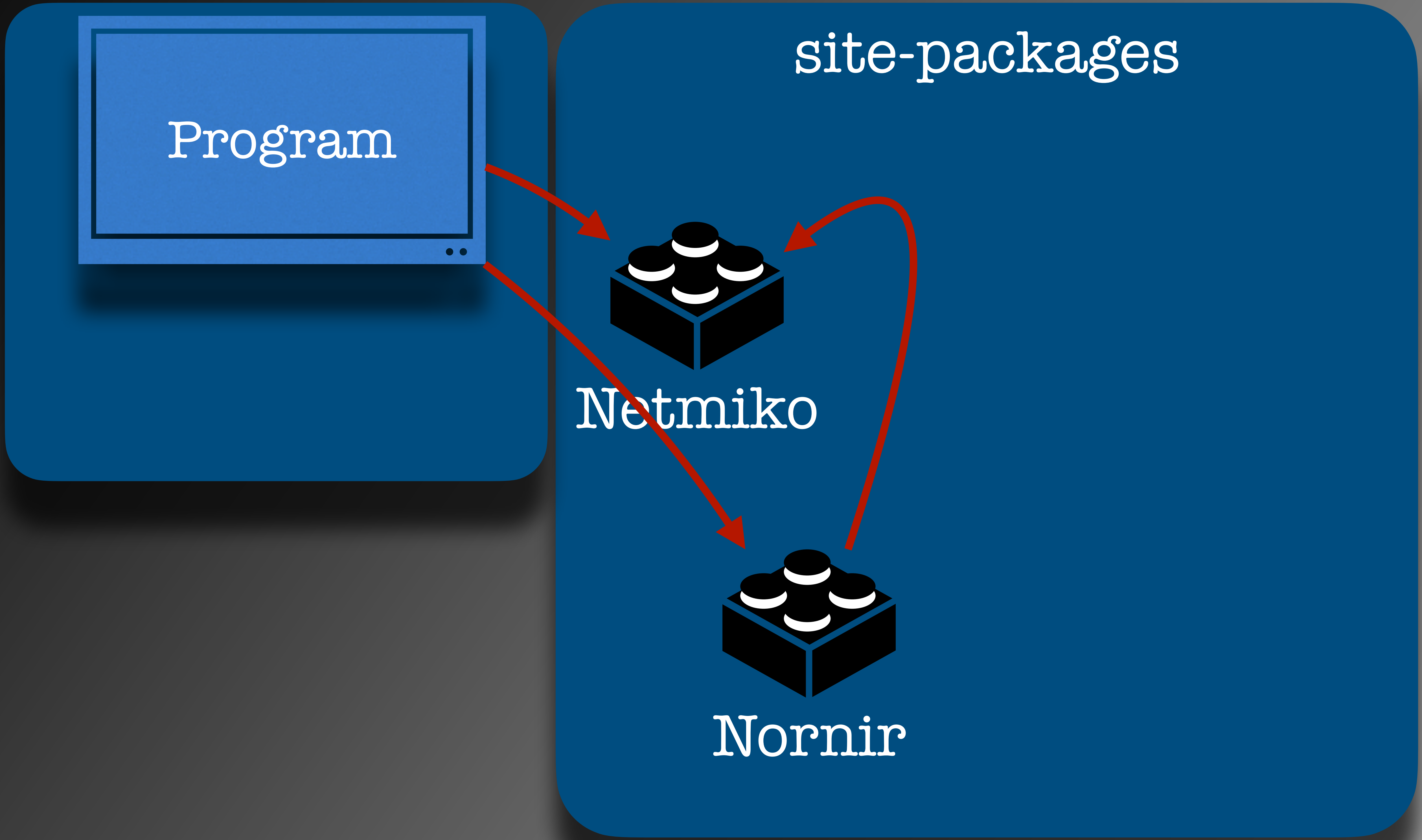
Dependency Hell



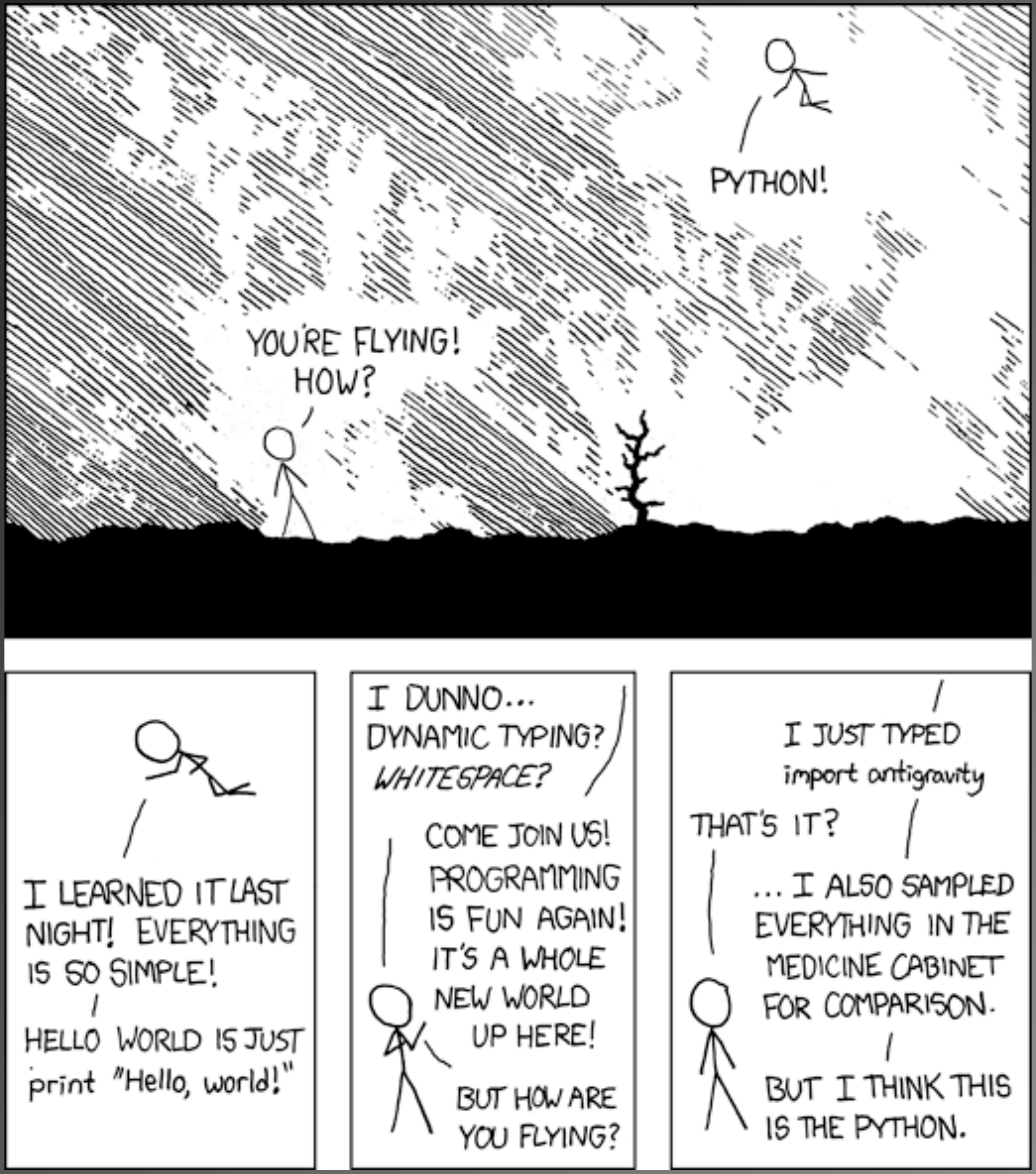
Dependency Hell



Dependency Hell (cont.)



When you first learn Python, it's like this

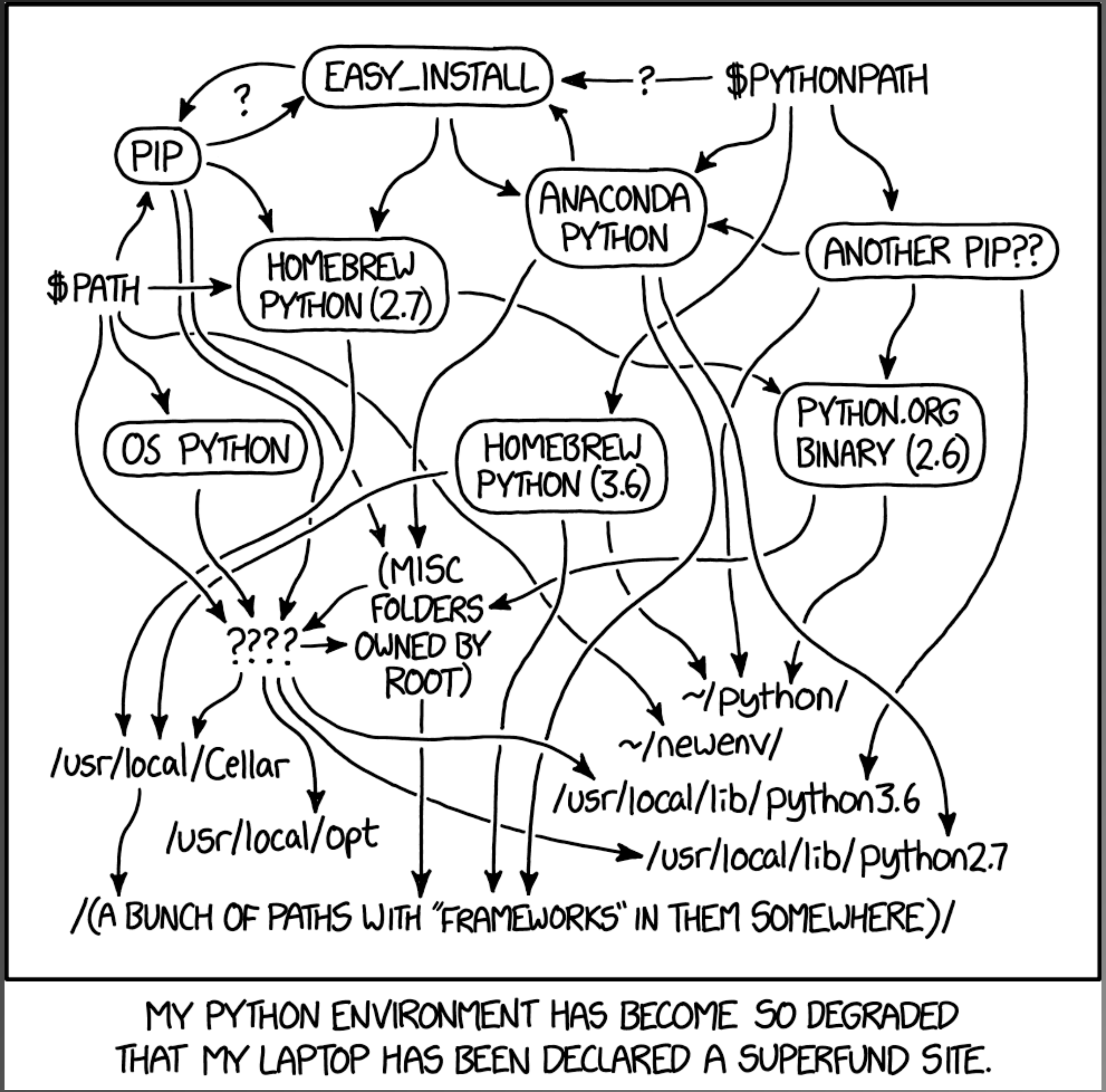


PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

https://xkcd.com/353/

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Eventually, it becomes this...



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

https://xkcd.com/1987/

What about Python's ability to run on different platforms?

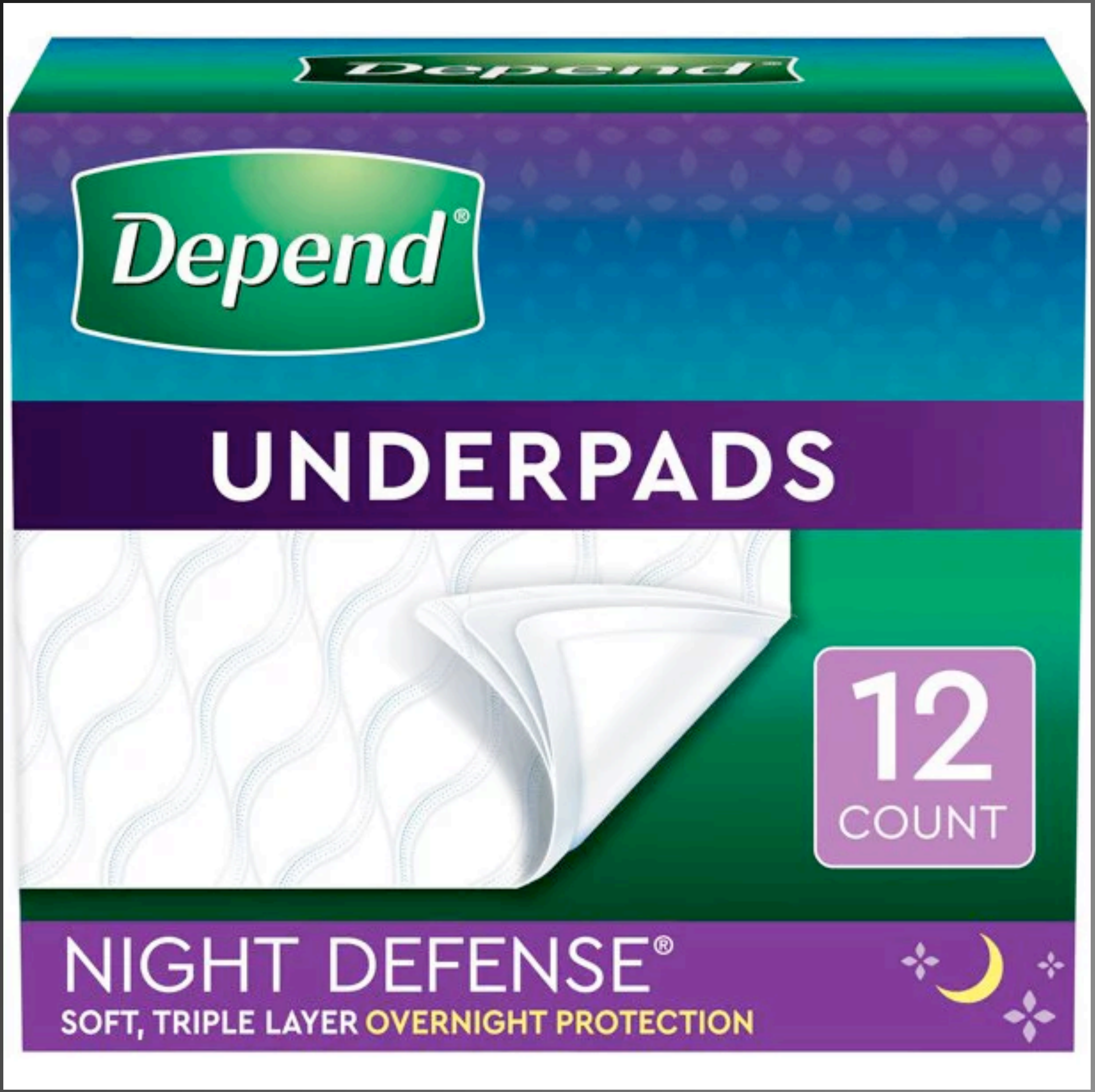
Go Cross-Compilation

Go Cross-Compilation

- Go creates binary executables specific to an OS/architecture (e.g., x64 Windows, ARM64 Linux)
- Go can cross-compile to ANY supported OS/architecture combination FROM any supported OS/architecture. Simply set GOOS and GOARCH environment variables.

```
$ GOOS=linux GOARCH=arm64 go build .
```

So when should you use Go?


Depend
Depend®
UNDERPADS
12
COUNT
NIGHT DEFENSE®
SOFT, TRIPLE LAYER OVERNIGHT PROTECTION

It Depends.

What makes Go worth considering

- Pythonic code (relatively easy transition)
- Go routines / native multi-core support
- Single binary executable with NO EXTERNAL DEPENDENCIES
- Can compile to any supported architecture/ OS from a single platform
- Performant: best balance between coding speed and execution speed
- BONUS: Fyne is a nice, cross-platform GUI framework

Thank You

https://frank.seesink.com/presentations/Internet2TechEx-Fall2023/

Frank Seesink
frank@seesink.com
frank@unc.edu

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL