2021-06-03 @ Ocado, Sofia

Golang To Production #2

Outside the happy path

```
if err !=nil {
    return nil, fmt.Errorf(“error
with error %v”, err)
}
```

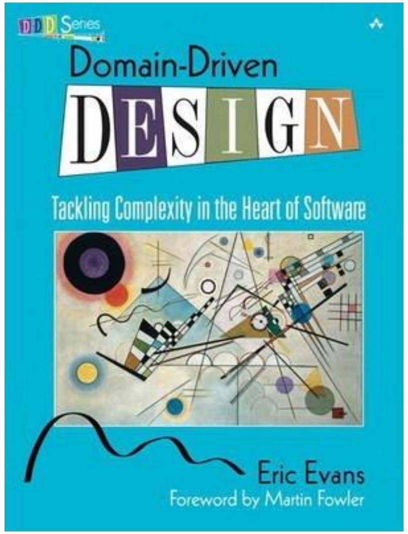
Error handling - a mouthful but explicit

04.

Next time

Even more code while talking about...

- Concurrency and what we have in go
- Testing
- The Standard Library
- Domain Driven Design as applicable to Go
- Prepare for the project!

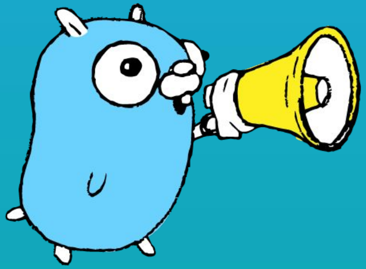

DDD Series
Domain-Driven
DESIGN
Tackling Complexity in the Heart of Software
Eric Evans
Foreword by Martin Fowler

Resources

- Course - https://github.com/bbsbb/golang-at-ocado
- Go By Example - https://gobyexample.com/
- A Tour of Go - https://tour.golang.org
- Simplicity is Complicated (Rob Pike) - Video
- Clear is better than Clever (Dave Cheney) - Video

Go doesn't *implicitly* anything.
-- #go-nuts

https://www.youtube.com/watch?v=VBlFHuCzPgY



Golang To Production w/ GRPC

01.

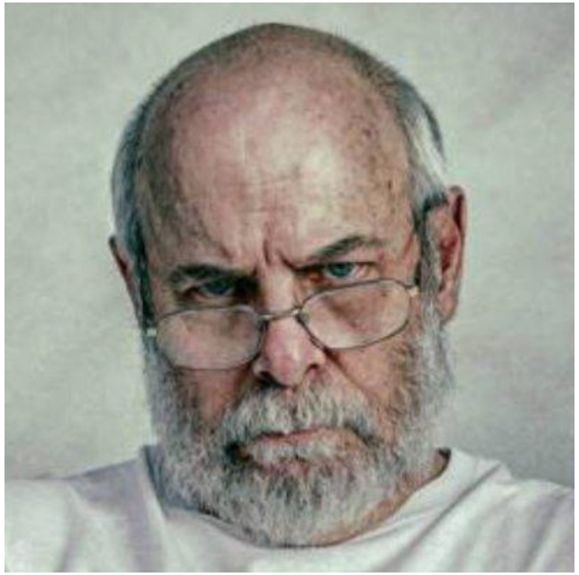
The Story So Far

Recap of last time

- Talk to me about types
- Pointers, references, others
- On dealing with errors
- Control flow in 15 seconds or less

Are we good to GO?

DID YOU SETUP YOUR TOOLING?
(...yes you did...)

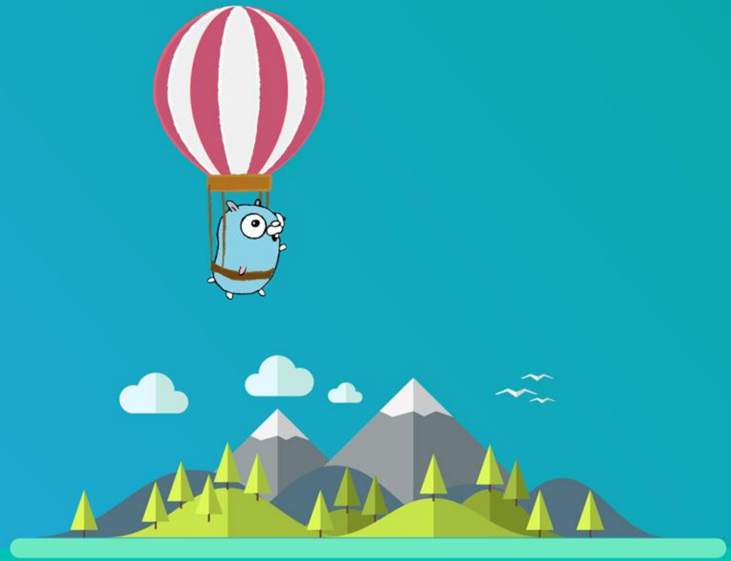


02.

Concurrency

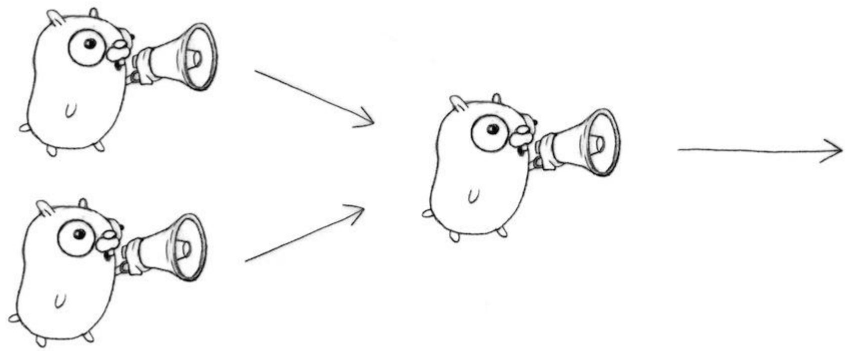
“

….let’s define what concurrency is….



-- Us, right now ”

Brief History of Concurrency



- Mutex - Communicate by sharing
- Actor model - Addressable units: Messaging and encapsulation
  CSP - Anonymous units for computation, channels for information sharing.

The tools

+ Runtime + Scheduler
+ Goroutines
+ Channels
+ Packages - “sync”, “unsafe”, a couple in “x/*”

Implementation

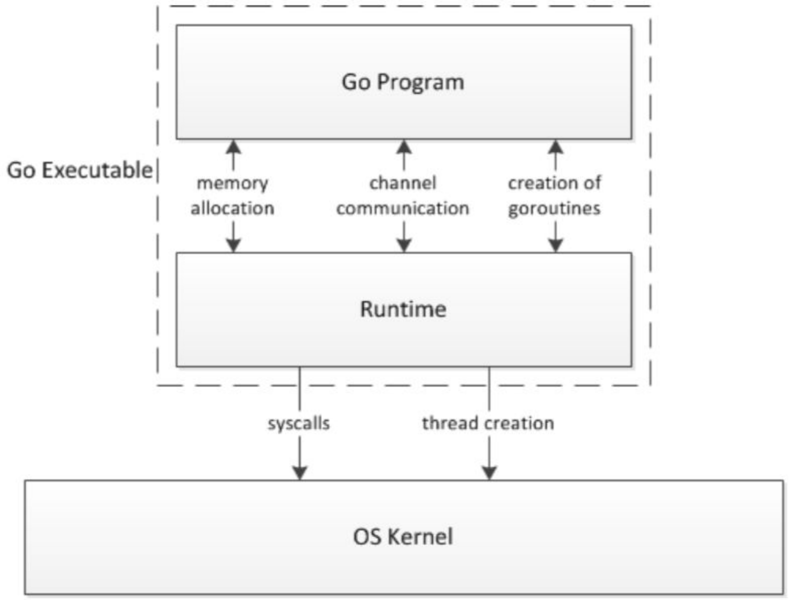


Figure 1: Diagram of the relationships between the runtime, OS, and programmer defined code

Types, we got ‘em

```
// The keyword go

go func (){
// does the return value matter?
}()


// Channels
intCh := make(chan int)
```

Jump into code

03.

Testing

Built in, but...

- Package “testing”
- go test -cover -v -race ./...
- Multiple enhancing libraries
    - https://github.com/stretchr/testify
    - https://github.com/onsi/ginkgo
    - https://github.com/franela/goblin
    - https://github.com/golang/mock
- Let’s look at tests in a real project

04.

Standard library

Comprehensive, not exhaustive

- “net” is civilization
- “io” / “buffio” make IO easy
- “database/sql” facilitates third party implementations
- https://pkg.go.dev/std

05.

A look inside a project

The tools we didn’t mention

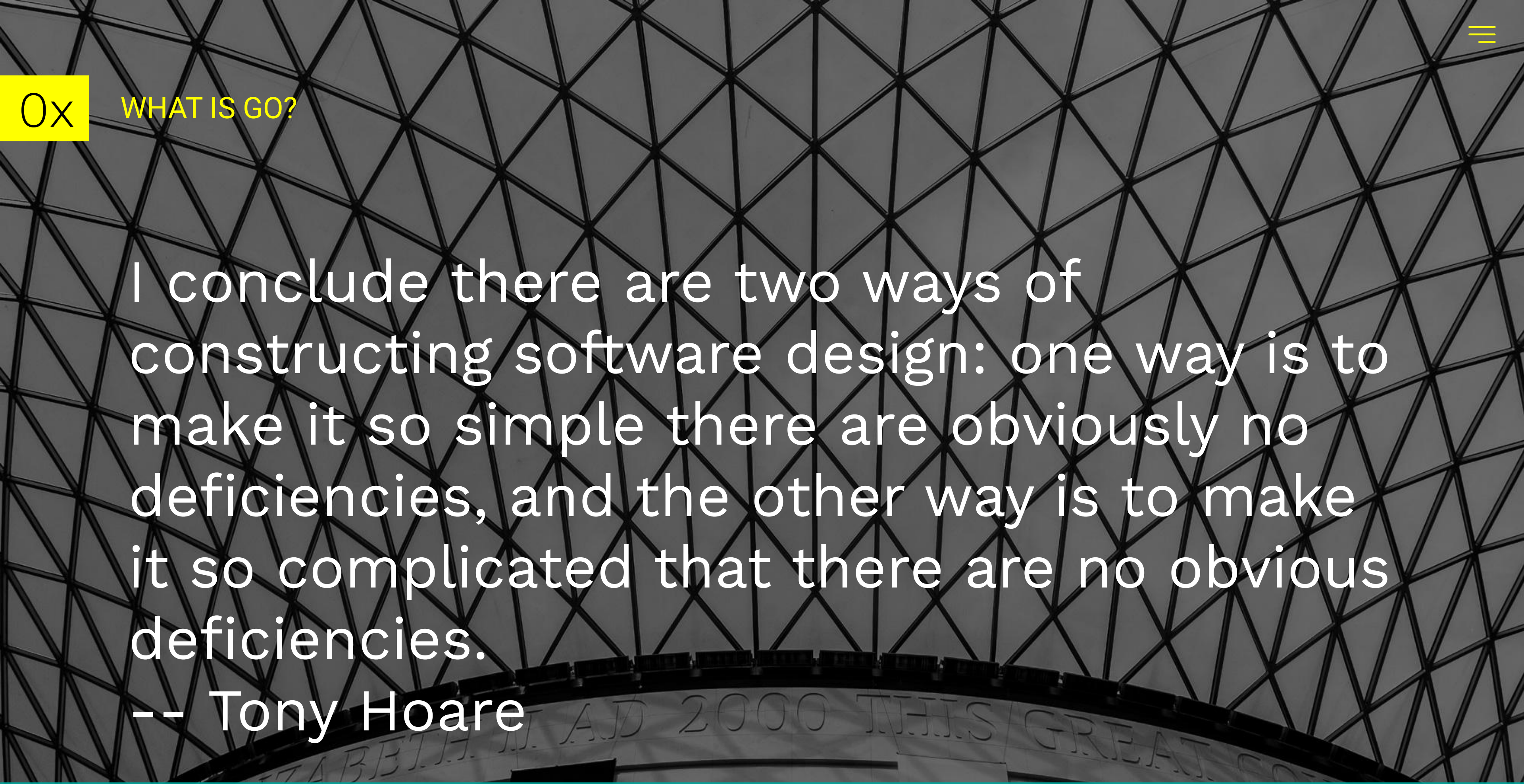
- Dependency management with go mod
  - Package proxy with Athens
- Project layout - “internal” and “pkg” directories
  - Flat popular, DD appropriate
- Shipping the application - single binary compile
  - Minimal containers

06.

Next time

You will be writing code with GRPC

+ Get familiar with the problem we are solving
+ Level up on RPC, interface definition languages and binary protocols
+ ...before you go: http://google.github.io/proto-lens/installing-protoc.html



I conclude there are two ways of constructing software design: one way is to make it so simple there are obviously no deficiencies, and the other way is to make it so complicated that there are no obvious deficiencies.
-- Tony Hoare