Golang Programming Workshop
Intermediate

Wojciech Barczynski

Contents

1 CLI app with cobra13

2 Generics

3 Golang concurrency

Golang concurrency:

- green threads (go routines)
- can run hundreds of thousands routines
- low overhead (dynamic stack)
- channels for communication
- scalable model
- you cannot control them
- see also Rob Pike's talk Concurrency is not parallelism[1]

Basic primitives:

- Spawning goroutine: `go myFunc()`
- Share memory by communicating with channels: `chan`
- Subcribe to multiple channels: `select`
- If you cannot use channel, you have also mutexes: `sync.Mutex` Coordinate multiple goroutines: `sync.WaitGroup`

4 Goroutines

The first meeting with Goroutines:

```
package main
import (
   "fmt"
  "time"
```

[1] blog.golang.org/concurrency-is-not-parallelism

```
)
func say(s string, num int) {
   for i := 0; i < num; i++ {
     time.Sleep(100 * time.Millisecond)
     fmt.Println(s)
   }
}

func main() {
   go say("world", 5)
   say("hello", 5)
}
```

- What is the output?
- What is the output when change 5 to 10 for saying world.

5 Simple producer and consumer