GO on the toilet

episode #9

Golang wants you to be safe

One of the most powerful features of go is it's concurrency model built around goroutines and channels. **Goroutine** is a lightweight thread managed by the Go runtime. **Channel** enables unidirectional communication between goroutines. If you want to know more about channels and goroutines, stay tuned - in upcoming episodes we will take a closer look at them.

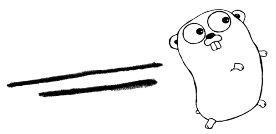
One of the problems with concurrent programming is a **data race**. Data race is a situation where a state of a variable is being changed concurrently, which effects in indefinite state of the variable. As said before, golang wants you to be safe. Consider running your code with **`-race`** flag. With this flag, golang builds a special instrumented binary, which warns you about every encountered data race event.

Let's have a look at an example, where a counter is increased 1000 times. Each write operation happens in a different goroutine.

```
func main() {
      counter := 0
      wg := sync.WaitGroup{}    // Used only to wait for all goroutines to finish
      iterations := 1000
      wg.Add(iterations)
      for i := 0; i < iterations; i++ {
            go func() {
                  counter++     // Concurrent write to the variable
                  wg.Done()
            }()
      }
      wg.Wait()
      fmt.Println(counter)
}
```

The result of running the example (using **`go run -race main.go`**) prints detailed report that states which goroutine, in which line of code causes data race. Another clue that the code doesn't behave in a deterministic fashion is that each execution provides a different result. It's worth to note that the production code we write is slightly more complex and spotting data races may be quite a challenge. It is a good practice to run your tests with **`-race`** flag to spot early some of non-obvious errors on CI pipeline.

In the next episode we will take a look at one of golangs proverb *"Do not communicate by sharing memory; instead, share memory by communicating.* Unfortunately here we've seen communicating by sharing memory, which we will fix next time.



Bathroom magazine with golang trivia, examples and patterns
Inspired by original Google's "Testing on the Toilet"
Browse previous episodes: https://github.com/jedruniu/goot