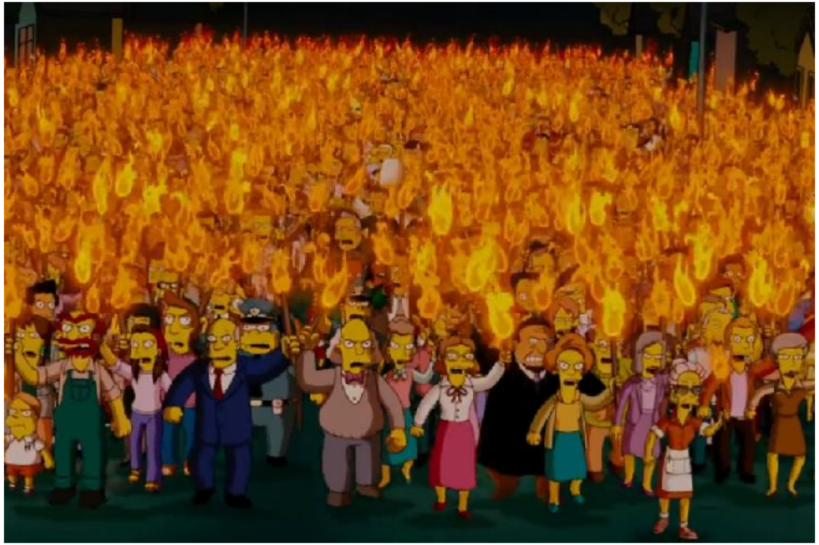
2022-05-12 @ Golang BG, Sofia

Testing:
A Look into Quick & Fuzz

Agenda

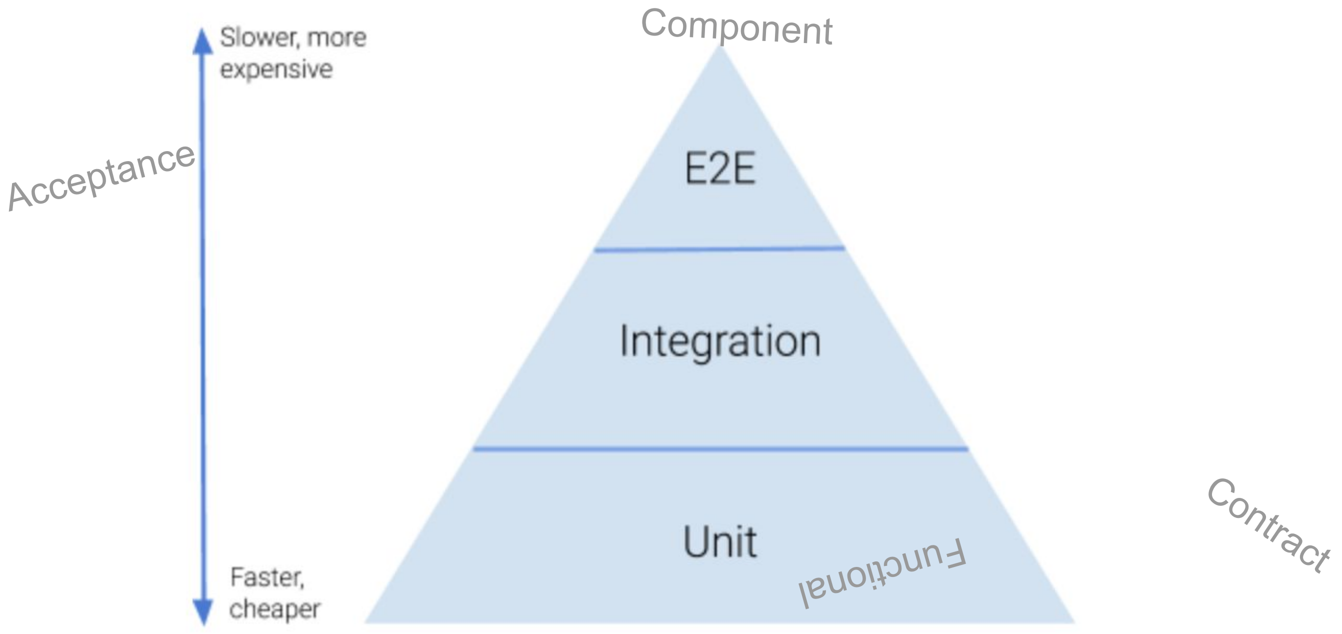
- A zoomed out view on testing
- Close up on unit
- The who & how of Generative.
- State of Golang + Code

Testing - 100% Not Controversial For Developers



– “TDD” preachers vs “Integration First” aficionados – Anno 2019, Pre Covid

The Testing ~~Ponzi S...~~ Pyramid

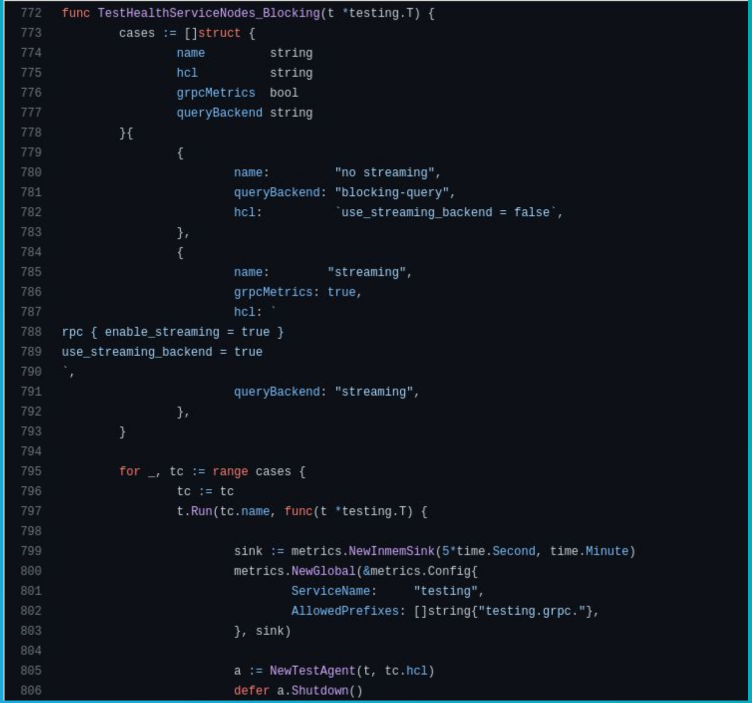


04. A Dive Into Unit-land

“In computer programming, **unit testing** is a software testing method by which individual units of source code—sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures—are tested to determine whether they are fit for use”

– Wikipedia, so can’t be wrong

05. The how of unit testing



```go
func TestHealthServiceNodes_Blocking(t *testing.T) {
	cases := []struct {
		name         string
		hcl          string
		grpcMetrics  bool
		queryBackend string
	}{
		{
			name:         "no streaming",
			queryBackend: "blocking-query",
			hcl:          `use_streaming_backend = false`,
		},
		{
			name:        "streaming",
			grpcMetrics: true,
			hcl: `
rpc { enable_streaming = true }
use_streaming_backend = true
`,
			queryBackend: "streaming",
		},
	}

	for _, tc := range cases {
		tc := tc
		t.Run(tc.name, func(t *testing.T) {

			sink := metrics.NewInmemSink(5*time.Second, time.Minute)
			metrics.NewGlobal(&metrics.Config{
				ServiceName:     "testing",
				AllowedPrefixes: []string{"testing.grpc."},
			}, sink)

			a := NewTestAgent(t, tc.hcl)
			defer a.Shutdown()
```

- Decide on the right unit
- Follow an input strategy
    - Hardcoded/Manual fixtures
    - Factories
    - Automated mocks
- Cover the happy/unhappy paths

...and then?

- Adjust the bug.
- Write the unit test
- Ship it....again

07. Property Based Testing

- Usually about ***correctness***
- Get the generators - describe input(and sometimes output)
- Detect a bug, maybe shrink the description.

Haskell: QuickCheck
Scala: ScalaCheck
Clojure: test.check
...aaand Go: testing/quick and Gopter

08. Fuzz

- Usually about ***stability and scale***
- Decide on the boundaries for fuzzing
- Get the fuzzers
  - Generate
  - Shrink
- Often a meta test

Golang: fuzz

09. Code code code

Open The Editor, buddy!

....any....any questions?