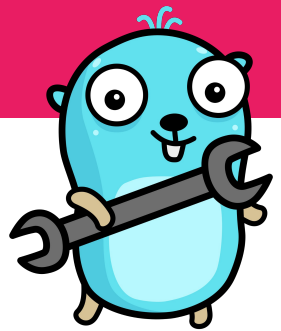
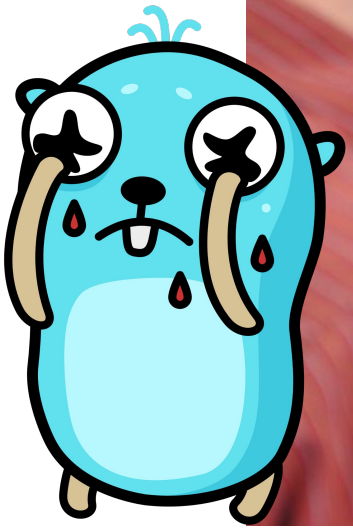
Testing Go CLI Applications with Test Script



Jakub Jarosz

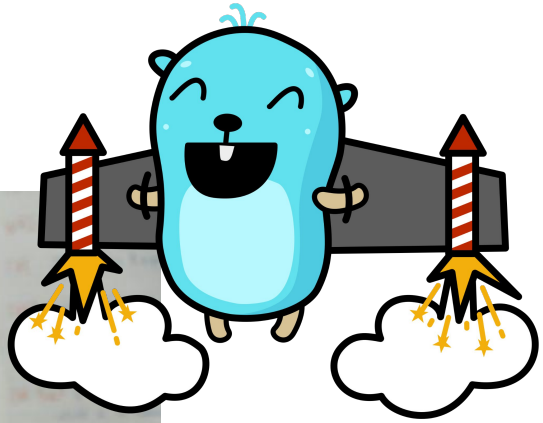
Belfast Gophers Meetup - February 2023

cli applications



http://blog.chriss-baumann.de/2010/07/26/because-life-is-too-short-for-manual-testing

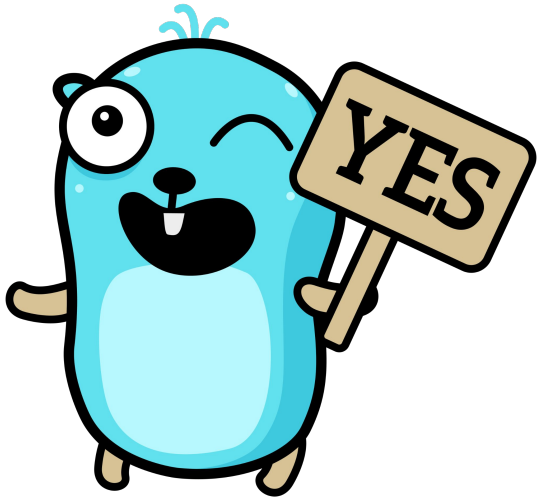
cli applications & testscript



http://blog.chriss-baumann.de/2010/07/26/because-life-is-too-short-for-manual-testing

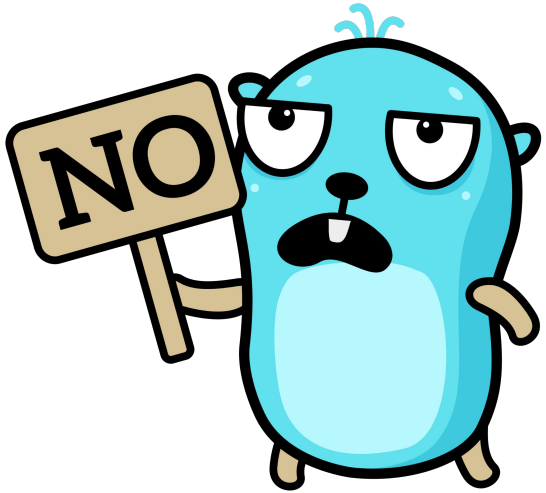
testscript - run commands

```
---
exec echo 'Hello Gophers!'
stdout 'Hello Gophers\n'
```



testscript - run commands

```
! exec cat not_existing_file.txt
stderr 'cat: not_existing_file.txt: No such file or directory'
```



testscript - run binaries

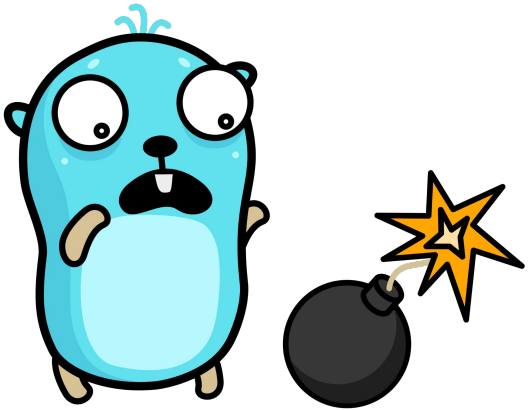
```
exec hello
stdout 'Hello Belfast Gophers\n'
```

testscript - run binaries with args

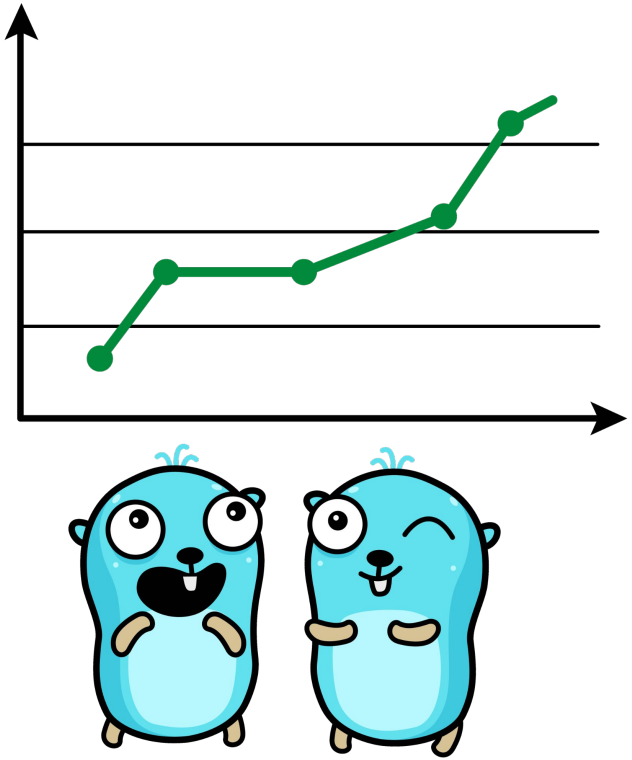
```
exec hello Shawn
stdout 'Hello Gopher, Shawn'
! stderr .
```

testscript - run binaries with args

```
! exec hello
! stdout .
stderr 'usage: Hello Gopher NAME'
```



testscript - code coverage

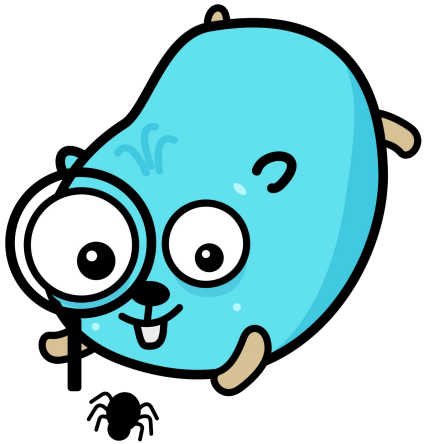


testscript - cmp & golden files

```
exec hello Kate
cmp stdout golden.txt

-- golden.txt --
Hello Gopher, Kate!
```

testscript - exists, grep & regex

```
exec hello Michael -o greet.txt
exists greet.out
grep 'Michael' greet.out
```



testscript - txtar files & dir structure

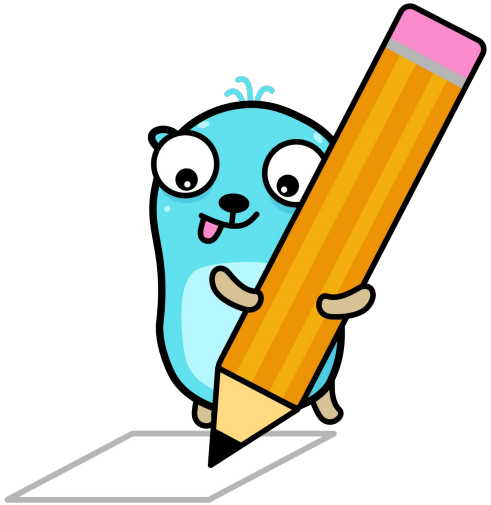
```
---
-- misc/file1.txt --
...

-- misc/subfolder/file2.txt --
...


-- etc/file3.txt --
...
```

testscript - file operations

```
cp

rm

mv

mkdir
…
```

testscript - comments & test phases

```
---
# Test 1 - run hello without argos
...

# Test 2 - run hello with args
...

# Test 3 - run hello with invalid args

...
```

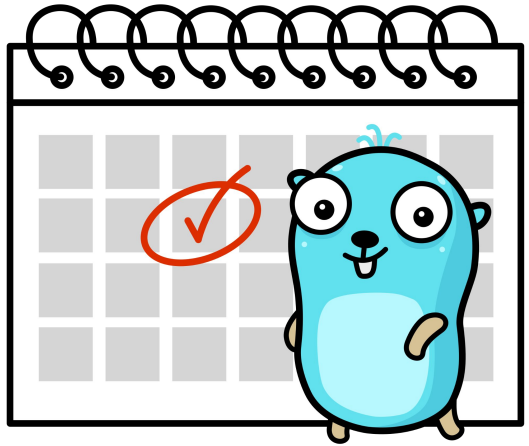


testscript - env vars

```
env API_KEY=
! exec myprog
stderr 'API_KEY must be set'
```

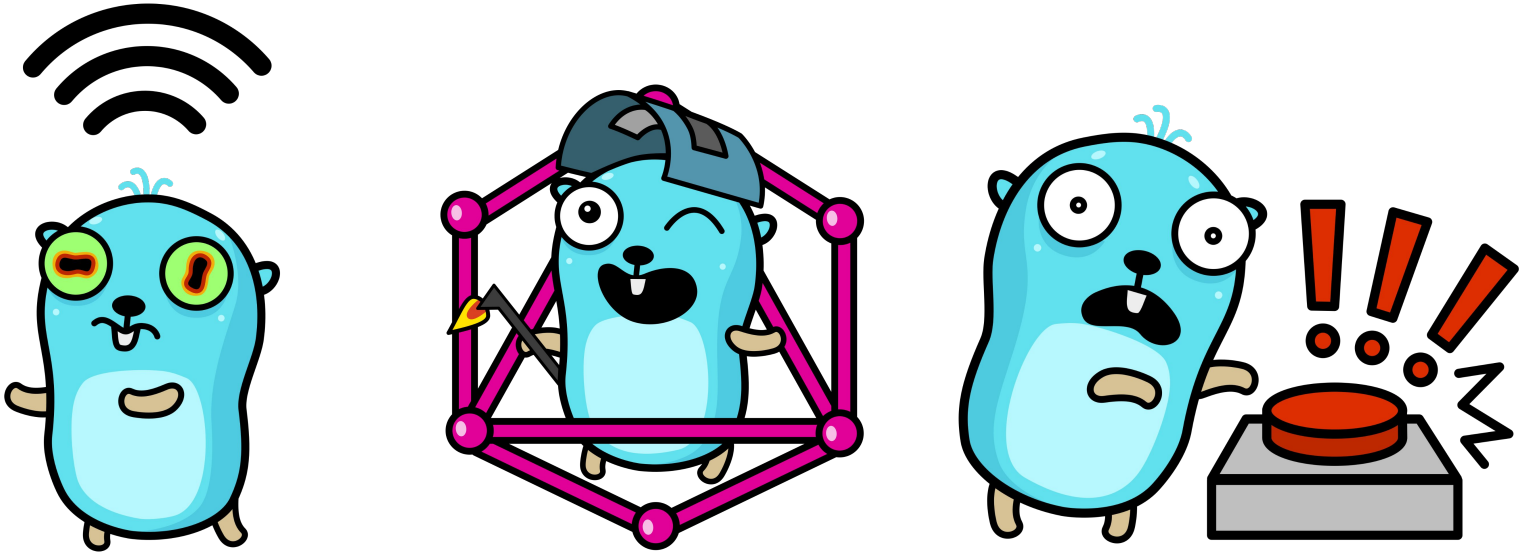
testscript - conditions

```
[go1.16] exec echo 'We have at least Go 1.16'

[!arm64] exec echo 'This is a non-arm64 machine'

[darwin] exec echo 'We''re on macOS'
```



testscript - extensions (custom functions)

testscript - background test steps

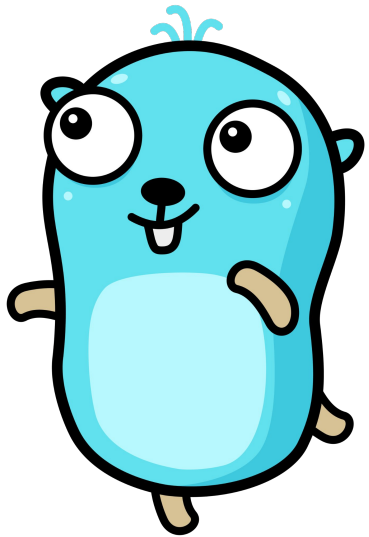


testscript - example project structure

```
.
└── hello
    ├── go.mod
    ├── go.sum
    ├── hello.go
    ├── hello_test.go
    └── testdata
        └── script
            └── hello.txtar
```

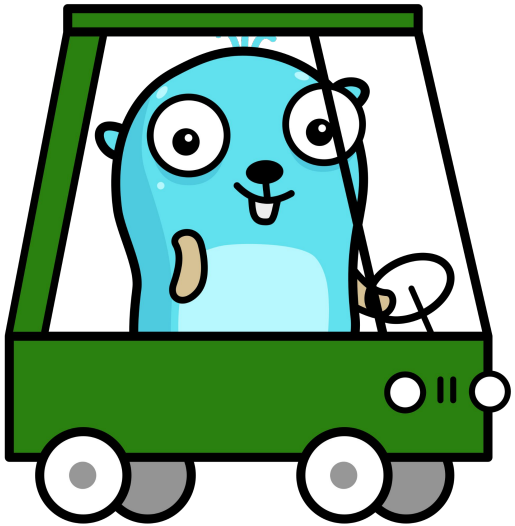
testscript.Run

```
func TestScript(t *testing.T) {
    testscript.Run(t, testscript.Params{
        Dir: "testdata/script",
    })
}
```



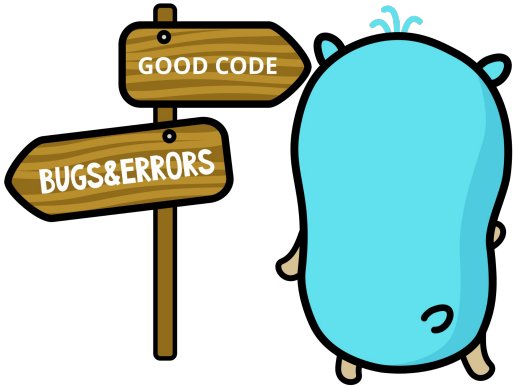
testscript.RunMain

```
func TestMain(m *testing.M) {
    os.Exit(testscript.RunMain(m, map[string]func() int{
        "hello": hello.RunCLI,
    }))
}
```



testscript - standalone testscript runner

```
testscript testdata/script/*
```

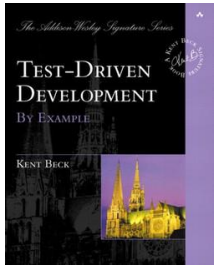


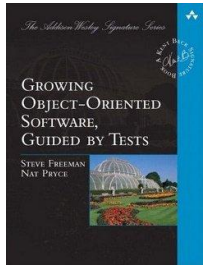
summary

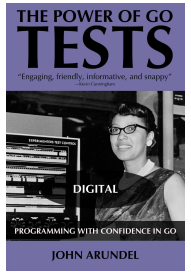
- Excellent for testing CLI tools and binaries
- Runs along other tests
- Saves time
- Builds confidence
- It’s derived directly from the code used to test Go tool itself!

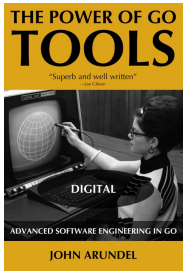
resources

- pkg.go.dev/github.com/rogpeppe/go-internal/testscript
- github.com/mvdan/go-internal
- bitfieldconsulting.com/books/tests
- github.com/MariaLetta/free-gophers-pack
- github.com/qba73/belfast-go-meetup
- github.com/qba73/habit









Thank You!

Jakub Jarosz

github, twitter @qba73 | linkedin.com/in/jakubjarosz