Занятие 8

Тестирование

Tinkoff.ru

Тестирование

Это **процесс** проверки ПО на соответствие между реальным и ожидаемым поведением

Зачем?

- Описание ожиданий (TDD)
- Проверка соответствия ожиданиям
- Проверка регрессии

План

- Подходы
- Автоматизация

Подходы

Какие тесты бывают?

- Функциональные
  - **Модульные (unit)**
  - **Интеграционные**
  - Приемочные и UI
- Нефункциональные
  - **Производительности**
  - Надежности (отказоустойчивости)
  - Удобство пользования, ...
- Связанные с изменениями
  - Регрессионные

- **Автоматические**
- Ручные

Функциональные тесты

Модульные тесты

Для тестирования отдельных “модулей” кода: **отдельных функций** и их композиции

```go
func Int2Str(val int) string {
	return fmt.Sprint(val)
}

func TestInt2Str() {
	if got := Int2Str(7); got != "7" {
		// AAAaaa!!!
	}
}
```

Модульные тесты

Для тестирования отдельных “модулей” кода: отдельных функций и **их композиции**

```
func Int2Str(val int) string {
	return fmt.Sprint(val)
}

func Str2Int(val string) (res int) {
	_, _ = fmt.Sscan(val, &res)
	return
}

func TestInt2StrAndStr2Int() {
	const in = 7
	if got := Str2Int(Int2Str(in)); in != got {
		// AAAaaa!!!
	}
}
```

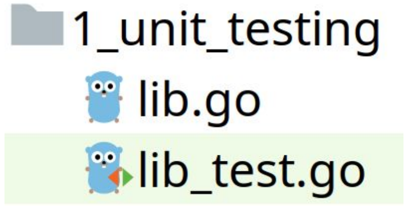
Модульные тесты - “из коробки”

lib.go:

```go
	package lecture07
	import "fmt"
	func Int2Str(val int) string {
		return fmt.Sprint(val)
	}
```

lib_test.go:

```go
	package lecture07
	import "testing"
	func TestInt2Str(t *testing.T) {
		const expect = "7"
		if got := Int2Str(7); got != expect {
			t.Errorf(`Expect %v got %v`, expect, got)
		}
	}
```



```
1_unit_testing
    lib.go
    lib_test.go
```

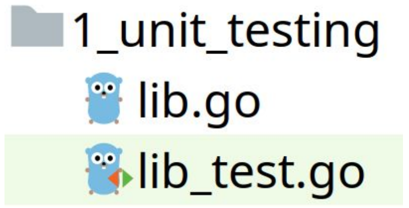
Модульные тесты - “из коробки”

lib.go:

```go
	package lecture07
	import "fmt"
	func Int2Str(val int) string {
		return fmt.Sprint(val)
	}
```

lib_test.go:

```go
	package lecture07
	import "testing"
	func TestInt2Str(t *testing.T) {
		const expect = "7"
		if got := Int2Str(7); got != expect {
			t.Errorf(`Expect %v got %v`, expect, got)
		}
	}
```



```
1_unit_testing
    lib.go
    lib_test.go
```

- Общий* пакет
- *_test.go в имени файла
- Test* в именах функций
- import "testing"
- *testing.T в сигнатуре функций

Модульные тесты - “из коробки”

```
$ ll
lib.go
lib_test.go

$ go test
PASS
ok      github.com/tfs-go/lections21/lecture07/code/1_unit_testing      0.001s



$ go test -v
=== RUN   TestInt2Str
--- PASS: TestInt2Str (0.00s)
PASS
ok      github.com/tfs-go/lections21/lecture07/code/1_unit_testing      0.001s
```

Модульные тесты - “из коробки”

```
$ cd $GOPATH/src/github.com/tfs-go/lections21
$ go test
no Go files in $GOPATH/src/github.com/tfs-go/lections21
```

Модульные тесты - “из коробки”

```
$ cd $GOPATH/src/github.com/tfs-go/lections21
$ go test
no Go files in $GOPATH/src/github.com/tfs-go/lections21

$ go test ./...
...
ok      github.com/tfs-go/lections21/lecture07/code/1_unit_testing      0.001s
```

Модульные тесты - “из коробки”

```
$ cd $GOPATH/src/github.com/tfs-go/lections21
$ go test
no Go files in $GOPATH/src/github.com/tfs-go/lections21

$ go test ./...
...
ok      github.com/tfs-go/lections21/lecture07/code/1_unit_testing      0.001s


$ go help test
```

Модульные тесты - “из коробки”

```go
func TestInt2Str(t *testing.T) {
	const expect = "7"
	if got := Int2Str(7); got != expect {
		t.Errorf(`Expect %v got %v`, expect, got)
	}
}
```

Модульные тесты - “из коробки”

```go
func TestInt2Str(t *testing.T) {
	const expect = "100500"
	if got := Int2Str(7); got != expect {
		t.Errorf(`Expect %v got %v`, expect, got)
	}
}
```

```
$ go test
--- FAIL: TestInt2Str (0.00s)
   lib_test.go:9: Expect 100500 got 7
FAIL
exit status 1
FAIL	github.com/tfs-go/lections21/lecture07/code/1_unit_testing	0.001s
```

Модульные тесты - “из коробки”

```go
func TestInt2Str(t *testing.T) {
	const expect = "100500"
	if got := Int2Str(7); got != expect {
		t.Errorf(`Expect %v got %v`, expect, got)
	}
}
```

Модульные тесты - “из коробки”

```go
func TestInt2Str(t *testing.T) {
	if expect, got := "100500", Int2Str(7); got != expect {
		t.Errorf(`Expect %v got %v`, expect, got)
	}
	if expect, got := "100500", Int2Str(9); got != expect {
		t.Errorf(`Expect %v got %v AGAIN`, expect, got)
	}
}
```

Модульные тесты - “из коробки”

```
func TestInt2Str(t *testing.T) {
	if expect, got := "100500", Int2Str(7); got != expect {
		t.Errorf(`Expect %v got %v`, expect, got)
	}
	if expect, got := "100500", Int2Str(9); got != expect {
		t.Errorf(`Expect %v got %v AGAIN`, expect, got)
	}
}
```

```
$ go test
--- FAIL: TestInt2StrFailed (0.00s)
    lib_test.go:15: Expect 100500 got 7
    lib_test.go:18: Expect 100500 got 9 AGAIN
FAIL
exit status 1
FAIL    github.com/tfs-go/lections21/lecture07/code/1_unit_testing      0.001s
```

Модульные тесты - “из коробки”

Методы	**Что происходит, кроме вывода сообщения**
Log	Вывести сообщение, только если тест упал или с *-v*
Error	Отметить тест упавшим, но продолжить его
Fatal	Отметить упавшим и прервать его
Skip	Отметить пропущенным и прервать его
panic()	Отметить упавшим, вывести стек

Модульные тесты - “из коробки”

```go
func TestParallel_1(t *testing.T) {
	t.Parallel()
	t.Log(`parallel 1:`, t.TempDir())
}

func TestParallel_2(t *testing.T) {
	t.Parallel()
	t.Log(`parallel 2:`, t.TempDir())
}

func TestSubtests(t *testing.T) {
	t.Run(`sub1`, TestParallel_1)
	t.Run(`sub2`, TestParallel_2)
}
```

Модульные тесты - табличные тесты

```go
if expect, got := "7", Int2Str(7); got != expect {
      t.Errorf(`Expect %v got %v`, expect, got)
}
if expect, got := "0", Int2Str(0); got != expect {
      t.Errorf(`Expect %v got %v`, expect, got)
}
// ...
```

Модульные тесты - табличные тесты

```go
type Test struct {
      In     int
      Expect string
}
tests := [...]Test{
      {7, "7"},
      {0, "0"},
      // ...
}

for idx, test := range tests {
      got := Int2Str(test.In)
      if got != test.Expect {
            t.Fatalf(`test%d: expect %v got %v`, idx, test.Expect, got)
      }
}
```

Модульные тесты - табличные тесты

```go
type Test struct {
	Name   string
	In     int
	Expect string
}
tests := [...]Test{
	{"Non zero",  7, "7"},
	{"Zero",      0, "0"},
	{"Negative", -1, "1"}, // bug!
}

for _, test := range tests {
	got := Int2Str(test.In)
	if got != test.Expect {
		t.Fatalf(`test %q: expect %v got %v`, test.Name, test.Expect, got)
	}
}
```

Модульные тесты - табличные тесты

```go
import (
	"reflect"
	"testing"
)

a := map[int]int{1: 2, 4: 2}
b := map[int]int{4: 2, 1: 2}
c := map[int]int{4: 2, 1: 4}

if !reflect.DeepEqual(a, b) {
	t.Fatal("a is not equal to b")
}

if reflect.DeepEqual(a, c) {
	t.Fatal("a is equal to c")
}
```

Модульные тесты - setup & teardown

```go
func TestMain(m *testing.M) {
      fmt.Println("Before all tests")
      code := m.Run()
      fmt.Println("After all tests")
      os.Exit(code)
}
```

Модульные тесты - setup & teardown

```
teardown := func() {
	fmt.Println("After test")
}

setup := func(t *testing.T) {
	t.Cleanup(teardown)
	fmt.Println("Before test")
}

t.Run("with Cleanup", func(t *testing.T) {
	setup(t)
	panic("Ooops! I did it again!")
})
```

Модульные тесты

https://pkg.go.dev/testing

Картинка из https://www.amazon.com/Golang-Gopher-Go-Panic-Sticker/dp/B08128TKST

Модульные тесты - testify

```go
import (
	"math/rand"
	"testing"

	"github.com/stretchr/testify/assert"
)

func TestInt2Str_Testify(t *testing.T) {
	assert.Equal(t, "7", Int2Str(7))

	assert.Equal(t, "10", Int2Str(0), "zero value")

	assert.ElementsMatch(t, []int{1, 2, 3}, []int{2, 3, 1})

	assert.InDelta(t, 7, 5+rand.Intn(4), 3)
}
```

~ 140 методов

Модульные тесты - rapid

```go
type Test struct {
      In     int
      Expect string
}
tests := [...]Test{
      {7, "7"},
      {0, "0"},
      // ...
}

for idx, test := range tests {
      got := Int2Str(test.In)
      if got != test.Expect {
            t.Fatalf(`test%d: expect %v got %v`, idx, test.Expect, got)
      }
}
```

Модульные тесты - rapid

```go
func Int2StrWrong(val int) string {
  if val == -1 || val == math.MaxInt16 {
     return "0"
  }
  return fmt.Sprint(val)
}
```

Модульные тесты - rapid

```go
import "pgregory.net/rapid"

func TestInt2StrWrong_Rapid(t *testing.T) {
  rapid.Check(t, func(t *rapid.T) {
     val := rapid.Int32().Draw(t, "val").(int32)

     got := Int2StrWrong(int(val))
     expect := fmt.Sprint(val)

     if got != expect {
        t.Fatalf("expect %v got %v", expect, got)
     }
  })
}
```

Модульные тесты - rapid

```
$ go test
--- FAIL: TestInt2StrWrong_Rapid (0.00s)
   lib_rapid_test.go:11: [rapid] failed after 1 tests: expect -1 got 0
       To reproduce, specify -run="TestInt2StrWrong_Rapid"
-rapid.failfile="TestInt2StrWrong_Rapid-20211030221149-46991.fail" (or -rapid.seed=13840173142288367618)
       Failed test output:
   lib_rapid_test.go:12: [rapid] draw val: -1
   lib_rapid_test.go:18: expect -1 got 0
FAIL
```

Модульные тесты - go-fuzz

Что это вообще?

https://www.youtube.com/watch?v=EJVp13f_aIs

Встроенная поддержка в Go 1.18

https://github.com/golang/go/issues/44551

Интеграционные тесты

Интеграционные тесты

Для тестирования взаимодействия модулей и сервисов

lib.go:

```
func HttpReq(addr string) (string, error) {
	resp, err := http.DefaultClient.Get(addr)
	if err != nil {
		return "", err
	}
	defer resp.Body.Close()

	body, err := ioutil.ReadAll(resp.Body)
	if err != nil {
		return "", err
	}
	return string(body), nil
}
```

Интеграционные тесты - hold my beer

lib_test.go:

```
type server struct{}

func (s *server) ServeHTTP(resp http.ResponseWriter, req *http.Request) {
	fmt.Printf("HTTP handler: %q\n", req.RequestURI)
	_, _ = resp.Write([]byte(req.RequestURI))
}
```

Интеграционные тесты - hold my beer

lib_test.go:

```
	func setup(ipAddr string, t *testing.T) (int, func() error) {
		ipAddr += ":0"
		server := &http.Server{Addr: ipAddr, Handler: &server{}}

		ln, err := net.Listen("tcp", ipAddr)
		if err != nil {
			t.Fatalf("Could not listen port: %s", err)
		}
		go server.Serve(ln)

		port := ln.Addr().(*net.TCPAddr).Port

		return port, server.Close
	}
```

Интеграционные тесты - hold my beer

lib_test.go:

```go
	func TestHttpReq(t *testing.T) {
		const ipAddr = "127.0.0.1"

		port, closer := setup(ipAddr, t)
		defer closer()

		addrWithPort := net.JoinHostPort(ipAddr, strconv.Itoa(port))

		const expect = "/hello_world"
		got, _ := HttpReq("http://" + addrWithPort + expect)
		if got != expect {
			t.Fatalf("Expect %v got %v", expect, got)
		}
	}
```

Интеграционные тесты

Автор рисунка https://github.com/quasilyte

Интеграционные тесты

lib_test.go:

```
func TestHttpReq(t *testing.T) {
  server := httptest.NewServer(http.HandlerFunc(func(resp http.ResponseWriter, req *http.Request) {
     fmt.Printf("HTTP handler: %q\n", req.RequestURI)
     _, _ = resp.Write([]byte(req.RequestURI))
  }))
  defer func() { server.Close() }()

  const expect = "/hello_world"

  got, err := HttpReq(server.URL + expect)
  assert.NoError(t, err)

  assert.Equal(t, expect, got)
}
```

Покрытие тестами

Покрытие тестами

```
$ go test ./...
...

$ go test -cover ./...
ok      .../1_unit_testing          0.006s  coverage: 50.0% of statements
ok      .../2_integration_testing   0.003s  coverage: 75.0% of statements
ok      .../3_benchmark_testing     0.001s  coverage:  0.0% of statements [no tests to run]
```

Покрытие тестами

```go
func Int2StrWrong(val int) string {

  if val == -1 || val == math.MaxInt16 {

    return `0`
  }

  return fmt.Sprint(val)
}
```

$\longrightarrow$

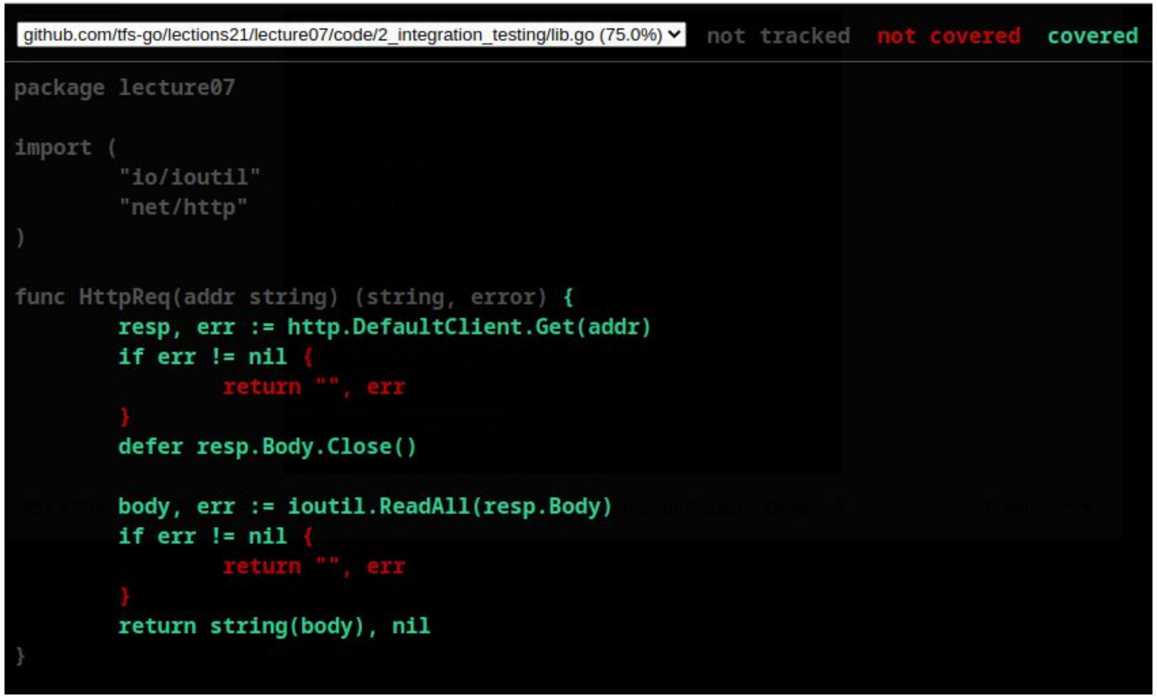
```go
func Int2StrWrong(val int) string {
  GoCover.Count[1] = 1
  if val == -1 || val == math.MaxInt16 {
    GoCover.Count[2] = 1
    return `0`
  }
  GoCover.Count[3] = 1
  return fmt.Sprint(val)
}
```

Покрытие тестами

```
$ go test -cover -coverprofile=coverage.out ./... && go tool cover -func=coverage.out
ok      .../1_unit_testing      0.010s  coverage: 33.3% of statements
ok      .../2_integration_testing       0.003s  coverage: 75.0% of statements
ok      .../3_benchmark_testing 0.001s  coverage: 0.0% of statements [no tests to run]

.../1_unit_testing/lib.go:8:           Int2Str         100.0%
.../1_unit_testing/lib.go:16:          Int2StrWrong    0.0%
.../1_unit_testing/lib.go:26:          Str2Int         100.0%
.../2_integration_testing/lib.go:8:    HttpReq         75.0%
.../3_benchmark_testing/lib.go:8:      Int2Str         0.0%
.../3_benchmark_testing/lib.go:12:     Int2StrFast     0.0%
.../3_benchmark_testing/lib.go:16:     Int2ByteSlice   0.0%
total:                                 (statements)    45.0%
```

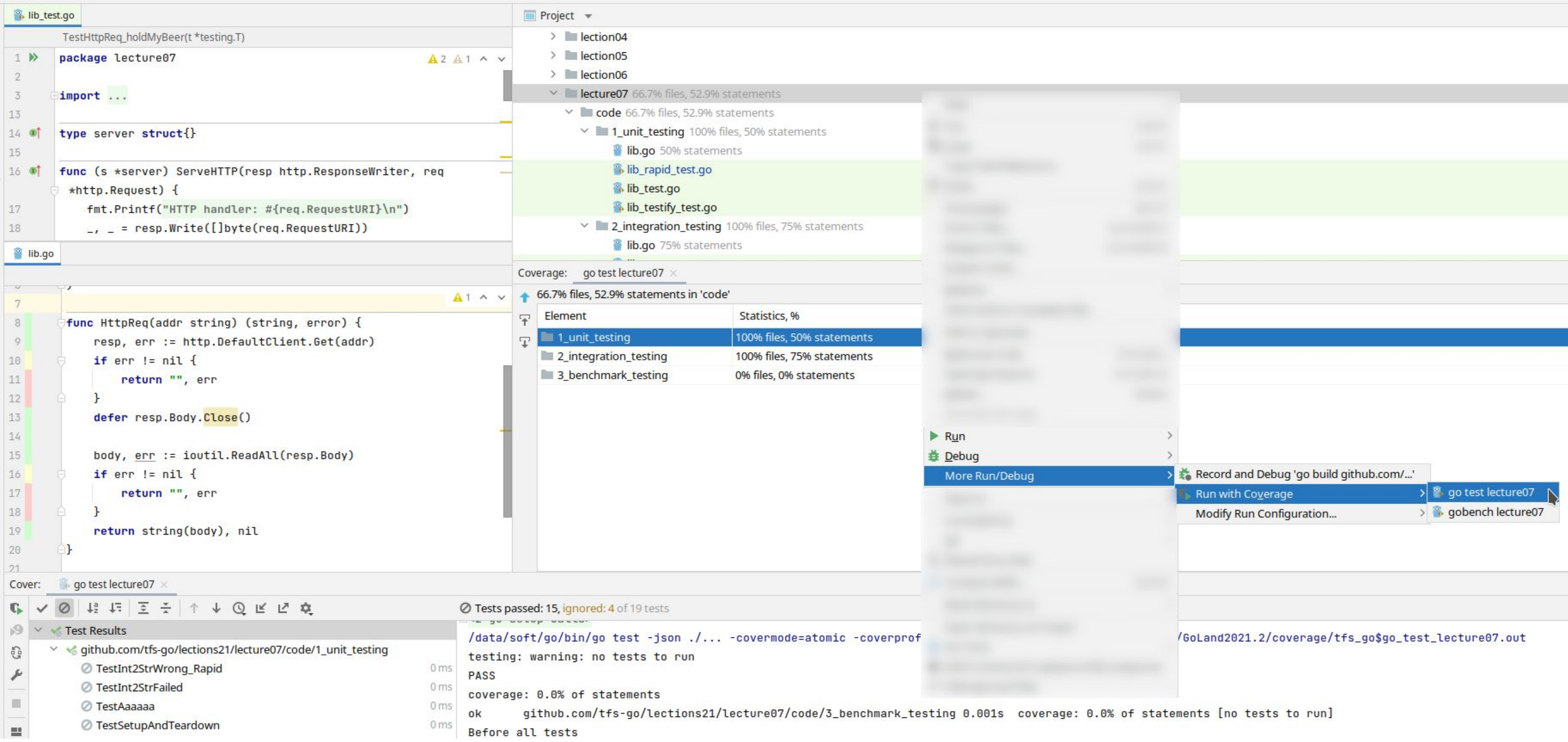
Покрытие тестами

```
$ go test -cover -coverprofile=coverage.out ./... && go tool cover -html=coverage.out
```



```
github.com/tfs-go/lections21/lecture07/code/2_integration_testing/lib.go (75.0%)   not tracked   not covered   covered

package lecture07

import (
	"io/ioutil"
	"net/http"
)

func HttpReq(addr string) (string, error) {
	resp, err := http.DefaultClient.Get(addr)
	if err != nil {
		return "", err
	}
	defer resp.Body.Close()

	body, err := ioutil.ReadAll(resp.Body)
	if err != nil {
		return "", err
	}
	return string(body), nil
}
```

Покрытие тестами



lib_test.go

TestHttpReq_holdMyBeer(t *testing.T)

```go
package lecture07

import ...

type server struct{}

func (s *server) ServeHTTP(resp http.ResponseWriter, req
 *http.Request) {
    fmt.Printf("HTTP handler: #{req.RequestURI}\n")
    _, _ = resp.Write([]byte(req.RequestURI))
```

lib.go

```go
func HttpReq(addr string) (string, error) {
    resp, err := http.DefaultClient.Get(addr)
    if err != nil {
        return "", err
    }
    defer resp.Body.Close()

    body, err := ioutil.ReadAll(resp.Body)
    if err != nil {
        return "", err
    }
    return string(body), nil
}
```

Project

- lection04
- lection05
- lection06
- lecture07 66.7% files, 52.9% statements
  - code 66.7% files, 52.9% statements
    - 1_unit_testing 100% files, 50% statements
      - lib.go 50% statements
      - lib_rapid_test.go
      - lib_test.go
      - lib_testify_test.go
    - 2_integration_testing 100% files, 75% statements
      - lib.go 75% statements

Coverage: go test lecture07

66.7% files, 52.9% statements in 'code'

Element	Statistics, %
1_unit_testing	100% files, 50% statements
2_integration_testing	100% files, 75% statements
3_benchmark_testing	0% files, 0% statements

Cover: go test lecture07

Tests passed: 15, ignored: 4 of 19 tests

- Test Results
  - github.com/tfs-go/lections21/lecture07/code/1_unit_testing
    - TestInt2StrWrong_Rapid 0 ms
    - TestInt2StrFailed 0 ms
    - TestAaaaaa 0 ms
    - TestSetupAndTeardown 0 ms

```
/data/soft/go/bin/go test -json ./... -covermode=atomic -coverprof                /GoLand2021.2/coverage/tfs_go$go_test_lecture07.out
testing: warning: no tests to run
PASS
coverage: 0.0% of statements
ok      github.com/tfs-go/lections21/lecture07/code/3_benchmark_testing 0.001s  coverage: 0.0% of statements [no tests to run]
Before all tests
```

Тесты

https://github.com/avelino/awesome-go#testing

Нефункциональные тесты

Тесты производительности

Тесты производительности

lib.go:

```
      func Int2Str(val int) string {
            return fmt.Sprint(val)
      }
      func Int2StrFast(val int) string {
        return strconv.Itoa(val)
      }
```

Тесты производительности

lib.go:

```go
func Int2Str(val int) string {
      return fmt.Sprint(val)
}
func Int2StrFast(val int) string {
  return strconv.Itoa(val)
}
```

lib_test.go:

```go
func BenchmarkInt2Str(b *testing.B) {
  for i := 0; i < b.N; i++ {
     _ = Int2Str(i)
  }
}
```

Отличия:

- Benchmark* в именах функций
- *testing.B в сигнатуре функций
- Нужно учитывать b.N

Тесты производительности

```
$ go test -bench . -cpu 1
...
BenchmarkInt2Str         12898684        94.39 ns/op
BenchmarkInt2StrFast     47510143        28.34 ns/op
...
```

Тесты производительности

```
$ go test -bench . -benchmem -cpu 1
...
BenchmarkInt2Str          12898684         94.39 ns/op          16 B/op       1 allocs/op
BenchmarkInt2StrFast      47510143         28.34 ns/op           7 B/op       0 allocs/op
...
```

Тесты производительности

lib.go:

```
      func Int2Str(val int) string {
            return fmt.Sprint(val)
      }
      func Int2StrFast(val int) string {
        return strconv.Itoa(val)
      }
      func Int2ByteSlice(val int, dst []byte) []byte {
        return strconv.AppendInt(dst, int64(val), 10)
      }
```

Тесты производительности

```
$ go test -bench . -benchmem -cpu 1
...
BenchmarkInt2Str          12898684        94.39 ns/op         16 B/op       1 allocs/op
BenchmarkInt2StrFast      47510143        28.34 ns/op          7 B/op       0 allocs/op
BenchmarkInt2ByteSlice    82708846        16.74 ns/op          0 B/op       0 allocs/op
...
```

Тесты производительности

```
$ GODEBUG=gctrace=1 go test -bench . -benchmem -cpu 1
gc 1 @0.013s 4%: 0.10+5.5+0.040 ms clock, 0.65+1.0/3.2/0+0.24 ms cpu, 4->5->1 MB, 5 MB goal, 6 P
...
```

https://www.ardanlabs.com/blog/2019/05/garbage-collection-in-go-part2-gctraces.html

Тесты производительности

https://dave.cheney.net/2013/06/30/how-to-write-benchmarks-in-go

Профилирование

Почитать и посмотреть

“Профилирование и оптимизация программ на Go”

https://habr.com/ru/company/badoo/blog/301990/

Почитать и посмотреть

Минутка саморекламы :)

“Работа с сетью в Go” GopherCon Russia 2018

https://youtu.be/p1ILhiq5Clw

“Опыт участника Highload Cup” Highload++ 2017

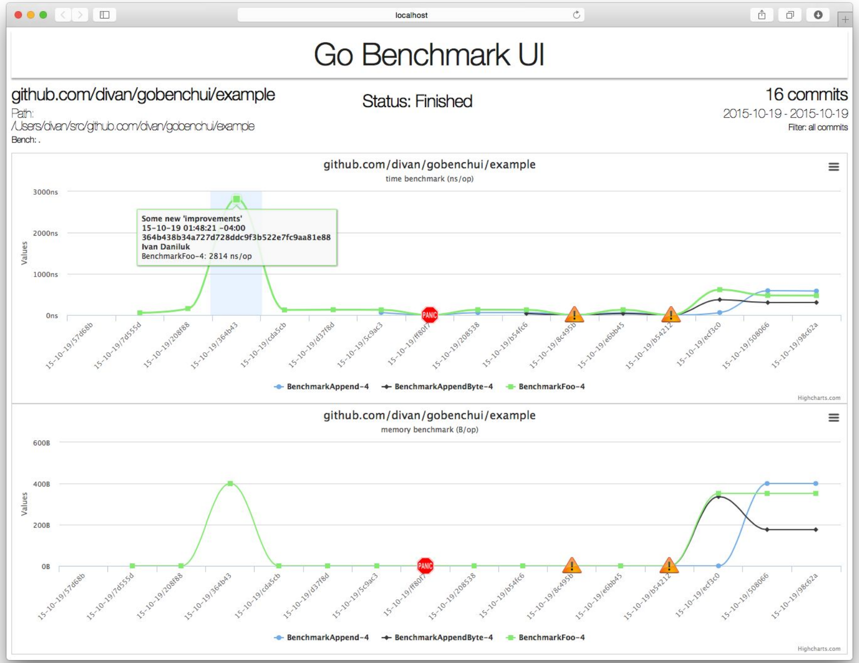
https://highloadcup.ru/ru/round/1/

https://youtu.be/WGYQus5J2Eo

Почитать и посмотреть

GoBenchUI (заброшено)

https://github.com/divan/gobenchui



localhost

Go Benchmark UI

github.com/divan/gobenchui/example

Path:
/Users/divan/src/github.com/divan/gobenchui/example

Bench: .

Status: Finished

16 commits

2015-10-19 - 2015-10-19

Filter: all commits

Some new 'improvements'
15-10-19 01:48:21 -04:00
364b438b34a727d728ddc9f3b522e7fc9aa81e88
Ivan Daniluk
BenchmarkFoo-4: 2814 ns/op

Автоматизация

Автоматизация

push кода в репозиторий ⇒ запуск автоматических проверок

- Статический анализ кода
- Тесты и покрытие
- Проверка сборки

Автоматизация

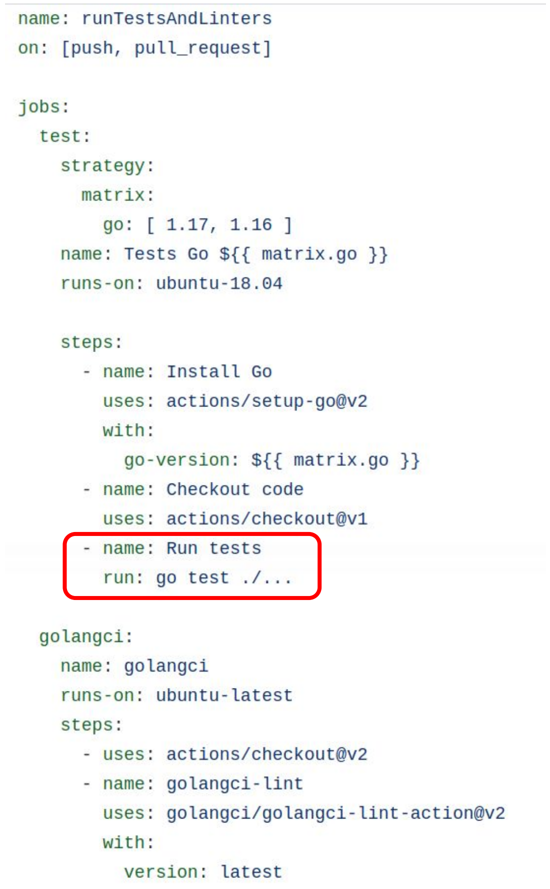
GitHub Actions

github.com/tfs-go/lections21/blob/main/.github/workflows/ci.yml

Автоматизация

GitHub Actions

github.com/tfs-go/lections21/blob/main/.github/workflows/ci.yml

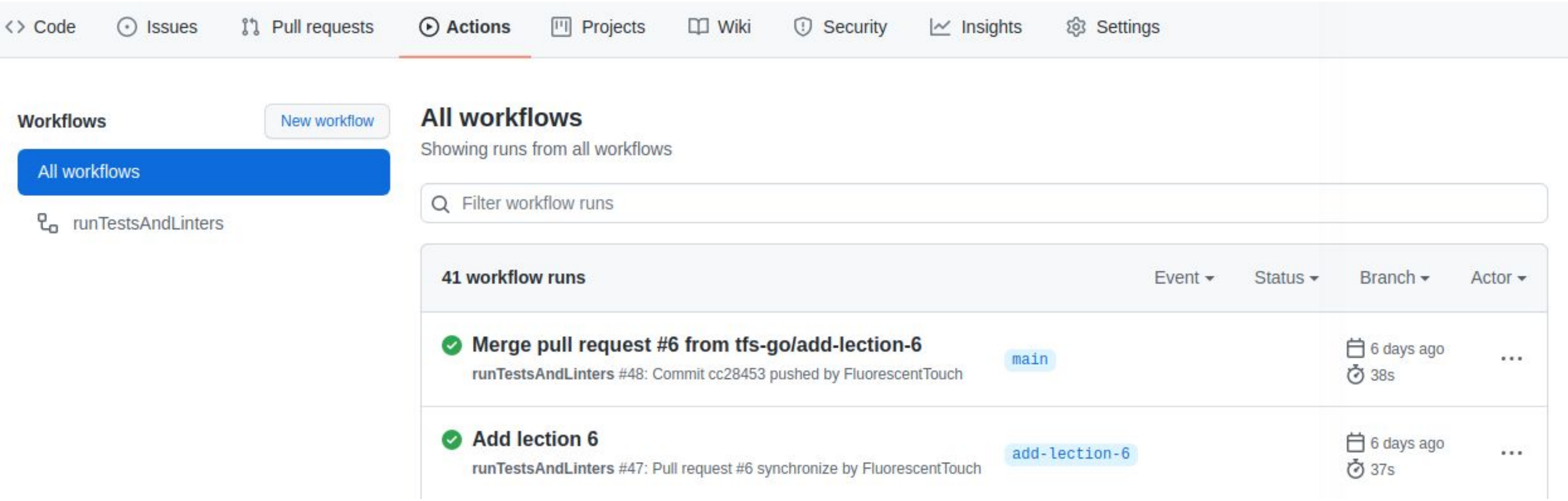


```yaml
name: runTestsAndLinters
on: [push, pull_request]

jobs:
  test:
    strategy:
      matrix:
        go: [ 1.17, 1.16 ]
    name: Tests Go ${{ matrix.go }}
    runs-on: ubuntu-18.04

    steps:
      - name: Install Go
        uses: actions/setup-go@v2
        with:
          go-version: ${{ matrix.go }}
      - name: Checkout code
        uses: actions/checkout@v1
      - name: Run tests
        run: go test ./...

  golangci:
    name: golangci
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - name: golangci-lint
        uses: golangci/golangci-lint-action@v2
        with:
          version: latest
```

Автоматизация

GitHub Actions

github.com/tfs-go/lections21/blob/main/.github/workflows/ci.yml



All workflows

Showing runs from all workflows

41 workflow runs

Merge pull request #6 from tfs-go/add-lection-6
runTestsAndLinters #48: Commit cc28453 pushed by FluorescentTouch
main
6 days ago
38s

Add lection 6
runTestsAndLinters #47: Pull request #6 synchronize by FluorescentTouch
add-lection-6
6 days ago
37s

https://docs.github.com/en/actions

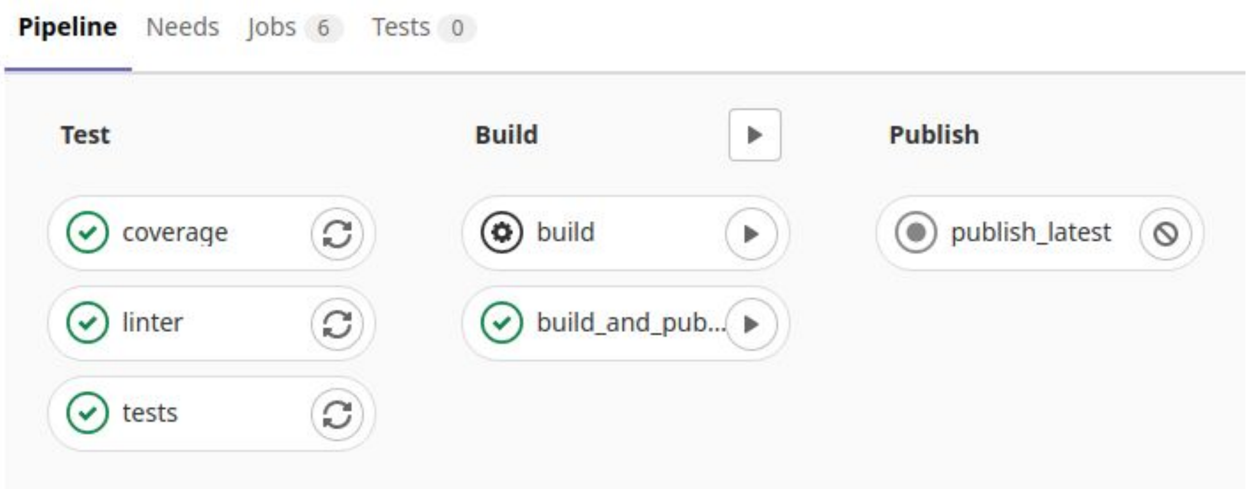
Автоматизация

GitLab CI / `.gitlab-ci.yml`

```yaml
stages:
  - test
  - build
  - publish

.tests:
  extends: .base
  image:
    name: $GOLANG_IMG
    entrypoint: [ "" ]
  script:
    - go test --race --vet= --count=1 ./... -v

.coverage:
  extends: .base
  image:
    name: $GOLANG_IMG
    entrypoint: [ "" ]
  variables:
    COVER_EXCLUDE: ""
  script:
    - file=coverage.count.out
    - go test --count=1 --covermode=count --coverprofile=$file --coverpkg=./... ./...
    - if [[ "$COVER_EXCLUDE" != "" ]]; then
        egrep -v "$COVER_EXCLUDE" $file > $file.tmp && mv $file.tmp $file ;
      fi
    - go tool cover --func=$file
```

Автоматизация

GitLab CI / `.gitlab-ci.yml`



Автоматизация

```
name: runTestsAndLinters
on: [push, pull_request]

jobs:
 test:
  strategy:
   matrix:
    go: [ 1.17, 1.16 ]
  name: Tests Go ${{ matrix.go }}
  runs-on: ubuntu-18.04

  steps:
   - name: Install Go
    uses: actions/setup-go@v2
    with:
     go-version: ${{ matrix.go }}
   - name: Checkout code
    uses: actions/checkout@v1
   - name: Run tests
    run: go test ./...

 golangci:
  name: golangci
  runs-on: ubuntu-latest
  steps:
   - uses: actions/checkout@v2
   - name: golangci-lint
    uses: golangci/golangci-lint-action@v2
    with:
     version: latest
```

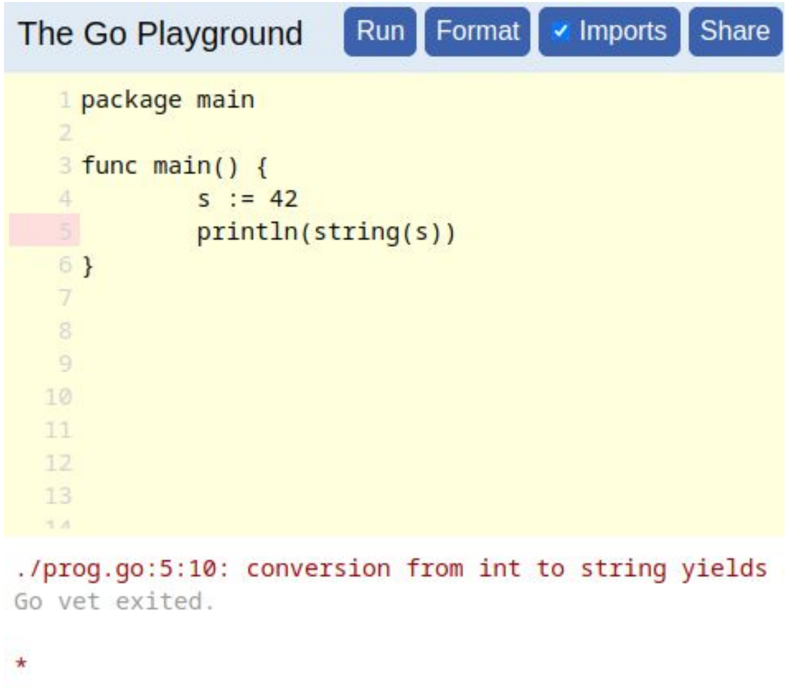
Линтеры

Статический анализ кода

Линтеры

Есть встроенный go vet:

https://play.golang.org/p/uw_odDhEpl-



```go
package main

func main() {
	s := 42
	println(string(s))
}
```

```
./prog.go:5:10: conversion from int to string yields
Go vet exited.

*
```

Линтеры

github.com/golangci/golangci-lint

asciicheck bodyclose contextcheck cyclop deadcode depguard dogsled dupl durationcheck errcheck errname errorlint exhaustive exhaustivestruct exportloopref forbid forcetypeassert funlen gci goanalysis gochecknoglobals gochecknoinits gocognit goconst gocritic gocyclo godot godox goerr113 gofmt_common gofmt gofmt_test gofumpt goheader goimports golint gomnd gomoddirectives gomodguard goprintffuncname gosec gosimple govet govet_test ifshort importas ineffassign interfacer ireturn lll makezero maligned misspell nakedret nestif nilerr nilnil nlreturn noctx nolintlint nolintlint paralleltest prealloc predeclared promlinter revive row errcheck scopelint sqlclosecheck staticcheck_common staticcheck structcheck stylecheck tagliatelle tenv testpackage thelper tparallel typecheck unconvert unparam unused util varcheck varnamelen wastedassign whitespace wrapcheck wsl

Почитать и посмотреть

“Линтеры в Go. Как их готовить”

https://habr.com/ru/post/457970/

“GoCritic — новый статический анализатор для Go”

https://youtu.be/6SDk8ibowW4

“That's all Folks!”