OTUS

Golang Developer. Professional

otus.ru

Проверить, идет ли запись

Меня хорошо видно && слышно?

Ставим “+”, если все хорошо
“-”, если есть проблемы

Тема вебинара

Тестирование в Go

Алексей Романовский

Разработчик в Resolver Inc. (Kroll)

Правила вебинара

Активно
участвуем

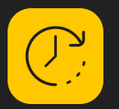
Off-topic обсуждаем
в учебной группе

Задаем вопрос
в чат или голосом

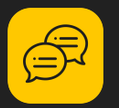
Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения

Индивидуально



Время, необходимое
на активность



Пишем в чат

Говорим голосом

Документ

Ответьте себе или
задайте вопрос

О чем будем говорить

- testing;
- testify;
- Приемы тестирования.

Зачем нужны тесты?

- Упрощают рефакторинг.
- Документируют код.
- Отделение интерфейса от реализации (mocks), менее связный код.
- Помогают найти неактуальный код.
- Помогают найти новые кейсы.
- Считают метрику для менеджмента (покрытие).
- Определяют контракт.
- Повышают качество кода.
- Придают уверенности при деплое в продакшен.

Знакомтьесь, тест в Go

```
strings_test.go // <-- ..._test.go
```

```go
func TestIndex(t *testing.T) { // <-- Test...(t *testing.T)
    const s, sub, want = "chicken", "ken", 4
    got := strings.Index(s, sub)
    if got != want {
        t.Errorf("Index(%q,%q) = %v; want %v", s, sub, got, want)
    }
}
```

https://goplay.tools/snippet/yybc8Np1JjK

testing: Error vs Fatal

```go
func TestAtoi(t *testing.T) {
    const str, want = "42", 42
    got, err := strconv.Atoi(str)
    if err != nil {
        t.Errorf("strconv.Atoi(%q) returns unexpeted error: %v", str, err)
    }
    if got != want {
        t.Errorf("strconv.Atoi(%q) = %v; want %v", str, got, want)
    }
}
```

https://goplay.tools/snippet/vjAsrBrQrxu

TitleCase

- Делает слова в строке с большой буквы.
- Кроме слов из второй строки.
- Первое слово всегда с большой буквы.

Пример:

- `TitleCase("the quick fox in the bag", "")` -> `"The Quick Fox In The Bag"`
- `TitleCase("the quick fox in the bag", "in the")` -> `"The Quick Fox in the Bag"`

testing: практика

Задание

- Дописать существующие тесты.
- Придумать один новый тест.
- Не закрывайте playground — еще пригодится :)

https://goplay.tools/snippet/PQCd4_FqLeZ

testify

https://github.com/stretchr/testify

```
func TestAtoi(t *testing.T) {
    const str, want = "42", 42
    got, err := strconv.Atoi(str)
    require.NoError(t, err)
    require.Equal(t, want, got)
}
```

https://goplay.tools/snippet/5cpT652lEyy (IDE)
https://goplay.tools/snippet/9h-9ha70qTb (playground)

testify: assert vs require

Если не уверены - используйте require.

testify: изучение API

- `require.Equal()` VS `require.Equalf()`
-
-
-

testify: изучение API

- `require.Equal()` VS `require.Equalf()`
- `require.True(t, err == nil, msg)`
-
-

testify: изучение API

- `require.Equal()` VS `require.Equalf()`
- `require.True(t, err == nil, msg)`
- `require.Nil(t, err)`
-

testify: изучение API

- `require.Equal()` VS `require.Equalf()`
- `require.True(t, err == nil, msg)`
- `require.Nil(t, err)`
- `require.NoError(t, err)`

testify: практика

Задание

- Переписать тесты на testify.
- Не закрывайте playground — еще пригодится :)

Табличные тесты

```go
func TestParseInt(t *testing.T) {
    tests := []struct {
        str      string
        expected int64
    }{
        {"-128", -128},
        {"0", 0},
        {"127", 127},
    }

    for _, tc := range tests {
        got, err := strconv.ParseInt(tc.str, 10, 8)
        require.NoError(t, err)
        require.Equal(t, tc.expected, got)
    }
}

func TestParseIntErrors(t *testing.T) {
    for _, str := range []string{"-129", "128", "byaka"} {
        _, err := strconv.ParseInt(str, 10, 8)
        require.Error(t, err)
    }
}
```

https://goplay.tools/snippet/p1Bxjoh1iZp (IDE)
https://goplay.tools/snippet/GWtEanaAKp9 (Playground)

Табличные тесты: t.Run

```
func TestParseInt(t *testing.T) {
    tests := []struct {
        str      string
        expected int64
    }{
        {"-128", -128},
        {"0", 0},
        {"127", 127},
    }

    for _, tc := range tests {
        tc := tc
        t.Run(tc.str, func(t *testing.T) {
            got, err := strconv.ParseInt(tc.str, 10, 8)
            require.NoError(t, err)
            require.Equal(t, tc.expected, got)
        })
    }
}
```

https://goplay.tools/snippet/R9YMRmsM2h3 (IDE)
https://goplay.tools/snippet/v-_TxOG6isX (Playground)

Табличные тесты: практика

Задание

- Переписать тесты на табличные.
- Постараться придумать еще один тест.
- Можно закрывать playground :)

Как запускать тесты

Все тесты в пакете и подпакетах:

```
go test ./...
go test ./pkg1/...
go test github.com/otus/superapp/...
```

Конкретные тесты по имени:

```
go test -run TestFoo
```

По тегам (`//go:build integration`):

```
go test -tags=integration
```

Coverage

- `go test -cover` — посмотреть покрытие
- `go test -coverprofile=c.out` — записать отчет о покрытии
- `go tool cover -html=c.out` — посмотреть отчет о покрытии

https://blog.golang.org/cover

Golden files

https://github.com/OtusGolang/webinars_practical_part/blob/master/03-unit-testing/golden_files/golden_test.go

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Заполните, пожалуйста, опрос о занятии по ссылке в чате

Следующий вебинар

21 февраля

Продвинутое тестирование в Go

Ссылка на вебинар будет в ЛК за 15 минут

Материалы к занятию в ЛК — можно изучать

Обязательный материал обозначен красной лентой

Спасибо за внимание!

Приходите на следующие вебинары

Алексей Романовский

Разработчик в Resolver Inc. (Kroll)