GO-06 04: Test coverage

Описание:

В этом задании мы научимся определять, хорошо ли мы написали тесты для нашей программы или же надо что-то доработать. В проверке нам будет помогать такой механизм, как test coverage.
Test coverage - это механизм, который определяет, какая часть кода выполняется во время запуска тестов. Также test coverage можно воспринимать как некую метрику, которая показывает, насколько код готов к работе.
Например, у нас есть notification сервис и нас попросили добавить в него новый функционал оповещений через sms. При этом покрытие этого сервиса (coverage) уже составляет 80%, это означает, что тесты, которые написаны к этому сервису, охватывают 80% кода сервиса.
Когда мы напишем весь нужный функционал sms оповещений и посмотрим на наш coverage, то увидим, что он уменьшился на N% из-за того, что появился новый функционал, который еще не покрыт тестами.
Важно понимать, что эта метрика не является истиной в последней инстанции, так как механизм проверки завязан не на то, какие сценарии использования покрывают тесты, а на то, сколько раз тест посещал ту или иную строчку кода (может быть, 1 раз, а может, 10, а может, и вообще не посещал). Но если вы в своих тестах стараетесь покрывать все возможные пользовательские сценарии, то и coverage будет явно стремиться к 100 процентам.
Запуск coverage
Чтобы проверить покрытие, в go не нужно устанавливать каких-либо сторонних библиотек, достаточно вызвать команду go test с аргументом -cover.

```
go test -cover
```

Вывод у данной команды будет следующим:

```
PASS
coverage: 92.0% of statements
ok    module06    0.002s
```

Как мы видим, coverage равен 92%, и это означает, что при тестировании было задействовано не меньше, чем 92% строк нашего кода (не тестового, а именно функционального).
Если же нам надо посмотреть более подробную информацию, тогда надо записать информацию о coverage в отдельный файл при помощи флага -coverprofile=[output_file]:

```
go test -coverprofile=profile.out
```

А затем вывести на просмотр в удобном нам формате при помощи утилиты go tool cover:

- в HTML, где графически будет показано, какие строчки кода тестируются и по сколько раз, а какие - нет.

```
go tool cover -html=profile.out
```

- в консоль, с более подробной информацией по каждой функции:

```
go tool cover -func=profile.out
```

Полезные ссылки:

- The cover story

Задание:

Заданием будет являться проверка coverage написанных ранее тестов.

1. В вашем проекте module06 сделайте новую ветку module06_04.
2. Запустите тесты которые вы написали ранее и посмотрите на общий coverage.
3. Посмотрите более подробную информацию о покрытии каждой функции.
4. Посмотрите информацию о том, какие строки остались без покрытия тестами. Если такие имеются, попробуйте покрыть их дополнительными тестами либо измените существующие.
5. В результате ваших действий у вас должен получиться файл profile.out, который надо закоммитить в корень проекта.
6. В качестве ответа пришлите ссылку на merge request в ветку master вашего проекта ветки module06_04.