GeekBrains

Реализация и тестирование веб-сервера

Backend-разработка на Go. Уровень 1

Что будет на уроке

1. Посмотрим на состав пакета net/http
2. Напишем веб-сервер с нуля
3. Посмотрим, как можно протестировать веб-сервер с помощью библиотеки httptest

net/http

- Сервер: https://golang.org/pkg/net/http/#Server
  - Хэндлер (+популярные виды): https://golang.org/pkg/net/http/#Handler
  - Простой роутер (обычно не используется): https://golang.org/pkg/net/http/#ServeMux
- Клиент: https://golang.org/pkg/net/http/#Client
  - Транспорт - тонкая настройка: https://golang.org/pkg/net/http/#Transport

net/http

- Всё для работы с http-запросом: https://golang.org/pkg/net/http/#Request
- Всё для работы с http-ответом: https://golang.org/pkg/net/http/#Response
- Заголовки: https://golang.org/pkg/net/http/#Header
- Полезные константы: https://golang.org/pkg/net/http/#pkg-constants
    - Текст по коду: https://golang.org/pkg/net/http/#StatusText

httptest

- Тестирование хэндлеров handler(request, responseWriter)
  - Request: https://golang.org/pkg/net/http/httptest/#NewRequest
  - ResponseWriter: https://golang.org/pkg/net/http/httptest/#ResponseRecorder
- Тестирование клиента (замокать сервер):
  - https://golang.org/pkg/net/http/httptest/#Server

Пример клиента для REST API

- Сам клиент: https://github.com/rumyantseva/go-velobike/blob/main/velobike/history.go
- Тесты к нему: https://github.com/rumyantseva/go-velobike/blob/main/velobike/history_test.go

Пример реализации сервера

- Сервер: https://github.com/rumyantseva/tenerife/blob/0ea9d08885e8fac7c7aee7183dd57c86d013a7e7/cmd/tenerife/main.go#L43
- Хэндлер: https://github.com/rumyantseva/tenerife/blob/0ea9d08885e8fac7c7aee7183dd57c86d013a7e7/internal/application/home.go
- Очень маленький пример на тестирование хэндлера: https://github.com/rumyantseva/tenerife/blob/0ea9d08885e8fac7c7aee7183dd57c86d013a7e7/internal/application/home_test.go

Практика: http.Server

1. Пишем минимальный сервер и смотрим на внутренности ListenAndServe
2. Смотрим пул соединений
3. Смотрим Graceful Shutdown

Практика: хэндлеры и тесты

1. Обрабатываем GET и POST запросы:

   - Смотрим, что есть в http.Request

   - Смотрим работу с заголовками

2. Пишем тесты

Практика: специальные хэндлеры - файловый сервер

Смотрим реализацию файлового сервера (по методичке)

Домашка

!!! Прочитать методичку к следующему уроку (про выбор роутера)!

На занятии будем смотреть разные роутеры и инструменты для генерации сервера по документации в OpenAPI.

Также поговорим про понятия “middleware” и “контекст запроса”.

Домашка

1. Добавить в пример с файловым сервером из методички возможность получить список всех файлов на сервере (имя, расширение, размер в байтах).
2. С помощью query-параметра, реализовать фильтрацию выводимого списка по расширению (то есть, выводить только .png файлы, или только .jpeg).
3. *Текущая реализация сервера не позволяет хранить несколько файлов с одинаковым названием (т.к. они будут храниться в одной директории на диске). Подумайте, как можно обойти это ограничение?
4. К коду, написанному в рамках заданий 1-3, добавьте тесты с использованием библиотеки httptest.

Вопросы
и
Ответы