**Содержание**

[1 Задание для выполнения 2](#_Toc161004759)

[1.1 Найти отсутствующую зависимость и указать ее в соответствующем блоке в build.gradle, чтобы проект снова начал собираться 2](#_Toc161004760)

[1.2 В некоторых классах поправить имя пакета 2](#_Toc161004761)

[2](#_Toc161004762)

[1.3 Собрать документацию проекта, найти в ней запросы состояния и сущности по идентификатору 3](#_Toc161004763)

[3](#_Toc161004764)

[1.4 Собрать jar со всеми зависимостями (так называемый UberJar), после чего запустить приложение. По умолчанию, сервер стартует на порту 8080. 4](#_Toc161004765)

[1.5 Запросить состояние запущенного сервера (GET запрос по адресу http://localhost:8080) и запросить сущность по идентификатору (GET запрос по адресу: http://localhost:8080/сущность/идентификатор) 4](#_Toc161004766)

[1.6 В задаче shadowJar добавить к jar-файлу вашу фамилию 4](#_Toc161004767)

[1.7 Выполнить задачу checkstyleMain. Посмотреть сгенерированный отчет. Устранить ошибки оформления кода. 5](#_Toc161004768)

[2 Ответы на вопросы 6](#_Toc161004769)

[Вывод 7](#_Toc161004770)

# 1 Задание для выполнения

# Вариант №14. Номер места вс списке группы 29.

## 1.1 Найти отсутствующую зависимость и указать ее в соответствующем блоке в build.gradle, чтобы проект снова начал собираться

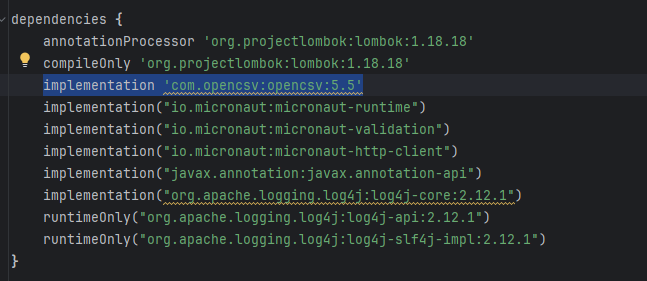
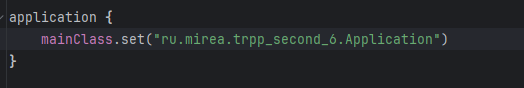


Рисунок 1 – Отсутствующая зависимость

## 1.2 В некоторых классах поправить имя пакета

## 

Рисунок 2 – Изменения в файле MessageController.java



## 

## 1.3 Собрать документацию проекта, найти в ней запросы состояния и сущности по идентификатору

## 

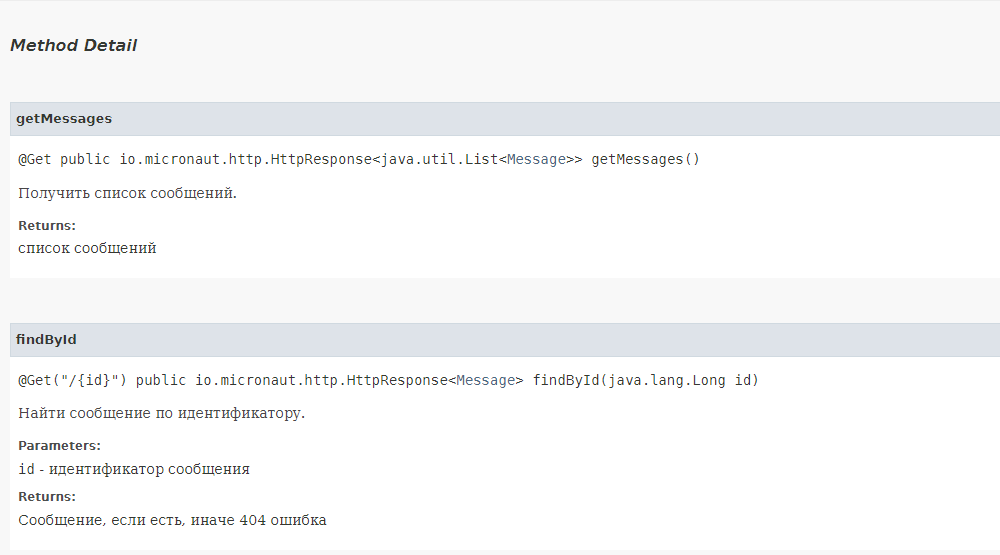
Рисунок 3 – Сбор документации проекта

Рисунок 4 – Запросы состояния и сущности

## 1.4 Собрать jar со всеми зависимостями (так называемый UberJar), после чего запустить приложение. По умолчанию, сервер стартует на порту 8080.

Рисунок 5 – Сборка Uberjar и запуск приложения

## 1.5 Запросить состояние запущенного сервера (GET запрос по адресу http://localhost:8080) и запросить сущность по идентификатору (GET запрос по адресу: http://localhost:8080/сущность/идентификатор)

Рисунок 6 – Запрос состояния и сущности

## 1.6 В задаче shadowJar добавить к jar-файлу вашу фамилию

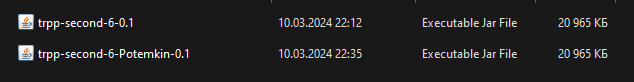
Рисунок 7 – Добавление фамилии к jar-файлу при создании

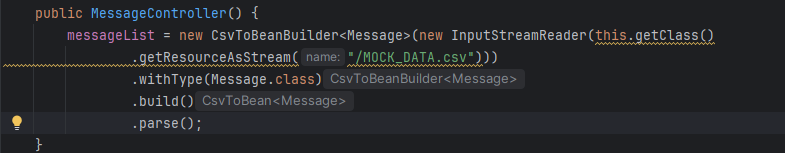
Рисунок 8 – Результат shadowJar

## 1.7 Выполнить задачу checkstyleMain. Посмотреть сгенерированный отчет. Устранить ошибки оформления кода.

Рисунок 9 – Выполнение checkstyleMain



Рисунок 10 – Исправление ошибок



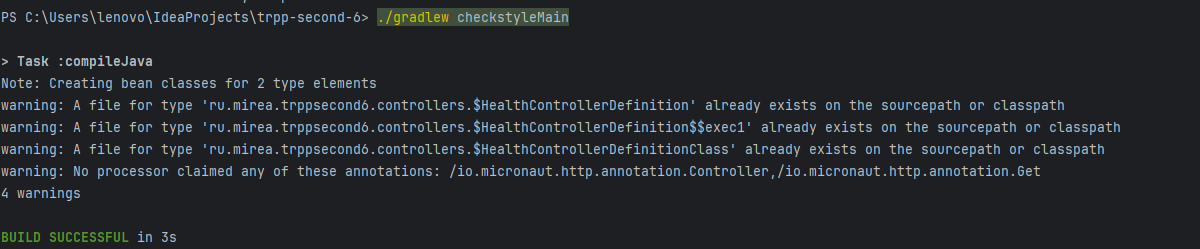
Рисунок 11 – Исправление ошибок

Рисунок 12 – Выполнение checkstyleMain

# 2 Ответы на вопросы

1. **Что делает задача clean?**

Задача clean в Gradle удаляет каталог сборки проекта, очищая все скомпилированные классы и другие результаты сборки.

1. **Что делает задача build?**

Задача build в Gradle является агрегирующей задачей, которая выполняет полный цикл сборки проекта. Она компилирует и тестирует ваш код, а затем собирает его в исполняемый jar-файл или war-файл (для веб-приложений).

1. **Что делает задача compileJava?**

Задача compileJava в Gradle компилирует исходный код Java проекта. Она преобразует файлы .java в байт-код Java, который сохраняется в файлах .class.

1. **Что делает задача run?**

Задача run в Gradle запускает приложение, используя определенную в проекте точку входа. Обычно это используется для запуска приложений, написанных на языках, таких как Java или Groovy.

1. **Что делает задача test?**

Задача test в Gradle запускает автоматические тесты, которые были написаны для приложения. Это могут быть модульные тесты, интеграционные тесты или любой другой тип тестов, которые вы написали.

1. **Что такое javadoc?**

Javadoc - это инструмент, включенный в JDK (Java Development Kit), который генерирует документацию API для исходного кода Java. Он читает комментарии Javadoc в исходном коде и создает HTML-страницы, которые описывают публичные и защищенные классы, интерфейсы, конструкторы, методы и поля каждого пакета.

# Вывод

В ходе выполнения этой практической работы мы углубили свои знания в области использования системы сборки Gradle. Мы изучили основные задачи, такие как clean, build, checkstyleMain,shadowJar, run, test и т.д. Также научились управлять зависимостями в проекте.