МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

Отчет по практической работе №2

по дисциплине «Технологии разработки программных приложений»

Тема практической работы: «Системы сборки проектов»

Выполнил:

Студент группы ИКБО-04-20 Хан А.А.

Проверил: Овчинникова М.А.

Москва 2022 г.

СОДЕРЖАНИЕ

Цель работы: Знакомство с системой сборки Gradle. Возможности Gradle. Управление зависимостями.

Индивидуальный вариант: 9

Часть 1. Ход работы

Перед началом работы форкнем репозиторий согласно варианту и клонируем его на компьютер.

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

Пройдемся по всем классам в поисках ошибки. В результате находим отсутствующую зависимость в классе Nomenclature (Рисунок 1.1).

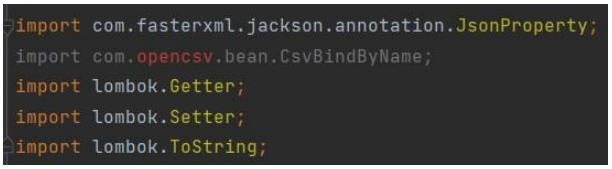

Рисунок 1.1 – Отсутствующая зависимость в классе Nomenclature

В файле build.gradle пропишем отсутствующую зависимость и соберем проект (Рисунок 1.2).

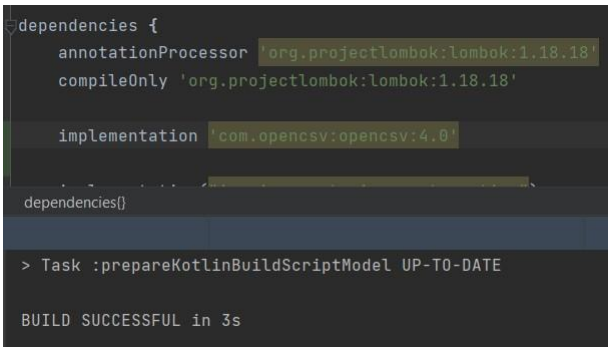

Рисунок 1.2 – Указание зависимости в build.gradle и проверка сборки проекта

Проверим устранение ошибки в файле класса Nomenclature (Рисунок 1.3).

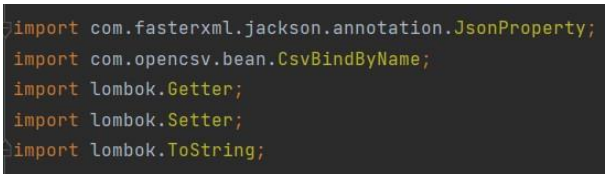

Рисунок 1.3 – Появление зависимости в классе Nomenclature

2. В некоторых классах поправить имя пакета (Рисунки 1.4-1.7).

```
@Get
public HttpResponse<HealthResponse> healthCheck() { return HttpResponse.ok(new HealthResponse("ok")); }
```

Рисунок 1.4 – Пакет HealthResponse не подключен в классе HealthController

```
@JsonProperty("price")
@CsvBindByName(column = "price")
private BigDecimal price;
```

Рисунок 1.5 – Пакет BigDecimal не подключен в классе Nomenclature

```
import java.math.BigDecimal;
```

Рисунок 1.6 – Подключение пакетов в классе Nomenclature

```
import io.micronaut.http.HttpResponse;
```

Рисунок 1.7 – Подключение пакетов в классе HealthController

3. Собрать документацию проекта, найти в ней запросы состояния и сущности по идентификатору (Рисунки 1.8-1.11).

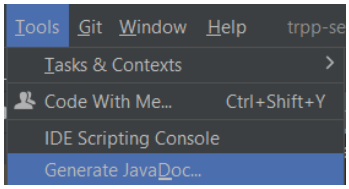

Рисунок 1.8 – Генерация JavaDoc

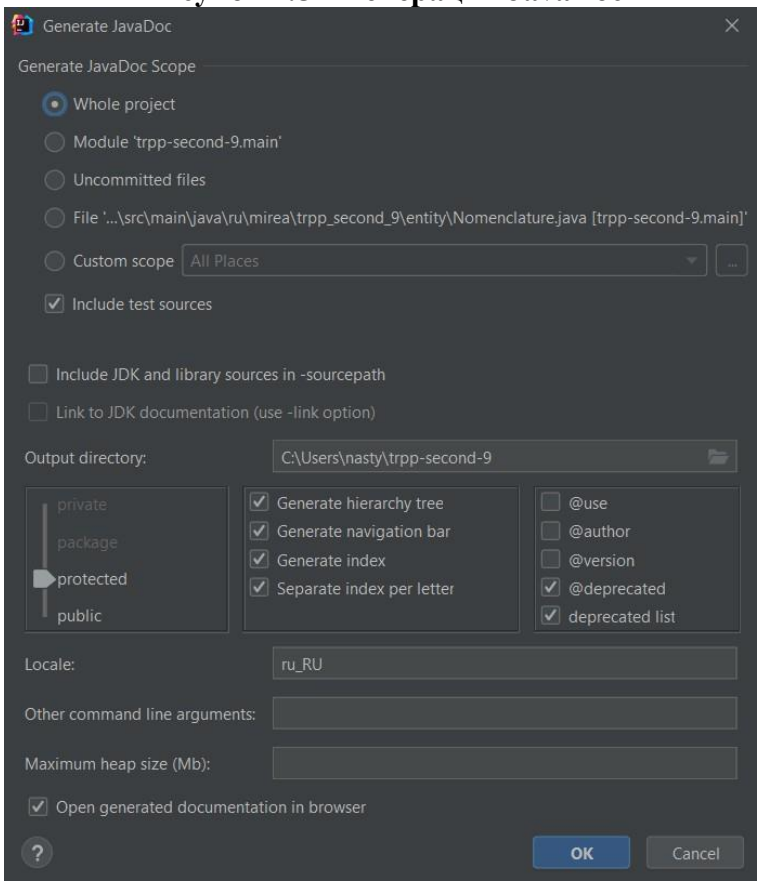

Рисунок 1.9 – Создание JavaDoc

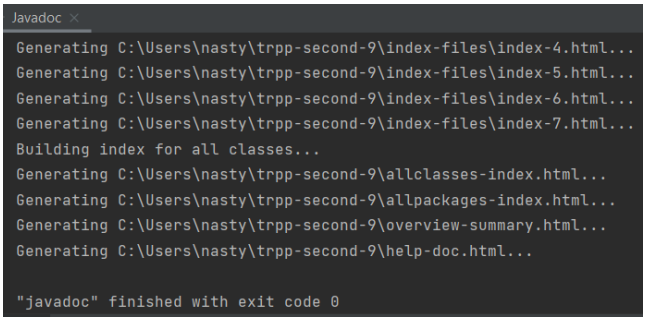

Рисунок 1.10 – Успешное завершение генерации

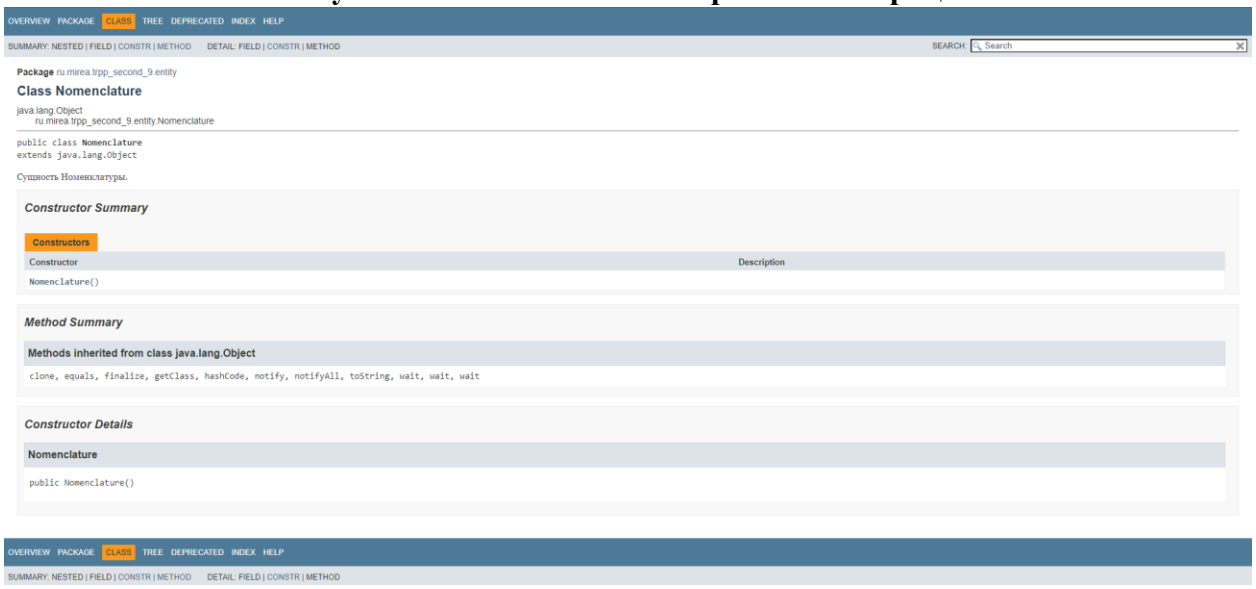

Рисунок 1.11 – Запрос состояния сущности ru.mirea.trpp_second_9.entity.Nomenclature

4. Собрать jar со всеми зависимостями, после чего запустить приложение. По умолчанию, сервер стартует на порту 8080 (Рисунки 1.12-1.3).

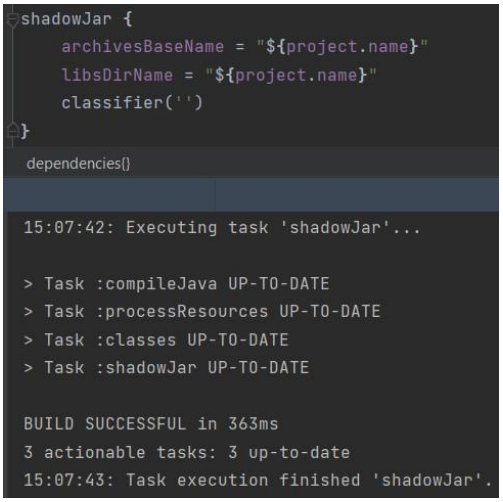

Рисунок 1.12 – Сборка jar со всеми зависимостями

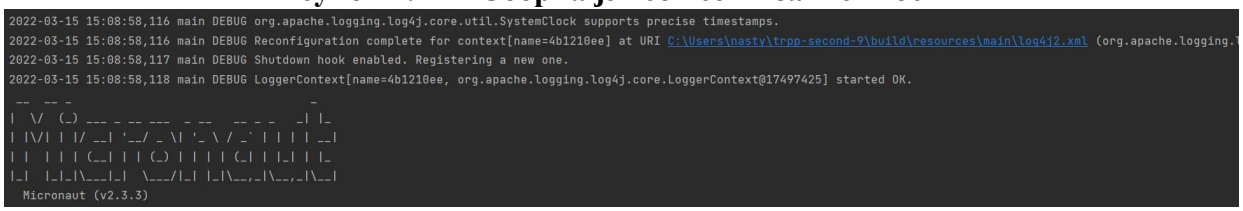

Рисунок 1.13 – Запуск приложения

5. Запросить состояние запущенного сервера (Рисунок 1.14).

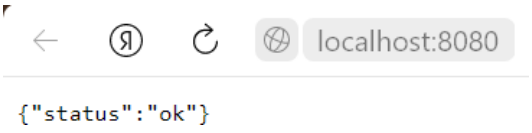

Рисунок 1.14 – Запрос состояния сервера

6. Запросить сущность по идентификатору (Рисунок 1.15).

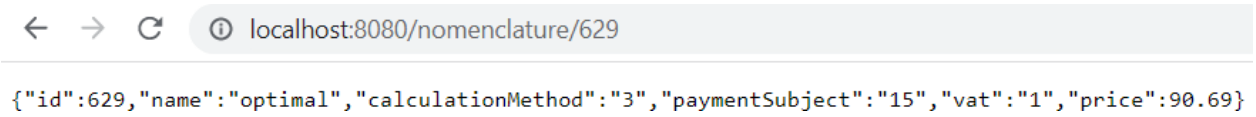

Рисунок 1.15 – Запрос сущности по идентификатору

7. В задаче shadowJar добавить к jar-файлу вашу фамилию

```
shadowJar {
    archivesBaseName = "${project.name}"
    libsDirName = "${project.name}"
    classifier('  Khan   ')
}
```

Рисунок 1.16 – Добавление фамилии

8. Выполнить задачу checkstyleMain. Посмотреть сгенерированный отчет. Устранить ошибки оформления кода.

Запустим checkstyleMain и перейдем по указанной ссылке (Рисунок 1.17).

CheckStyle Audit

Designed for use with CheckStyle and Ant.

Summary	
Files	Errors
5	6

Files	
Name	Errors
C:\Users\nasty\trpp-second-9\src\main\java\ru\mirea\trpp_second_9\controllers\NomenclatureController.java	2
C:\Users\nasty\trpp-second-9\src\main\java\ru\mirea\trpp_second_9\Application.java	1
C:\Users\nasty\trpp-second-9\src\main\java\ru\mirea\trpp_second_9\controllers\HealthController.java	1
C:\Users\nasty\trpp-second-9\src\main\java\ru\mirea\trpp_second_9\entity\HealthResponse.java	1
C:\Users\nasty\trpp-second-9\src\main\java\ru\mirea\trpp_second_9\entity\Nomenclature.java	1

File C:\Users\nasty\trpp-second-9\src\main\java\ru\mirea\trpp_second_9\Application.java	
Error Description	Line
Name 'ru.mirea.trpp_second_9' must match pattern '^[a-z]+(\.[a-z][a-z0-9]{1,})*$'.	1

Back to top

File C:\Users\nasty\trpp-second-9\src\main\java\ru\mirea\trpp_second_9\controllers\HealthController.java	
Error Description	Line
Name 'ru.mirea.trpp_second_9.controllers' must match pattern '^[a-z]+(\.[a-z][a-z0-9]{1,})*$'.	1

Back to top

File C:\Users\nasty\trpp-second-9\src\main\java\ru\mirea\trpp_second_9\controllers\NomenclatureController.java	
Error Description	Line
Name 'ru.mirea.trpp_second_9.controllers' must match pattern '^[a-z]+(\.[a-z][a-z0-9]{1,})*$'.	1
Line is longer than 120 characters (found 185).	22

Back to top

File C:\Users\nasty\trpp-second-9\src\main\java\ru\mirea\trpp_second_9\entity\HealthResponse.java

Рисунок 1.17 – Запуск checkstyleMain

Исправим 6 ошибок оформления кода. Первые пять из них связаны с ошибкой в имени пакета (Рисунок 1.18), уберем из его названия недопустимые символы (Рисунок 1.19).

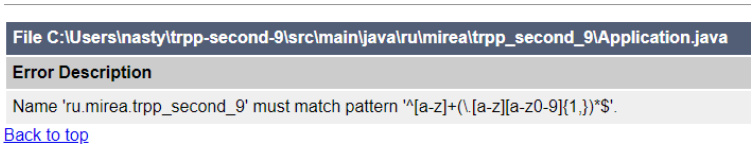

Рисунок 1.18 – Ошибка в имени пакета

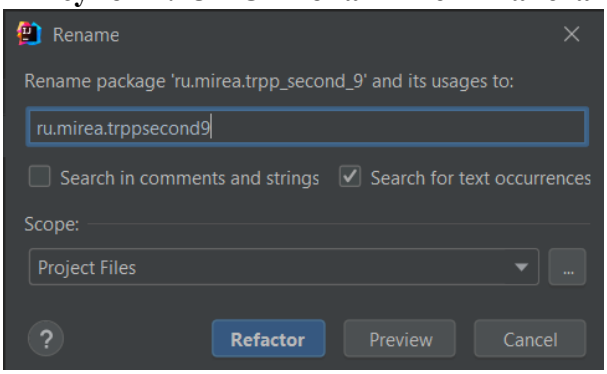

Рисунок 1.19 – Изменение имени пакета

Последняя ошибка заключается в превышении длины строки в файле класса NomenclatureController (Рисунок 1.20), исправим ее (Рисунок 1.21) и снова выполним задачу checkstyleMain. Итоговый отчет не содержит ошибок (Рисунок 1.22).

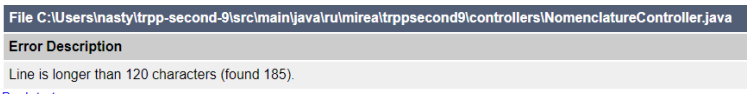

Рисунок 1.20 – Ошибка в классе NomenclatureController

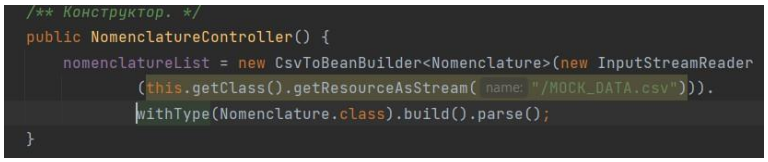

Рисунок 1.21 – Разделение строки в коде

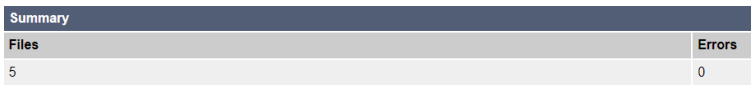

Рисунок 1.22 – Результат исправления всех ошибок

Часть 2. Ответы на вопросы

Вопрос 1: Чем компиляция отличается от сборки?

Ответ: Компиляция — это процесс превращения исходного кода в объектный код. Сборка — это последовательность, состоящая из компиляции

и компоновки с возможными другими задачами, такими как создание установщика.

Вопрос 3: Что такое репозиторий?

Ответ: Репозиторий – набор файлов и папок, являющихся частями какого-либо проекта.

Вопрос 11: Что делает задача run?

Ответ: Задача run отвечает запуск приложения.

Вопрос 14: Что такое checkstyle?

Ответ: Checkstyle - это инструмент с открытым исходным кодом, который проверяет код на соответствие настраиваемому набору правил. Отчет Checkstyle состоит из трех основных частей:

Файлы: этот раздел отчета предоставляет нам список файлов, в которых произошли нарушения. Он также показывает количество нарушений по уровням их серьезности.

Правила: в этой части отчета дается обзор правил, которые использовались для проверки нарушений. Он показывает категорию правил, количество нарушений и серьезность этих нарушений.

Подробности: подробный раздел отчета предоставляет нам подробную информацию о произошедших нарушениях. Подробная информация представлена на уровне номера строки.

Вопрос 18: Что такое postman?

Ответ: Это инструмент для работы с API, который позволяет тестировщику посылать запросы к сервисам и работать с их ответами. С его помощью можно протестировать бекэнд и убедиться, что он корректно работает.

Вопрос 19: Что такое аннотация в Java?

Ответ: Это специальная форма синтаксических метаданных, которая может быть добавлена в исходный код. Аннотации используются для анализа кода, при компиляции или во время выполнения программы. Их можно

применять к пакетам, классам, методам, переменным и параметрам. Одной из самых известных является аннотация @Override, которая обозначает что мы собираемся переопределить метод родительского класса. Пример аннотации @Override представлен на Рисунке 2.1.

```
@Override
public void onClick(View view) {
    Intent i = new Intent(MainActivity.this, DisplayMessageActivity.class);
    MainActivity.this.startActivity(i);
}
```

Рисунок 2.1 – Переопределение метода onClick

Вывод

В ходе работы были приобретены практические навыки по работе с системой сборки проектов Gradle по управлению зависимостями.

Список информационных источников

1. Методические указания для выполнения практической работы №2 «Системы сборки проектов» М. А. Овчинникова, МИРЭА – Российский технологический университет, 2022.
2. JavaRush – [Электронный ресурс] URL: https://javarush.ru/groups/posts/1896-java-annotacii-chto-ehto-i-kak-ehtim-poljhzovatjhsja (Последнее обращение 15.03.2022).
3. Хабр – [Электронный ресурс] URL: https://habr.com/ru/company/maxilect/blog/596789/ (Последнее обращение 15.03.2022).
4. Введение в checkstyle – [Электронный ресурс] URL: https://ru.minecraftfullmod.com/744-introduction-to-checkstyle (Последнее обращение 15.03.2022).