|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| *Федеральное государственное бюджетное образовательное учреждение высшего образования* ***«МИРЭА – Российский технологический университет»***  **РТУ МИРЭА** |

Институт Информационных технологий (ИТ)

Кафедра Математического обеспечения и стандартизации информационных технологий (МОСИТ)

|  |
| --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 2** |
| **по дисциплине** |
| **«Технология разработки программных приложений»**  **Тема: «Системы сборки»** |

|  |  |
| --- | --- |
| Выполнил студент группы \_\_\_\_\_ИКБО-03-21\_ | Хречко С.В, |
| Принял ассистент кафедры МОСИТ | Петрова А. А. |

|  |  |  |
| --- | --- | --- |
| Практическая работа выполнена | «\_\_»\_\_\_\_\_\_\_2023\_г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_2023\_ г. | *(подпись руководителя)* |

Москва 2023г.

Оглавление

[Задачи 2](#_Toc130843464)

[Выполнение 3](#_Toc130843465)

[Пункт 1 3](#_Toc130843466)

[Пункт 2 3](#_Toc130843467)

[Пункт 3 4](#_Toc130843468)

[Пункт 4 5](#_Toc130843469)

[Пункт 5 6](#_Toc130843470)

[Пункт 6 6](#_Toc130843471)

[Пункт 7 6](#_Toc130843472)

[Пункт 8 7](#_Toc130843473)

[Ответы на контрольные вопросы 9](#_Toc130843474)

[Вопрос 2 9](#_Toc130843475)

[Вывод 9](#_Toc130843476)

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

# Задачи

Для выполнения необходимо клонировать (или форкнуть) git-репозиторий согласно варианту, и выполнить следующие задания:

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

2. В некоторых классах поправить имя пакета

3. Собрать документацию проекта, найти в ней запросы состояния и сущности по идентификатору

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

5. Запросить состояние запущенного сервера (GET запрос по адресу

http://localhost:8080)

6. Запросить сущность по идентификатору (GET запрос по адресу: <http://localhost:8080/сущность/идентификатор>) Идентификатором будут 3 последних цифры в серийном номере вашего студенческого билета.

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

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

Вариант 12

Репозиторий: <https://github.com/rtu-mirea/trpp-second-12>,

Сущность: ru.mirea.entity.Animal

# Выполнение

## Пункт 1

На рисунке 1 представлена ошибка из-за отсутствующей зависимости.

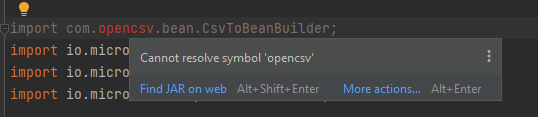


Рисунок 1 – Ошибка подключения библиотеки

На рисунке 2 представлено добавление недостающей зависимости.



Рисунок 2 – Добавление зависимости

## Пункт 2

В некоторых классах используются классы из другого пакета, решение этой проблемы будет подключение этого класса. На рисунке 3 представлено добавление класса.



Рисунок 3 – Добавление класса Animal

На рисунке 4 представлено добавление класса BigDecimal для класса Animal.



Рисунок 4 – Добавление класса Animal

## Пункт 3

Теперь, когда проект успешно собирается, соберем документацию. Команда для сбора документации представлена на рисунке 5.



Рисунок 5 – Сбор документации

Найдем в документации запросы статуса и сущности по идентификатору из варианта задания, это ru.mirea.entity.Animal. Найденное представлено на рисунках 6, 7.

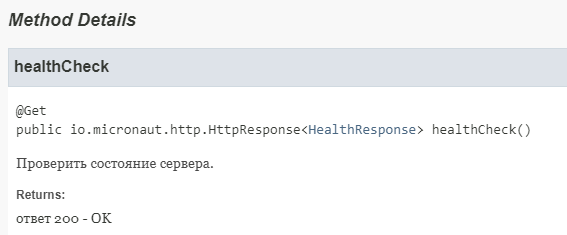


Рисунок 6 – Проверка состояния

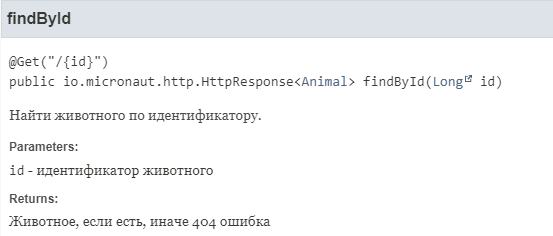


Рисунок 7 – Запрос сущности по идентификатору

## Пункт 4

Соберем UberJar. Для этого выполняем команду shadowJar. Однако для ее успешного выполнения требуется использовать более новую версию плагина – представлено на рисунке 8.



Рисунок 8 – Обновление плагина

Запустим приложение, используя команду run. Получаем ошибку представленную, на рисунке 9. Для решения проблемы, в build.gradle прописываем правильный путь до класса Application, находящегося в папке ru.mirea.trpp\_second\_12.

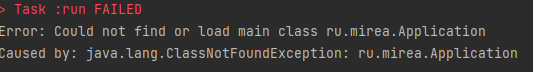


Рисунок 9 – Ошибка названия приложения

После успешного запуска, видим сообщение о запуске сервера, представленное на рисунке 10.

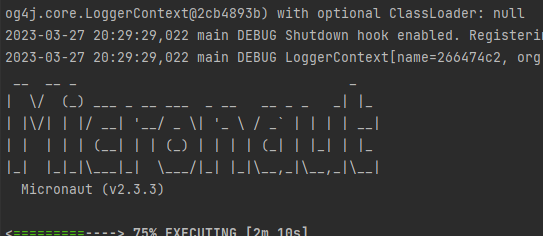


Рисунок 10 – Успешный запуск

## Пункт 5

Для проверки статуса сервера посылаем Get запрос по адресу <http://localhost:8080>. Сделать это можно воспользовавшись curl в командной строке или же просто веб-браузером. Результат представлен на рисунке 11.

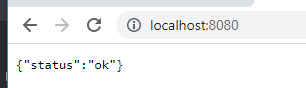


Рисунок 11 – Статус сервера

## Пункт 6

Запросим сущность по адресу <http://localhost:8080/сущность/идентификатор>. Идентификатором выступают последние три цифры в серийном номере студенческого билета, а именно: 649. Сущность это animal, в соответствии с вариантом задания. Результат представлен на рисунке 12.

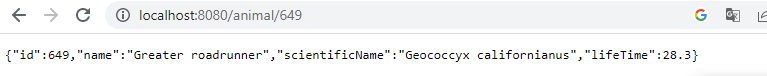


Рисунок 12 – Запрос сущности по идентификатору

## Пункт 7

Изменим задачу shadowJar, так чтобы в имени jar файла присутствовала фамилия. Изменения представлены на рисунке 13.



Рисунок 13 – Изменения в команде shadowJar

## Пункт 8

Исполним команду checkstyleMain. Полученное сообщение представлено на рисунке 13, а отчет на рисунке 14.

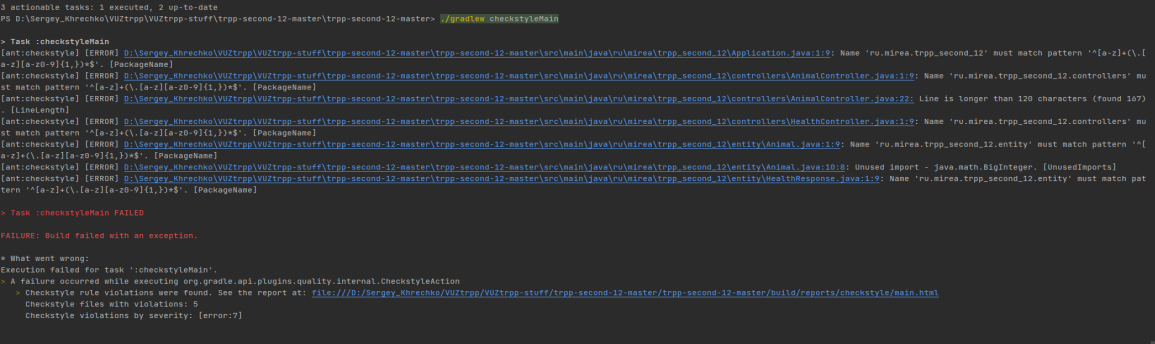


Рисунок 13 – Изменения в команде shadowJar

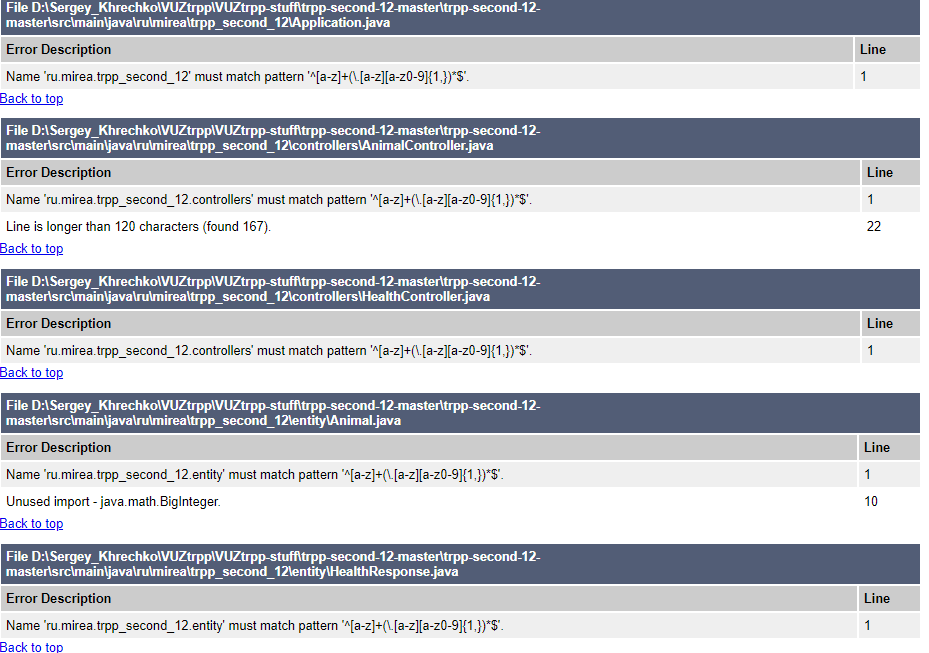


Рисунок 13 – Изменения в команде shadowJar

Исправим ошибки. Например имена не должны содержать нижних подчеркиваний, заменим их на точки. А также удалим неиспользуемое подключение библиотеки BigInteger. Также разобьем слишком длинные строки на две. Полученный результат после исправления ошибок представлен на рисунке 14.

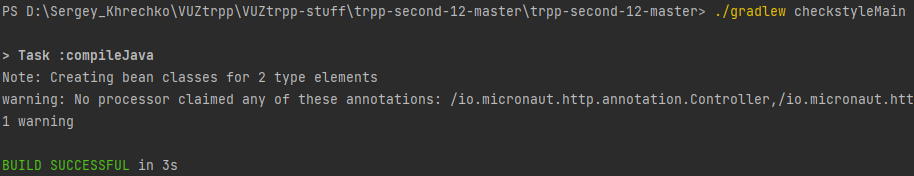


Рисунок 14 – Отсутствие ошибок

# Ответы на контрольные вопросы

## Вопрос 4

Вопрос:

Как указать зависимости проекта?

Ответ:

Чтобы указать зависимости проекта их нужно добавить в раздел dependencies файла build.gradle.

## Вопрос 11

Вопрос:

Что делает команда run?

Ответ:

Команда run запускает главный класс проекта, то есть запускает приложение.

## Вопрос 15

Вопрос:

Что такое UberJar?

Ответ:

UberJar – это единый jar архив проекта cо всеми классами, ресурсами и зависимостями проекта. Чтобы его создать нужно выполнить команду shadowJar.

## Вопрос 13

Вопрос:

Что такое javadoc?

Ответ:

javadoc – это инструмент генерирующий документацию для кода Java.

## Вопрос 8

Вопрос:

Что делает задача clean?

Ответ:

Команда clean очищает проект от созданных системой сборки файлов и папок, в частности удаляет папку build.

## Вопрос 2

Вопрос:

Что такое система сборки?

Ответ:

Система сборки – это ПО, которое автоматизирует сборку проекта из исходных файлов.

## Вопрос 1

Вопрос:

Чем компиляция отличается от сборки?

Ответ:

Компиляция – это процесс преобразования высокоуровневого кода в машинный исполняемый код. А сборка – это процесс объединения и разрешения зависимостей между файлами и библиотеками проекта.

# Вывод

В процессе выполнения работы были освоены основы работы с системой git, а также с GitHub. Была проделана обширная практическая работа для закрепления навыков. Были выполнены поставленные задачи и даны ответы на контрольные вопросы.