

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 № 8

по дисциплине «Проектирование и разработка мобильных приложений»

# Выполнил:

Студент группы ИКБО-68-23 Клейменов М.Д.

Москва, 2025 г.

**Оглавление**

[Задание 3](#__RefHeading___Toc1509_1805702352)

[Ход работы 4](#__RefHeading___Toc1511_1805702352)

[1. Реализовать запуск последовательно 3 задач. 4](#__RefHeading___Toc632_4235534367)

[2. Реализовать запуск 2 задач параллельно. 5](#__RefHeading___Toc490_390074292)

[3. Разработать приложение, которое при нажатии кнопки будет загружать 6](#__RefHeading___Toc492_390074292)

[Вывод 10](#__RefHeading___Toc496_390074292)

[Ссылка на репозиторий: репозиторий 10](#__RefHeading___Toc498_390074292)

# Задание

1. Реализовать запуск последовательно 3 задач.
2. Реализовать запуск 2 задач параллельно.
3. Разработать приложение, которое при нажатии кнопки будет загружать изображение по Rest API: https://random.dog/woof.json и выводить его на экран.

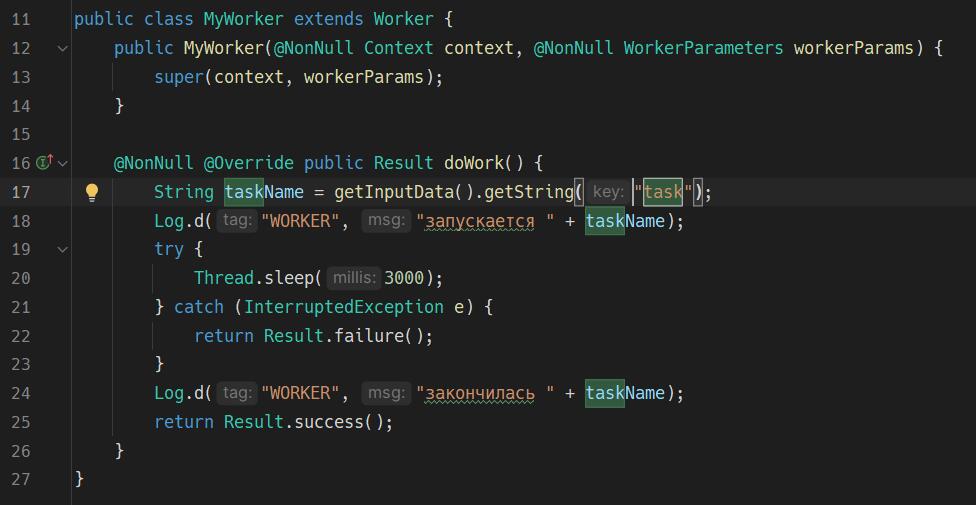
# Ход работы

# 1. Реализовать запуск последовательно 3 задач.

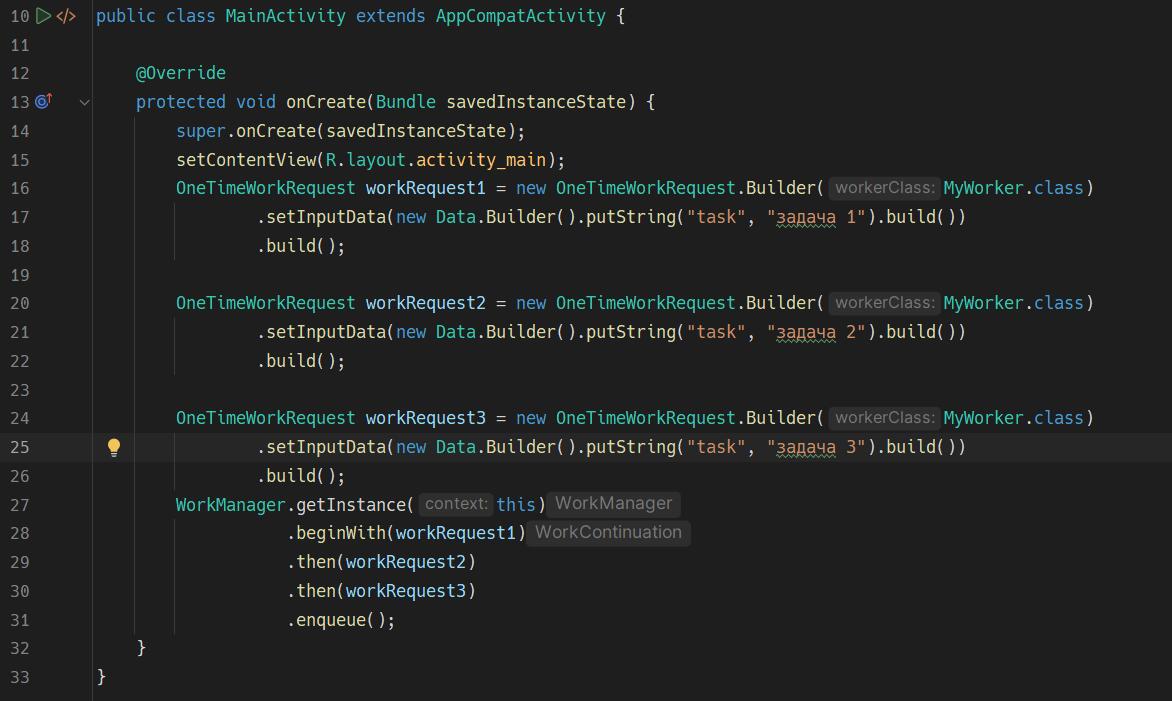
Для работы с многопоточностью подключаем в gradle зависимость work-runtime.

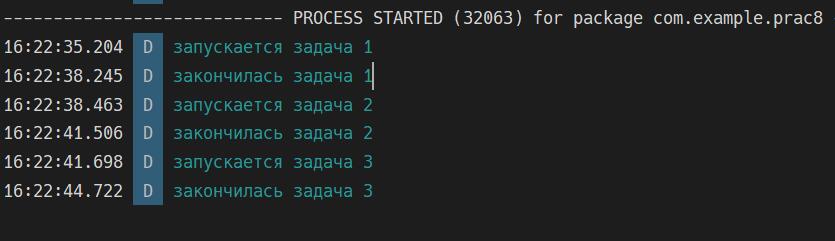
Рисунок 1 — библиотека work-runtime.

Создаем класс MyWorker, который наследуется от Worker класса, который выводит раз в 3 секунды лог.

Рисунок 2 — класс MyWorker.

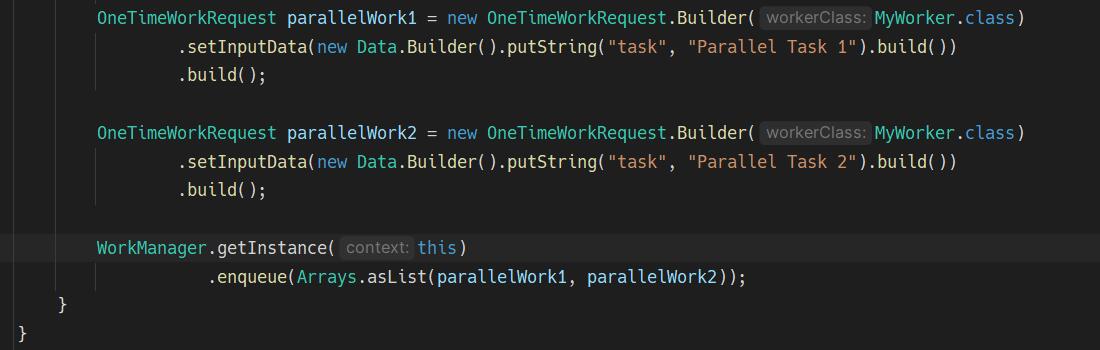
В MainActivity создаем через workRequest три потока, которые выполняются последовательно.

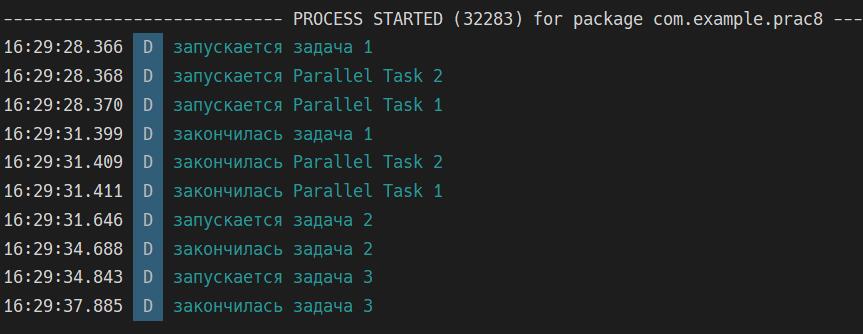
Рисунок 3 — MainActivity с передачей названий задач.

Рисунок 4 — логи вывода задач.

# 2. Реализовать запуск 2 задач параллельно.

Для параллельно выполняемых задач используем enqueue.

Рисунок 5 — MainActivity.

Рисунок 6 — вывод логов параллельных задач.

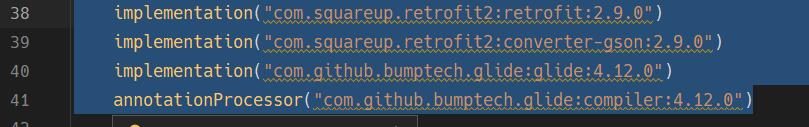
# 3. Разработать приложение, которое при нажатии кнопки будет загружать

# изображение по Rest API: https://random.dog/woof.json и выводить его на экран.

Для того, чтобы приложение смогло получить доступ к интернету, в AndroidManifest добавляем разрешение.

Рисунок 7 — разрешение интернета.

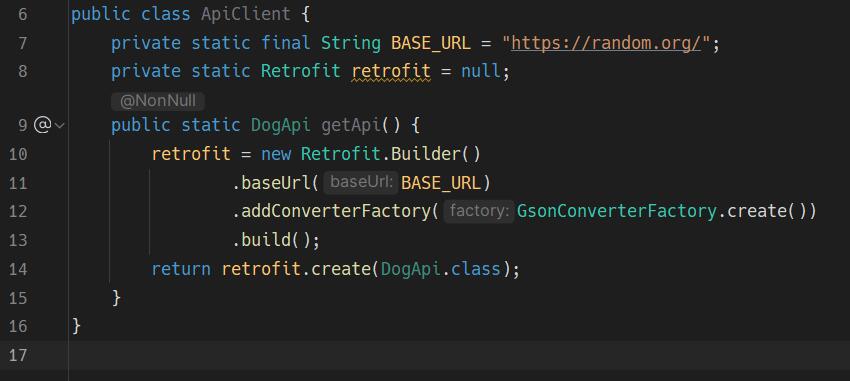
Для задачи загружаем несколько библиотек — retrofit для REST клиента и glide – для асинхронной подгрузки изображений из сети.

Рисунок 8 — зависимости retrofit2 и glide.

Определяем интерфейс, который определяет HTTP запросы к апи.

Рисунок 9 — интерфейс DogApi.

Создаем класс ApiClient, который реализует клиент для работы с DogApi.

Рисунок 10 — ApiClient.

В MainActivity создаем взаимотношение Api и интерфейса, через glide выводим на экран изображение.

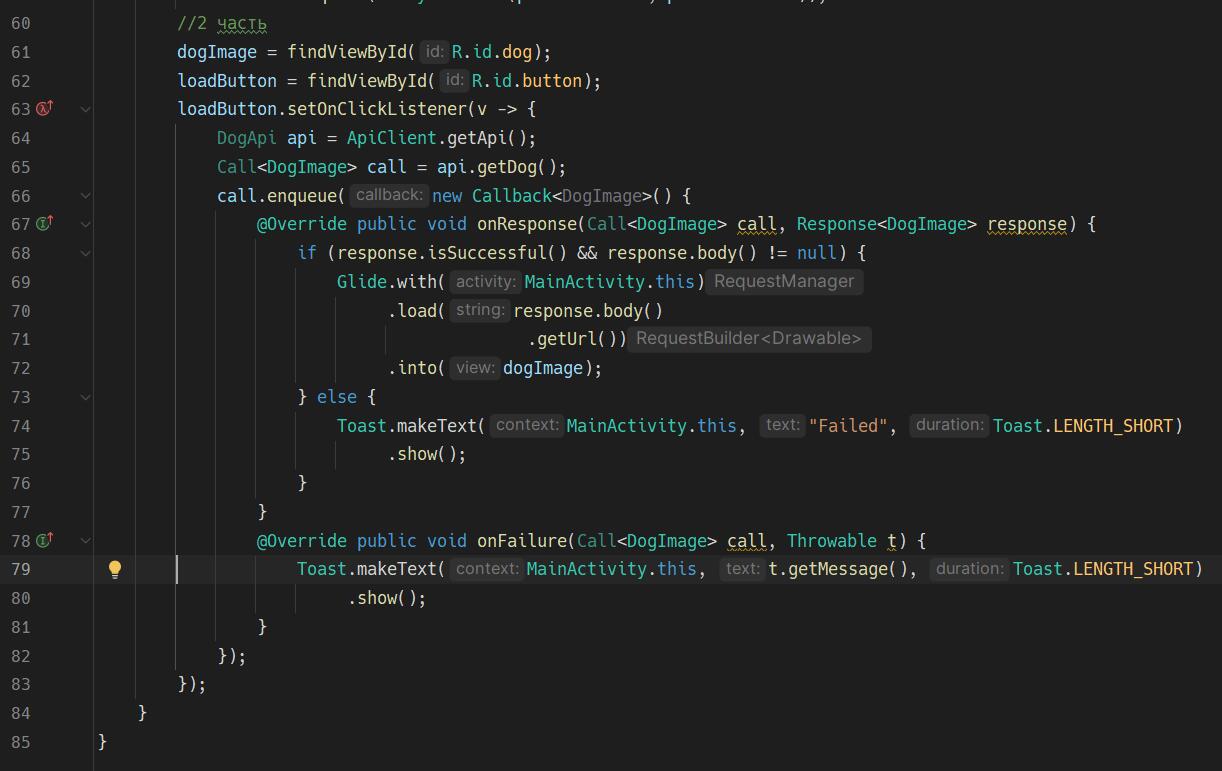
Рисунок 11 — код в MainActivity.

Рисунок 12 — загруженная случайная картинка из <https://random.dog/woof.json>.

# Вывод

В данной работе мы познакомились со Thread и многопоточностью, а также познакомились с библиотеками для REST клиента и загрузки картинок.

# Ссылка на репозиторий: [репозиторий](https://github.com/mihailkl/mobile-razrabotka_semestr4/tree/main/Prac8)