**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

Факультет прикладной математики и информатики

Кафедра информационных систем управления

Чернышева Ксения Юрьевна

Программирование Мобильных и Встраиваемых Систем

Отчет по лабораторной работе №7

«Разработка мобильного

приложения с поддержкой многопоточности и

сетевым взаимодействием»

## студентки 3 курса 12 группы

**Преподаватель**

**Давидовская Мария Ивановна**

**Минск 2024**

**Упражнение 1**

Изучить примеры реализации многопоточности

https://www.geeksforgeeks.org/multithreading-in-android-with-examples/ на Kotlin и Java.  
  
Изучила

**Упражнение 2**

1. Изучить и реализовать пример использования потоков из

https://metanit.com/java/android/10.1.php.   
  
Изучила реализовала

2. Изучить и реализовать пример использования потоков при проектировании фрагментов <https://metanit.com/java/android/10.2.php>  
  
  
Изучила реализовала

**Упражнение 3**

1. Изучить документацию https://developer.android.com/kotlin/coroutines

**2. Изучить пример из учебного курса Use Kotlin Coroutines in your Android App**Изучила **ЗАДАНИЕ 1**

На основании изученных примеров разработать приложение на Java или Kotlin с подключением к сервису Last FM (или другому музыкальному сервису), сохранять сведения о 2-3 артистах и его популярных композициях в базу данных и выводить список из 10 популярных произведений артиста при поиске по его личным данным.

Для сохранения песни и названия необходимо создать базу данных, содержащую таблицу со следующими полями:

1) ID

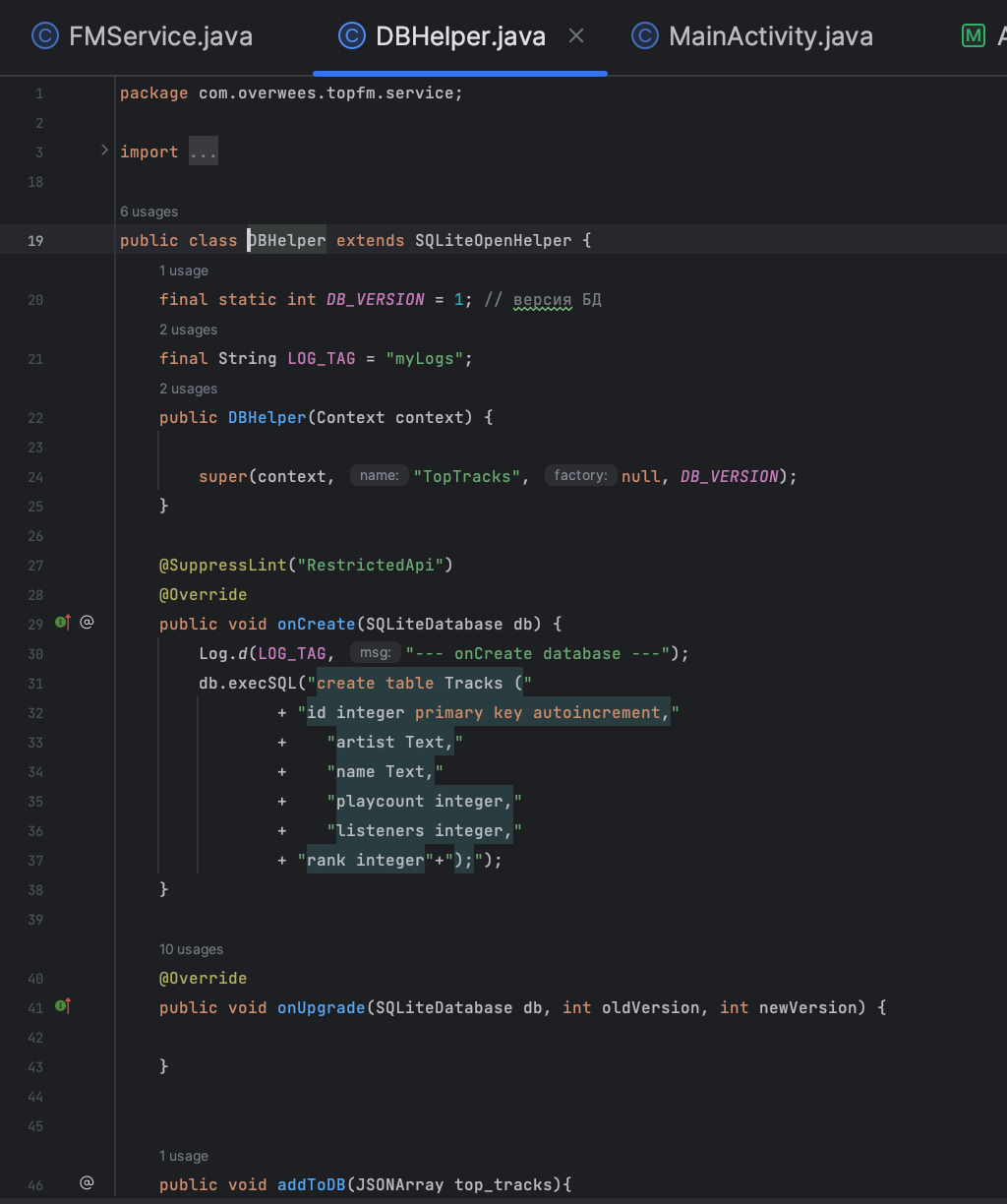
2) Исполнитель

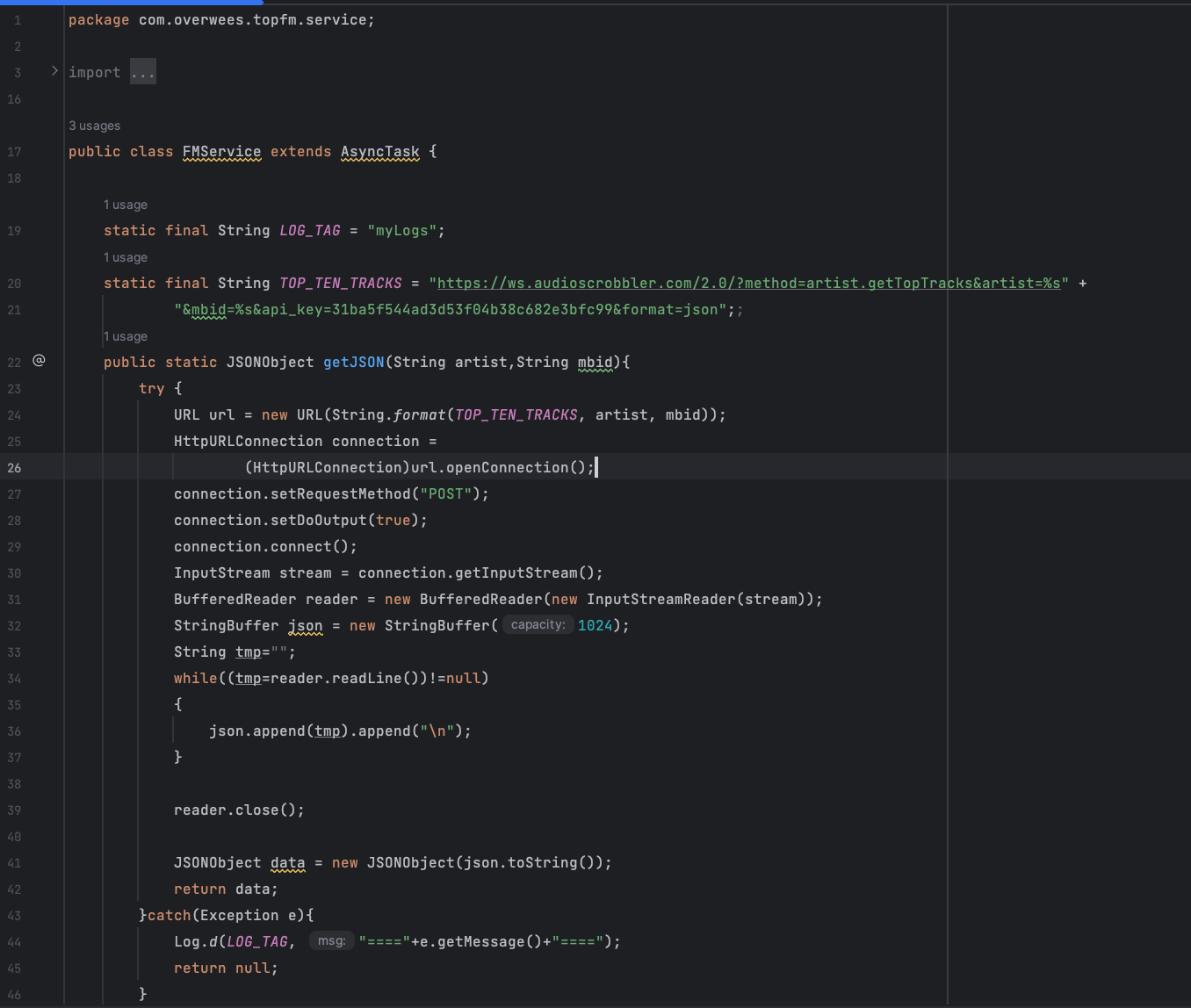
3) Название трека

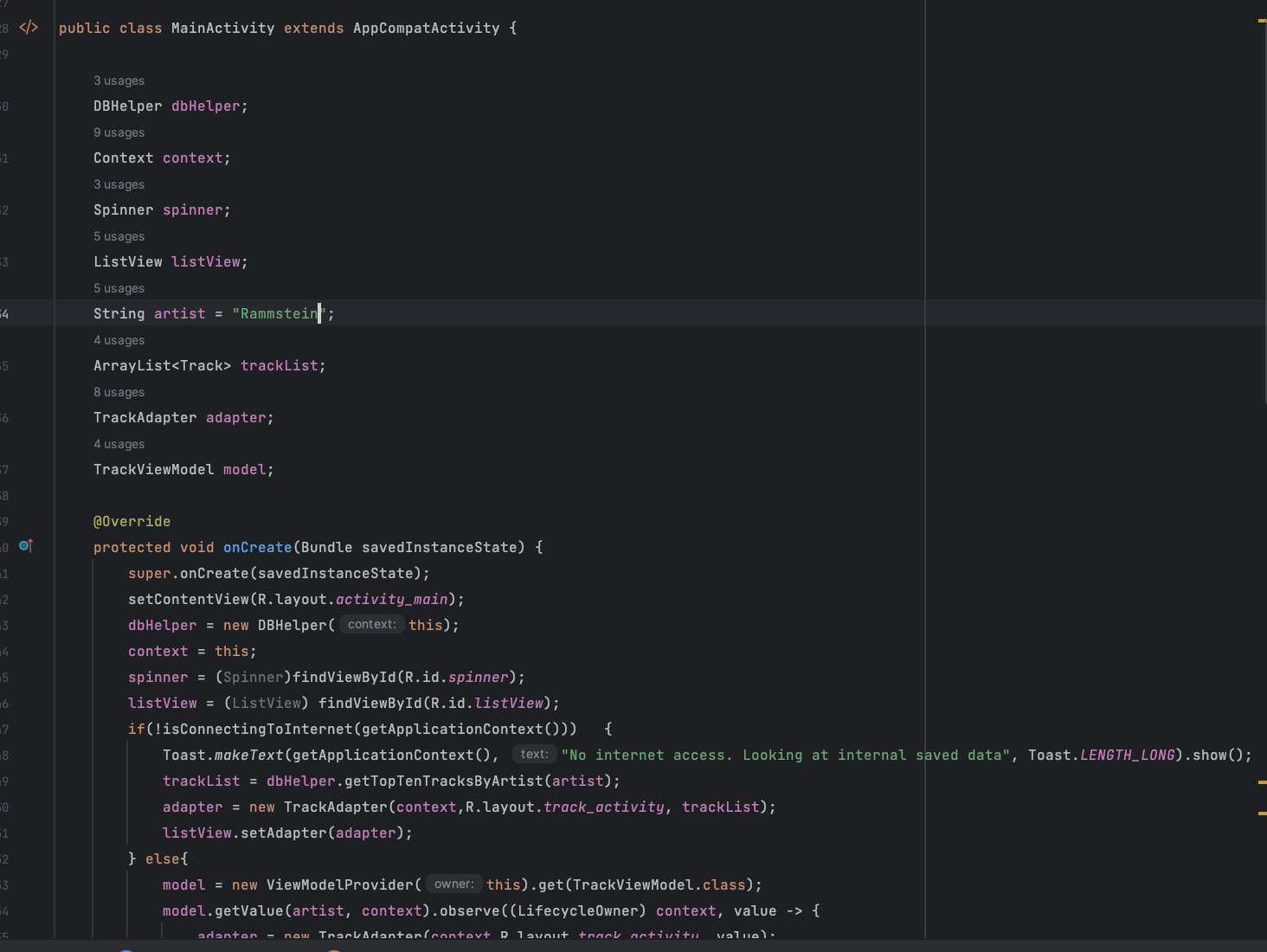
4) Количество прослушиваний

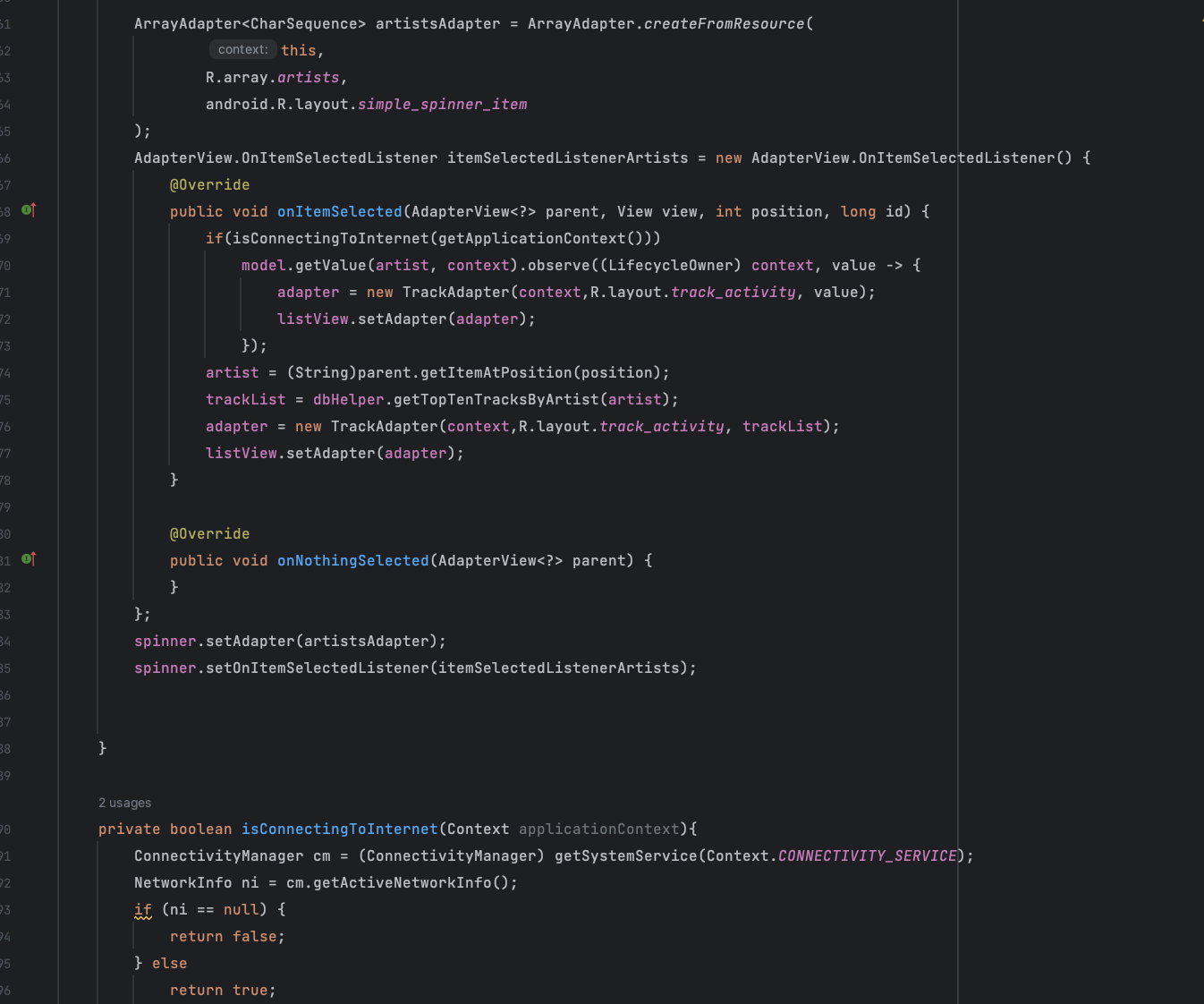
5) Количество пользователей, прослушавших композицию

При включении приложения необходимо производить проверку подключения к Интернету. В случае если подключение отсутствует – выводить всплывающее сообщение (Toast) с предупреждением о запуске в автономном режиме (доступен только просмотр внесённых ранее записей). Приложение должно содержать операцию (activity), позволяющее просматривать внесённые в базу данных записи.



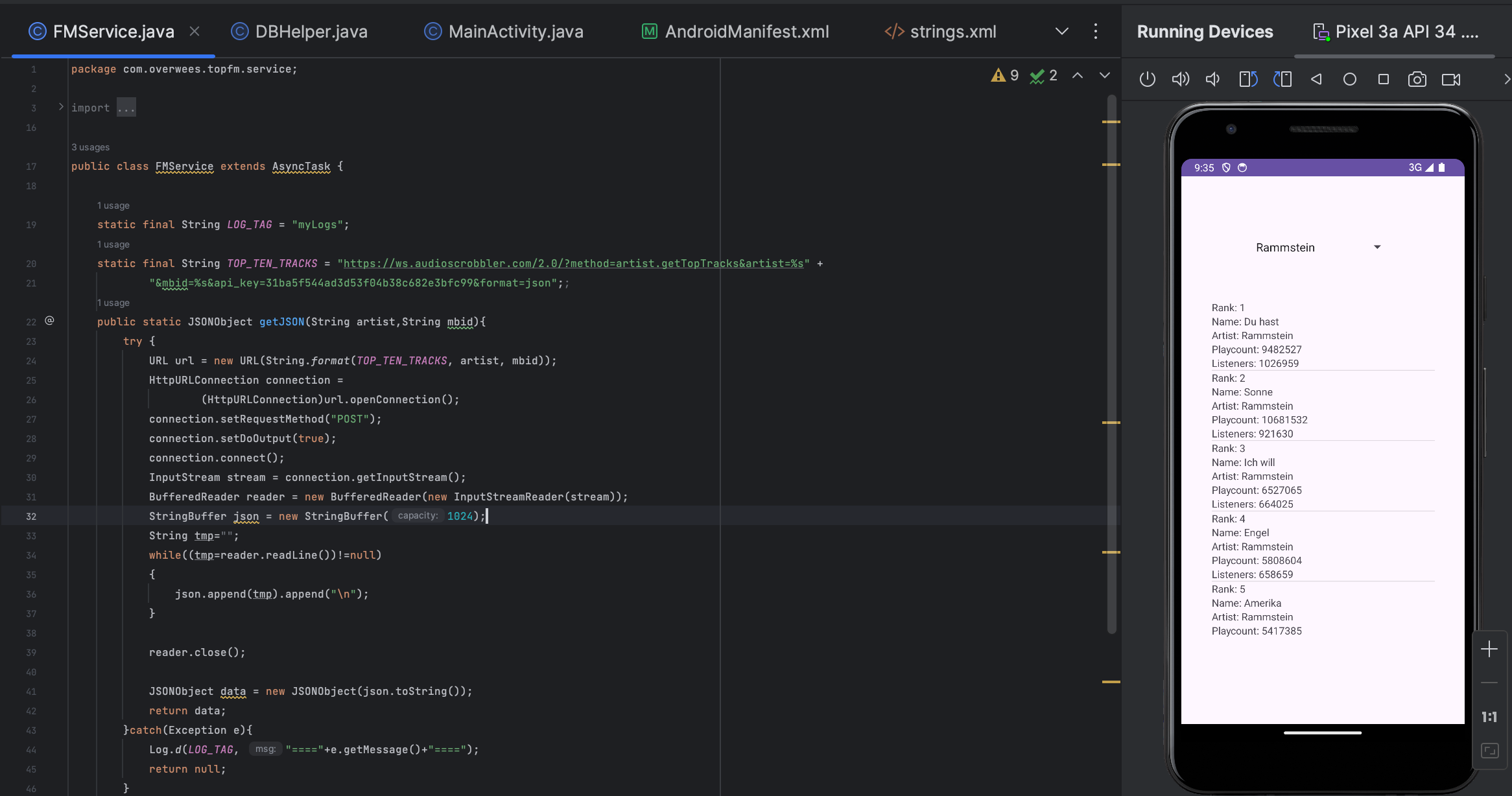


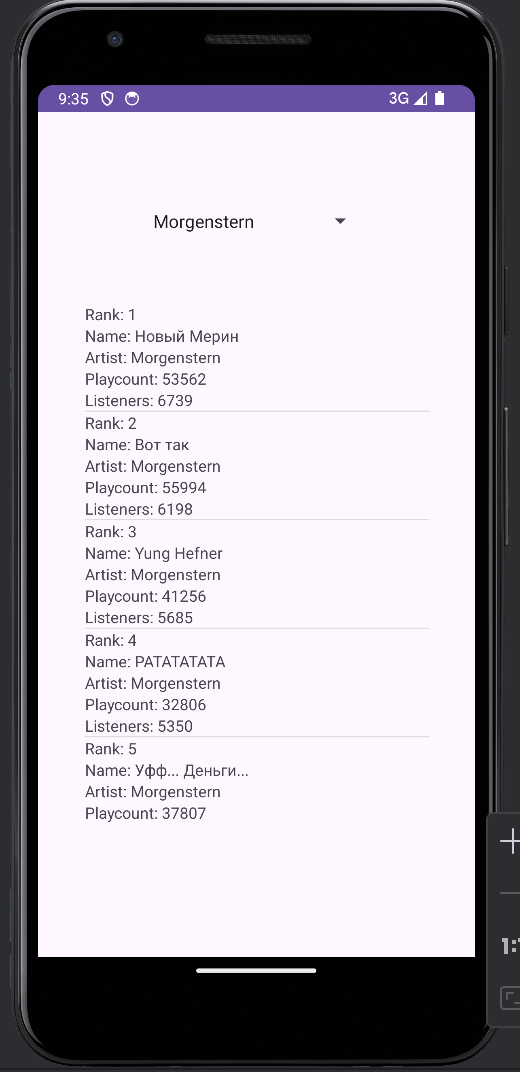












Какие концепции реализации многопоточности в Android можем использовать?

**Ответ**: В Android существует несколько концепций для реализации многопоточности. Вот некоторые из них:

1. Thread (Потоки): Это самый базовый механизм многопоточности в Android. Вы можете создать экземпляр класса Thread и выполнить в нем некоторую задачу. Однако использование непосредственно потоков может быть сложным и требует внимания к механизмам синхронизации.
2. AsyncTask: AsyncTask - это удобный способ выполнения асинхронных операций в фоновом потоке с возможностью обновления пользовательского интерфейса. Он предоставляет методы, позволяющие выполнять задачи в фоновом режиме и взаимодействовать с пользовательским интерфейсом в основном потоке.
3. Handler и Looper: Handler и Looper позволяют вам отправлять и обрабатывать сообщения между потоками. Вы можете создать свой собственный поток и использовать Handler для отправки сообщений из других потоков или выполнения задач в фоновом режиме.
4. Executors и ThreadPool: Executors предоставляют высокоуровневый интерфейс для управления пулом потоков. Вы можете использовать ThreadPoolExecutor для создания пула потоков определенного размера и выполнения асинхронных задач в этих потоках.
5. Kotlin Coroutines: Kotlin Coroutines - это новый подход к конкурентному программированию, предоставляемый языком Kotlin. Он предоставляет легковесные потоки выполнения, которые можно использовать для асинхронных операций. Coroutines позволяют писать асинхронный код в синхронном стиле, делая его более понятным и легким в использовании.
6. RxJava: RxJava - это библиотека для реактивного программирования. Она предоставляет возможности для работы с асинхронными потоками данных и управления обработкой событий. RxJava основана на наблюдаемых (Observable) и подписчиках (Subscriber) и предоставляет мощные операторы для манипуляции и комбинирования потоков данных.

2. Для каких основных компонент может применять асинхронную обработку и многопоточность и с помощью каких подходов ( привести примеры классов, библиотек и т. д.)?

**Ответ**: для фоновых задач или задач, требующих долгих вычислений используется многопоточность, чтобы главный поток не зависал и делал опыт использования приложения лучше