Лабораторная работа № 7. Разработка мобильного приложения с поддержкой многопоточности и сетевым взаимодействием

Цель лабораторной работы:

- Изучить работу с потоками.
- Изучить получить данных по сети.
- Изучить реализацию многопоточности в java.util.concurrent и Kotlin Coroutines.

Задачи лабораторной работы:

- Разработать приложение с подключением к сервису LastFM и получением сведений об артистах и композициях.

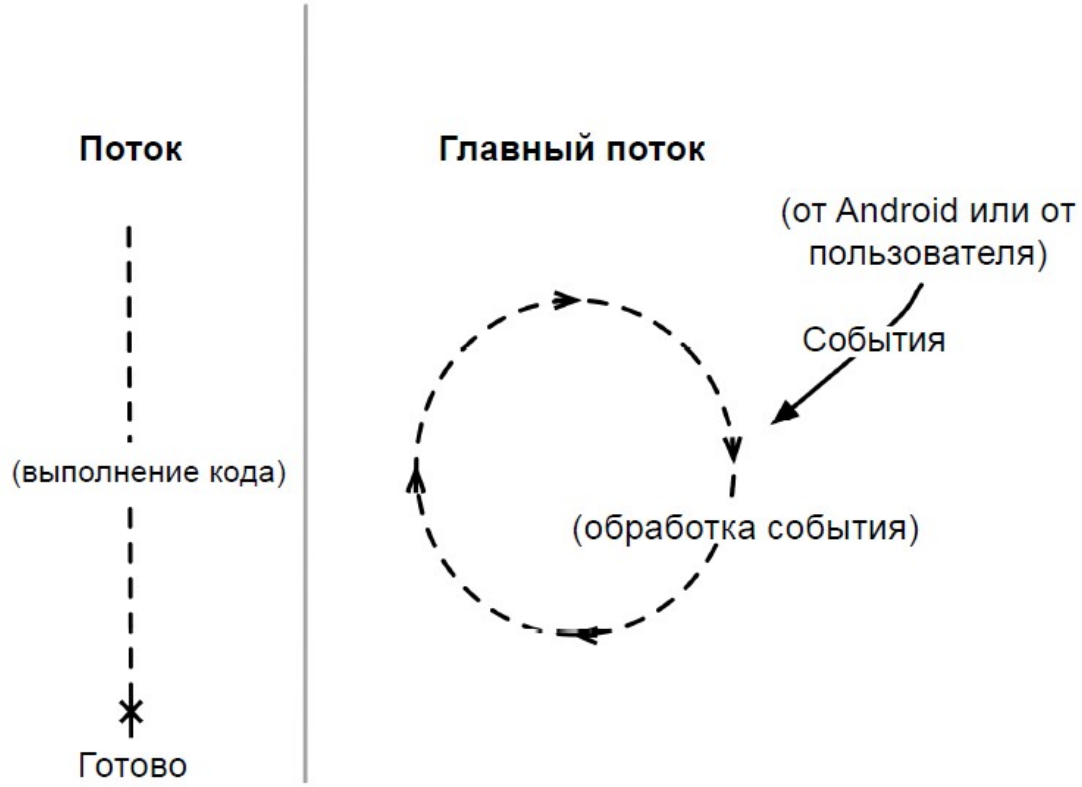
Table of Contents

7.1 Параллелизм в Android

Теоретические сведения

Познакомьтесь с документацией по процессам и потокам — https://developer.android.com/guide/components/processes-and-threads?hl=ru и асинхронной работе — https://developer.android.com/develop/background-work/background-tasks/asynchronous

Главный и обычный потоки



10.1. Графическое представление потоков

Асинхронные потоки в Android

AsyncTask (асинхронная задача) в Android — это абстрактный класс или, вспомогательный класс, который позволяет приложению выполнять ресурсоемкие задачи в фоновом режиме и параллельно выполнять изменения пользовательского интерфейса во внешнем интерфейсе. Фоновые задачи, такие как загрузка изображений, загрузка музыки, изображений, файлов и т. д., можно выполнять с помощью AsyncTask. Однако AsyncTask переведен в разряд устаревших.

Вместо данного классам можем использовать другие методы, чтобы выполнять фоновые задачи и одновременно обновлять элементы пользовательского интерфейса. К ним относятся:

- Использование комбинации Executor и Handler.
- Использование потока.

Изучить статьи и документацию:

1. Alternatives for the Deprecated AsyncTask in Android (https://www.geeksforgeeks.org/alternatives-for-the-deprecated-asynctask-in-android/) (на русском языке - https://progler.ru/blog/alternativy-ustarevshey-asynctask-v-android)
2. https://developer.android.com/guide/background/threading
3. https://guides.codepath.com/android/Managing-Threads-and-Custom-Services#learn-more-about-services

Критерии оценивания

4 — приложения на основе упражнений 1-2 на Java и ответы на контрольные вопросы;

5-6 — приложения на основе упражнений 1-3 и ответы на контрольные вопросы;

7-8 — приложения на основе упражнений 1-3 и задание 1 на Java и ответы на контрольные вопросы;

9 — приложения на основе упражнений 1-3 и задание 1 на Kotlin из лабораторной работы, дополнительный функционал и ответы на контрольные вопросы.

Задания для самостоятельной работы

Для реализации концепции параллельного программирования, начиная с API 30 рекомендуется использовать для проекта на java классы из пакета java.util.concurrent (https://developer.android.com/reference/java/util/concurrent/package-summary), так как класс AsyncTask помечен как deprecated (https://developer.android.com/reference/android/os/AsyncTask?hl=ru). В случае реализации проекта на Kotlin рекомендуется познакомиться с концепцией корутин (https://developer.android.com/topic/libraries/architecture/coroutines?hl=ru). При реализации самостоятельных заданий рекомендуются учитывать версию API и в зависимости от выбранного API использовать AsyncTask или классы из пакета java.util.concurrent для проекта на Java и корутины для проекта на Kotlin.

Упражнение 1

Изучить примеры реализации многопоточности https://www.geeksforgeeks.org/multithreading-in-android-with-examples/ на Kotlin и Java.

Упражнение 2

1. Изучить и реализовать пример использования потоков из https://metanit.com/java/android/10.1.php.

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

Упражнение 3

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

2. Изучить пример из учебного курса Use Kotlin Coroutines in your Android App.

Задание 1

Изучить из Android Apprentice пример реализации приложения для подкастов главы 20-27 (стр. 455 – 630).

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.

Контрольные вопросы

1. Какие концепции реализации многопоточности в Andoid можем использовать?
2. Для каких основных компонент может применять асинхронную обработку и многопоточность и с помощью каких подходов (привести примеры классов, библиотек и т. д.)?