Отчет за период 25.03-06.04
Запланировано задач к текущему дню: 7
Выполнено: 8
Всего выполнено: 20/36.
Общий процент выполнения плана: 55%
Остается недель до окончания: 3

Кратко, к этому дню пройден курс по RxJava, изучены все дополнительные статьи. Остается прочтение книги Reactive Design Patterns.

День 27.03
Прошел восьмой урок. Поработал с операторами преобразования: узнал, как работает zip, flatMap. Также узнал о существовании параметра maxConcurrent, который определяет максимальное количество потоков, на которых могут исполняться Observables.

День 28.03
Прошел девятый урок. В нем просто рассказывалось, как из популярной в Android библиотеке Retrofit получать данные в виде Observable. Ничего особенного, просто есть библиотека, оборачивающая сетевые вызовы в один из реактивных типов, которые эмитят данные.

День 29.03
Прошел десятый урок. В нем разбиралась тема Backpressure. По-сути, это такая стратегия, описывающая поведение, когда подписчик не успевает обработать данные от источника. То есть когда данные эмитется чаще, чем подписчик успевает их обрабатывать.
В общем, если подписчик и источник работают в одном потоке, источнику необходимо ожидать, пока подписчик обработает значение, прежде чем отсылать следующее значение. Если переключить подписчика на другой поток (observeOn), то источник освобождается от ожидания и может вернуть все значения, какие только может. Эти данные сохранятся в буффер, который есть у observeOn и подписчик, обрабатывая данные в своем темпе, будет получать следующие уже из буфера. Сам буфер по-умолчанию ограничен 16-ю элементами. После того, как буфер достаточно очистился, observeOn может снова его наполнить данными из источника. Вот эта возможность запросить данные и есть **backpressure.** Это описывался оператор range, который поддерживает механизм backpressure.

День 30.03
Прошел одиннадцатый урок. Это переходный урок на RxJava V2, здесь приводится различия. Также пробежался по короткому двенадцатому уроку. В нем просто

упоминается андроидовская библиотека для обращения к UI-элементам в парадигме Rx. Это уже старая технология, ничего полезного тут.

В RxJava источников данных может быть 5, помимо рассмотренных Observable и Flowable это: Single, Observable, Flowable, Completable, Maybe.

Отличия.

- Single возвращает 1 элемент в onNext или ошибку в onError. OnComplete отсутствует.
- Completable. Предоставляет либо onComplete, либо onError. Данные в onNext не передает.
- Maybe. Возвращает одно значение либо в onNext, либо onComplete. Но не оба. Также может прийти ошибка в onError.

День 02.04

Освежил в памяти статью про описание реактивного программирования в функциональном стиле. Укрепил понимание стратегий взаимодействия подписчика и источника данных (Push-Pull-Push and Pull). Также углубился в связную тему функционального программирования обобщенно от реактивного программирования.

День 06.04

К этому моменту закончил курс по RxJava. Прочитал статью AdvancedRxJava with stream nesting. В статье описывался только Backpressure и как с ним жить, но написано интересно и достаточно глубоко погружает внутрь этого механизма.