

УДК 004.738.5:004.438 RxJava
ББК 32.973.2
M15

M15 Томаш Нуркевич, Бен Кристенсен

Реактивное программирование с применением RxJava / пер. с англ. Слинкин А. А. – М.: ДМК Пресс, 2017. – 358 с.: ил.

ISBN 978-5-97060-496-0

В наши дни, когда программы асинхронны, а быстрая реакция – важнейшее свойство, реактивное программирование поможет писать более надежный, лучше масштабируемый и быстрее работающий код. Благодаря этой книге программист на Java узнает о реактивном подходе к задачам и научится создавать программы, вобравшие в себя лучшие черты этой новой и весьма перспективной парадигмы. Данная книга содержит глубокое и подробное изложение концепций и принципов использования реактивного программирования вообще и RxJava в частности.

Книга может использоваться как для последовательного изучения предмета, так и в качестве справочника по библиотеке.

УДК 004.738.5:004.438 RxJava
 ББК 32.973.2

Authorized Russian translation of the English edition of Reactive Programming with RxJava, ISBN 9781491931653. © 2017 Ben Christensen and Tomasz Nurkiewicz. This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to publish and sell the same.

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

Материал, изложенный в данной книге, многократно проверен. Но, поскольку вероятность технических ошибок все равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

ISBN 978-1-49193-165-3 (англ.)
 ISBN 978-5-97060-496-0 (рус.)

© Ben Christensen and Tomasz Nurkiewicz, 2017.
 © Перевод на русский язык, оформление,
 издание, ДМК Пресс, 2017



Оглавление

Предисловие	11
Вступление	14
Для кого написана эта книга	14
Несколько слов от Бена Кристенсена.....	14
Несколько слов от Томаша Нуркевича	16
О содержании книги	16
Ресурсы в Сети	17
Графические выделения	17
Как с нами связаться	18
Благодарности	18
От Бена	18
От Томаша.....	19
Глава 1. Реактивное программирование с применением RxJava	20
Реактивное программирование и RxJava	20
Когда возникает нужда в реактивном программировании.....	22
Как работает RxJava.....	23
Проталкивание и вытягивание.....	23
Синхронный и асинхронный режим	25
Конкурентность и параллелизм.....	28
Ленивые и энергичные типы.....	31
Двойственность	33
Одно или несколько?.....	34
Учет особенностей оборудования: блокирующий и неблокирующий ввод-вывод.....	39
Абстракция реактивности	44
Глава 2. Реактивные расширения	45
Анатомия rx.Observable.....	45
Подписка на уведомления от объекта Observable.....	48
Получение всех уведомлений с помощью типа Observer<T>	50
Управление прослушивателями с помощью типов Subscription и Subscriber<T>	50



Создание объектов Observable	52
Подробнее о методе Observable.create()	53
Бесконечные потоки	56
Хронометраж: операторы timer() и interval()	61
Горячие и холодные объекты Observable	61
Практический пример: от API обратных вызовов к потоку Observable	63
Управление подписчиками вручную	68
Класс rx.subjects.Subject	69
Тип ConnectableObservable	71
Реализация единственной подписки с помощью publish().refCount()	72
Жизненный цикл ConnectableObservable	74
Резюме	77

Глава 3. Операторы и преобразования 79

Базовые операторы: отображение и фильтрация	79
Взаимно однозначные преобразования с помощью map()	81
Обертывание с помощью flatMap()	85
Откладывание событий с помощью оператора delay()	90
Порядок событий после flatMap()	91
Сохранение порядка с помощью concatMap()	93
Более одного объекта Observable	95
Обращение с несколькими объектами Observable, как с одним, с помощью merge()	95
Попарная композиция с помощью zip() и zipWith()	97
Когда потоки не синхронизированы: combineLatest(), withLatestFrom() и amb()	100
Более сложные операторы: collect(), reduce(), scan(), distinct() и groupBy()	106
Просмотр последовательности с помощью Scan и Reduce	106
Редукция с помощью изменяемого аккумулятора: collect()	108
Проверка того, что Observable содержит ровно один элемент, с помощью single()	109
Устранение дубликатов с помощью distinct() и distinctUntilChanged()....	110
Выборка с помощью операторов skip(), takeWhile() и прочих	112
Способы комбинирования потоков: concat(), merge() и switchOnNext()	114
Расщепление потока по условию с помощью groupBy()	121
Написание пользовательских операторов	125
Повторное использование операторов с помощью compose()	125
Реализация более сложных операторов с помощью lift()	127
Резюме	133

Глава 4. Применение реактивного программирования в существующих приложениях 134

От коллекций к Observable	135
BlockingObservable: выход из реактивного мира	135

О пользе лени	138
Композиция объектов Observable	140
Ленивое разбиение на страницы и конкатенация	141
Императивная конкурентность	142
flatMap() как оператор асинхронного сцепления	148
Замена обратных вызовов потоками	153
Периодический опрос изменений	156
Многопоточность в RxJava	157
Что такое диспетчер?	158
Декларативная подписка с помощью subscribeOn()	167
Конкурентность и поведение subscribeOn()	171
Создание пакета запросов с помощью groupBy()	175
Декларативная конкурентность с помощью observeOn()	177
Другие применения диспетчеров	180
Резюме	181
Глава 5. Реактивность сверху донизу	183
Решение проблемы C10k	183
Традиционные HTTP-серверы на основе потоков	185
Неблокирующий HTTP-сервер на основе Netty и RxNetty	187
Сравнение производительности блокирующего и реактивного сервера	195
Обзор реактивных HTTP-серверов	200
Код HTTP-клиента	201
Доступ к реляционной базе данных	205
NOTIFY и LISTEN на примере PostgreSQL	207
CompletableFuture и потоки	211
Краткое введение в CompletableFuture	211
Сравнение типов Observable и Single	220
Создание и потребление объектов типа Single	221
Объединение ответов с помощью zip, merge и concat	223
Интероперабельность с Observable и CompletableFuture	225
Когда использовать тип Single?	226
Резюме	227
Глава 6. Управление потоком и противодавление	228
Управление потоком	228
Периодическая выборка и отбрасывание событий	228
Скользящее окно	237
Пропуск устаревших событий с помощью debounce()	238
Противодавление	243
Противодавление в RxJava	244
Встроенное противодавление	247
Производители и исключение MissingBackpressureException	250
Учет запрошенного объема данных	253
Резюме	259

Глава 7. Тестирование и отладка	260
Обработка ошибок.....	260
А где же мои исключения?	261
Декларативная замена try-catch	264
Таймаут в случае отсутствия событий.....	268
Повтор после ошибки.....	271
Тестирование и отладка	275
Виртуальное время	275
Диспетчеры и автономное тестирование	277
Автономное тестирование.....	279
Мониторинг и отладка.....	287
Обратные вызовы doOn...()	287
Измерение и мониторинг	289
Резюме.....	292
Глава 8. Практические примеры	293
Применение RxJava в разработке программ для Android.....	293
Предотвращение утечек памяти в компонентах Activity.....	294
Библиотека Retrofit со встроенной поддержкой RxJava	296
Диспетчеры в Android	301
События пользовательского интерфейса как потоки	304
Управление отказами с помощью Hystrix.....	307
Hystrix: первые шаги.....	308
Неблокирующие команды и HystrixObservableCommand.....	310
Паттерн Переборка и быстрое прекращение	311
Пакетирование и объединение команд	313
Мониторинг и инструментальные панели	318
Опрос баз данных NoSQL.....	321
Клиентский API Couchbase	322
Клиентский API MongoDB	323
Интеграция с Camel	325
Потребление файлов с помощью Camel	325
Получение сообщений от Kafka	326
Потоки Java 8 и CompletableFuture	326
Полезность параллельных потоков.....	328
Выбор подходящей абстракции конкурентности	330
Когда выбирать Observable?	331
Потребление памяти и утечки	332
Операторы, потребляющие неконтролируемый объем памяти.....	332
Резюме	337
Глава 9. Направления будущего развития	338
Реактивные потоки	338
Типы Observable и Flowable	338
Производительность	339

Миграция.....	340
Приложение А. Дополнительные примеры	
HTTP-серверов.....	341
Системный вызов fork() в программе на С	341
Один поток – одно подключение	343
Пул потоков для обслуживания подключений	345
Приложение В. Решающее дерево для выбора	
операторов Observable	347
Об авторах	352
Об изображении на обложке	352
Предметный указатель	353