

УДК 004.432
ББК 32.972.1
Д63

Д63 Олег Докука, Игорь Лозинский
Практика реактивного программирования в Spring 5. –
М.: ДМК Пресс, 2019. – 508 с.

ISBN 978-5-97060-747-3

Данная книга посвящена реактивному программированию в Spring. Описаны многочисленные возможности построения эффективных реактивных систем с помощью Spring 5 и других инструментов, таких как WebFlux, Spring Boot и Project Reactor. Приведены методы реактивного программирования и их использование для взаимодействий с базами данных и между серверами. Рассмотрено создание независимых и высокопроизводительных микросервисов с помощью Spring Cloud Streams.

Издание предназначено разработчикам на Java, использующим фреймворк Spring для своих задач и желающим научиться создавать надежные и реактивные приложения, способные автоматически масштабироваться в облаке.

УДК 004.432
ББК 32.972.1

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 978-5-97060-747-3 (рус.) © Олег Докука, Игорь Лозинский, 2018
ISBN 978-1-78728-495-1 (анг.) © Оформление, издание, ДМК Пресс, 2019

Оглавление

Предисловие	7
Глава 1. Причины выбора Spring	22
Основные преимущества реактивности	22
Взаимодействия на основе обмена сообщениями	25
Примеры использования реактивности	30
Причины добавления поддержки реактивности в Spring	33
Реактивность на уровне служб	34
Заключение	42
Глава 2. Реактивное программирование в Spring. Основные понятия	44
Первые реактивные решения в Spring	44
Шаблон «Наблюдатель»	45
Примеры использования шаблона «Наблюдатель»	49
Шаблон «Публикация/Подписка» с использованием @EventListener	52
Создание приложений с @EventListener	54
Создание приложения на основе Spring	54
Реализация бизнес-логики	55
Асинхронные взаимодействия по HTTP с помощью Spring Web MVC	57
Публикация конечной точки SSE	57
Настройка поддержки асинхронного выполнения	59
Создание пользовательского интерфейса с поддержкой SSE	60
Проверка приложения	61
Критический обзор решения	61
RxJava как реактивный фреймворк	62
«Наблюдатель» плюс «Итератор» равно «реактивный поток»	63
Производство и потребление потоков	65
Генерация последовательности асинхронных событий	68
Преобразование потоков и диаграммы Marble	69
Оператор map	69
Оператор filter	70
Оператор count	71
Оператор zip	71
Требования и преимущества RxJava	72
Переделка приложения с RxJava	75
Реализация бизнес-логики	75
Нестандартный SseEmitter	77
Публикация конечной точки SSE	78
Конфигурация приложения	79
Краткая история развития реактивных библиотек	80
Реактивный ландшафт	82
Заключение	84
Глава 3. Reactive Streams – новый стандарт потоков	85
Реактивность для всех	85
Проблема несовместимости API	86

Модели обмена PULL и PUSH	89
Проблема управления потоком данных	95
Медленный производитель и быстрый потребитель	95
Быстрый производитель и медленный потребитель	96
Неограниченная очередь	96
Ограниченная очередь со сбросом избыточных элементов	97
Ограниченная очередь с блокировкой	97
Решение	99
Основные положения стандарта Reactive Streams	99
Требования Reactive Streams в действии	106
Введение в понятие обработчика Processor	109
Проверка совместимости с Reactive Streams	113
Проверка издателя Publisher	115
Проверка подписчика Subscriber	117
JDK 9	121
Асинхронный и параллельный API в Reactive Streams	123
Преобразование реактивного ландшафта	125
Изменения в RxJava	125
Изменения в Vert.x	129
Усовершенствования в Ratpack	130
Драйвер MongoDB с поддержкой Reactive Streams	131
Комбинирование реактивных технологий на практике	132
Заключение	135
Глава 4. Project Reactor – основа реактивных приложений	137
Краткая история Project Reactor	137
Project Reactor 1.x	138
Project Reactor 2.x	141
Основы Project Reactor	142
Добавление библиотеки Reactor в проект	144
Реактивные типы: Flux и Mono	145
Flux	145
Mono	147
Реактивные типы из RxJava 2	148
Observable	148
Flowable	148
Single	148
Maybe	149
Completable	149
Создание последовательностей Flux и Mono	149
Подписка на реактивный поток	151
Реализация своих подписчиков	154
Преобразование реактивных последовательностей с помощью операторов	156
Отображение элементов реактивных последовательностей	157
Фильтрация реактивных последовательностей	158
Сбор данных из реактивных последовательностей	160

Сокращение элементов потока.....	161
Комбинирование реактивных потоков.....	164
Пакетная обработка элементов потока.....	164
Операторы flatMap, concatMap и flatMapSequential.....	168
Извлечение выборки элементов.....	170
Преобразование реактивных последовательностей в блокирующие структуры.....	170
Просмотр элементов при обработке последовательности.....	171
Материализация и дематериализация сигналов.....	172
Поиск подходящего оператора.....	173
Создание потоков данных программным способом.....	173
Фабричные методы push и create.....	173
Фабричный метод generate.....	174
Передача одноразовых ресурсов в реактивные потоки.....	175
Обертывание транзакций с помощью фабричного метода usingWhen.....	178
Обработка ошибок.....	180
Управление обратным давлением.....	183
Горячие и холодные потоки данных.....	184
Широковещательная рассылка элементов потока данных.....	185
Кеширование элементов потока.....	186
Совместное использование элементов из потока.....	187
Работа со временем.....	188
Компоновка и преобразование реактивных потоков.....	188
Процессоры.....	190
Тестирование и отладка Project Reactor.....	191
Дополнения к Reactor.....	192
Продвинутое средства в Project Reactor.....	193
Жизненный цикл реактивных потоков данных.....	193
Этап сборки.....	193
Этап подписки.....	195
Выполнение.....	196
Модель планирования потоков выполнения в Reactor.....	199
Оператор publishOn.....	199
Параллельная обработка с помощью publishOn.....	201
Оператор subscribeOn.....	202
Оператор parallel.....	204
Планировщик.....	204
Контекст.....	205
Особенности внутренней реализации Project Reactor.....	209
Макрослияние.....	210
Микрослияние.....	211
Заключение.....	214
Глава 5. Добавление реактивности с помощью Spring Boot 2.....	216
Быстрый старт как ключ к успеху.....	217
Использование Spring Roo для ускорения разработки приложений.....	219
Spring Boot как ключ к созданию быстро растущих приложений.....	219

Реактивность в Spring Boot 2.0	220
Реактивность в Spring Core	221
Поддержка преобразования реактивных типов	221
Реактивный ввод/вывод	222
Реактивность в Web	224
Реактивность в Spring Data	226
Реактивность в Spring Session	227
Реактивность в Spring Security	228
Реактивность в Spring Cloud	228
Реактивность в Spring Test	229
Реактивность в мониторинге	229
Заключение	230
Глава 6. Неблокирующие и асинхронные взаимодействия с WebFlux	231
WebFlux как основа реактивного сервера	231
Реактивное веб-ядро	234
Реактивные фреймворки Web и MVC	238
Чисто функциональные приемы в WebFlux	242
Неблокирующие взаимодействия между службами с WebClient	246
Реактивный WebSocket API	249
Серверный WebSocket API	250
Клиентский WebSocket API	251
Сравнение WebFlux WebSocket и Spring WebSocket	252
Реактивный поток SSE и легковесная замена WebSocket	253
Реактивные механизмы шаблонов	255
Реактивная безопасность	258
Реактивный доступ к SecurityContext	258
Использование реактивной безопасности	261
Взаимодействия с другими реактивными библиотеками	262
Сравнение WebFlux и Web MVC	263
Законы сравнения фреймворков	264
Закон Литтла	264
Закон Амдала	265
Универсальный закон масштабируемости	269
Анализ и сравнение	272
Модели обработки в WebFlux и Web MVC	272
Влияние моделей обработки на пропускную способность и задержку	274
Проблемы модели обработки в WebFlux	282
Потребление памяти разными моделями обработки	285
Влияние модели обработки на удобство	291
Практическое применение WebFlux	292
Системы на основе микросервисов	292
Системы, обслуживающие клиентов с медленными соединениями	294
Потоковые системы или системы реального времени	294
WebFlux в действии	295
Заключение	299

Глава 7. Реактивный доступ к базам данных	301
Модели обработки данных в современном мире	302
Предметно-ориентированное проектирование	302
Хранение данных в эпоху микросервисов	303
Использование хранилищ разного типа	306
База данных как услуга	307
Разделение данных между микросервисами	309
Распределенные транзакции	310
Событийно-ориентированные архитектуры	310
Согласованность в конечном счете	311
Шаблон SAGA	312
Регистрация событий	312
Разделение ответственности на команды и запросы	313
Бесконфликтно реплицируемые типы данных	314
Система обмена сообщениями как хранилище данных	315
Синхронная модель извлечения данных	316
Протокол связи для доступа к базе данных	316
Драйвер базы данных	318
JDBC	319
Управление соединениями	320
Реактивный доступ к базе данных	321
Spring JDBC	322
Spring Data JDBC	323
Добавление реактивности в Spring Data JDBC	326
JPA	326
Добавление реактивности в JPA	327
Spring Data JPA	327
Добавление реактивности в Spring Data JPA	328
Spring Data NoSQL	329
Ограничения синхронной модели	332
Достоинства синхронной модели	333
Реактивный доступ к данным с использованием Spring Data	334
Реактивное хранилище на основе MongoDB	336
Объединение операций с хранилищем	339
Как работают реактивные хранилища	344
Поддержка разбиения на страницы	345
Детали реализации ReactiveMongoRepository	345
Использование ReactiveMongoTemplate	346
Использование реактивных драйверов (MongoDB)	348
Использование асинхронных драйверов (Cassandra)	350
Реактивные транзакции	352
Реактивные транзакции в MongoDB 4	352
Распределенные транзакции с шаблоном SAGA	361
Реактивные коннекторы в Spring Data	361
Реактивный коннектор MongoDB	361
Реактивный коннектор Cassandra	362

Реактивный коннектор Couchbase	362
Реактивный коннектор Redis	363
Ограничения и ожидаемые улучшения	364
Асинхронный доступ к базам данных	365
Реактивное соединение с реляционной базой данных	367
Использование R2DBC вместе с Spring Data R2DBC	369
Преобразование синхронного хранилища в реактивное	370
С помощью библиотеки gxjava2-jdbc	371
Обертывание синхронного CrudRepository	373
Реактивный Spring Data в действии	378
Заключение	382
Глава 8. Масштабирование с Cloud Streams	383
Брокеры сообщений как основа систем, управляемых сообщениями	384
Балансировка нагрузки на стороне сервера	384
Балансировка нагрузки на стороне клиента с Spring Cloud и Ribbon	386
Брокеры сообщений как эластичный и надежный слой для передачи сообщений	392
Рынок брокеров сообщений	396
Spring Cloud Streams как мост в экосистему Spring	397
Реактивное программирование в облаке	406
Spring Cloud Data Flow	407
Модульная организация приложений с Spring Cloud Function	409
Spring Cloud – функция как часть конвейера обработки данных	416
RSocket для реактивной передачи сообщений с низкой задержкой	420
RSocket и Reactor-Netty	421
RSocket в Java	425
RSocket и gRPC	428
RSocket в Spring Framework	430
RSocket в других фреймворках	432
Проект ScaleCube	432
Проект Proteus	433
В заключение о RSocket	433
Заключение	433
Глава 9. Тестирование реактивных приложений	435
Почему реактивные потоки данных сложно тестировать?	435
Тестирование реактивных потоков с помощью StepVerifier	436
Основы StepVerifier	436
Продвинутые приемы тестирования с использованием StepVerifier	440
Виртуальное время	442
Проверка реактивного контекста	445
Тестирование WebFlux	445
Тестирование контроллеров с помощью WebTestClient	446
Тестирование WebSocket	451

Заключение	455
Глава 10. И наконец, выпуск!	456
Важность поддержки идеологии DevOps в приложениях	456
Мониторинг реактивных Spring-приложений	460
Spring Boot Actuator	460
Добавление механизма мониторинга в проект	460
Конечная точка для получения информации о службе	461
Конечная точка для получения информации о работоспособности	463
Конечная точка для получения информации о параметрах работы	466
Конечная точка управления журналированием	467
Другие важные конечные точки	468
Реализация своей конечной точки для Actuator	469
Безопасность конечных точек	470
Micrometer	472
Параметры по умолчанию в Spring Boot	473
Мониторинг реактивных потоков данных	474
Мониторинг потоков в Reactor	474
Мониторинг планировщиков в Reactor	475
Реализация своих параметров Micrometer	477
Распределенная трассировка с Spring Boot Sleuth	478
Пользовательский интерфейс Spring Boot Admin 2.x	480
Развертывание в облаке	482
Развертывание в Amazon Web Services	485
Развертывание в Google Kubernetes Engine	486
Развертывание в Pivotal Cloud Foundry	487
Обнаружение RabbitMQ в PCF	488
Обнаружение MongoDB в PCF	489
Развертывание в PCF без конфигурации с помощью Spring Cloud Data Flow	491
Knative для FaaS на основе Kubernetes и Istio	491
Советы по успешному развертыванию приложений	492
Заключение	493
Указатель	495