

УДК 004.432
ББК 32.972.1
К21

Джон Карнелл, Иллари Уайлупо Санчес

К21 Микросервисы Spring в действии / пер. с англ. А. Н. Киселева. – М.: ДМК Пресс, 2022. – 490 с.: ил.

ISBN 978-5-97060-971-2

В книге рассказывается о том, как создавать приложения на основе микросервисов с использованием Java и Spring. Описываются особенности управления конфигурацией микросервисов и передовые практики их разработки. Уделено внимание защите потребителей, когда один или несколько экземпляров микросервисов выходят из строя.

Начав с создания простых служб, читатель постепенно перейдет к знакомству с приемами эффективного журналирования и мониторинга, научится реструктурировать приложения на Java с помощью инструментов Spring, освоит управление API с помощью Spring Cloud Gateway.

Издание предназначено для разработчиков на Java, имеющим опыт создания распределенных приложений и использования Spring, а также всем, кому интересно узнать, что необходимо для развертывания приложения на основе микросервисов в облаке.

УДК 004.432
ББК 32.972.1

Copyright Original English language edition published by Manning Publications USA, USA. Copyright (c) 2021 by Manning Publications. Russian-language edition copyright (c) 2021 by DMK Press. All rights reserved.

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.

ISBN (анг.) 978-1-617-29695-6
ISBN (рус.) 978-5-97060-971-2

© 2021 by Manning Publications Co.
© Оформление, издание, перевод,
ДМК Пресс, 2022

Содержание

<i>Предисловие от издательства</i>	16
<i>Предисловие</i>	17
<i>Благодарности</i>	19
<i>Об этой книге</i>	21
 1 <i>Добро пожаловать в Spring Cloud</i>	27
1.1. Эволюция архитектуры микросервисов	28
1.1.1. N-уровневая архитектура	28
1.1.2. Что такое монолитная архитектура?	29
1.1.3. Что такое микросервис?	30
1.1.4. Зачем менять способ создания приложений?	32
1.2. Микросервисы со Spring	34
1.3. Что мы будем создавать?	36
1.4. О чем эта книга?	37
1.4.1. Что вы узнаете в этой книге	37
1.4.2. Почему эта книга актуальна для вас?	38
1.5. Облачные приложения и приложения на основе микросервисов	39
1.5.1. Создание микросервиса с помощью Spring Boot	39
1.5.2. Что такое облачные вычисления?	44
1.5.3. В чем преимущества облачных вычислений и микросервисов?	46
1.6. Микросервисы – это больше чем код	49
1.7. Базовый шаблон разработки микросервисов	50
1.8. Шаблоны маршрутизации	52
1.9. Устойчивость клиентов	54

1.10. Шаблоны безопасности	55
1.11. Шаблоны журналирования и трассировки	56
1.12. Шаблон сбора метрик приложения	58
1.13. Шаблоны сборки/развертывания микросервисов	59
Итоги	61

2

Обзор мира микросервисов через призму Spring Cloud..... 63

2.1. Что такое Spring Cloud?	64
2.1.1. <i>Spring Cloud Config</i>	65
2.1.2. <i>Spring Cloud Service Discovery</i>	66
2.1.3. <i>Spring Cloud LoadBalancer и Resilience4j</i>	66
2.1.4. <i>Spring Cloud API Gateway</i>	67
2.1.5. <i>Spring Cloud Stream</i>	67
2.1.6. <i>Spring Cloud Sleuth</i>	67
2.1.7. <i>Spring Cloud Security</i>	68
2.2. Пример использования Spring Cloud	68
2.3. Приемы создания облачных микросервисов	71
2.3.1. База кода.....	74
2.3.2. Зависимости.....	75
2.3.3. Конфигурация.....	76
2.3.4. Вспомогательные службы.....	76
2.3.5. Сборка, выпуск, выполнение.....	77
2.3.6. Процессы	78
2.3.7. Привязка портов	78
2.3.8. Масштабируемость	79
2.3.9. Одноразовость.....	79
2.3.10. Сходство окружений разработки/эксплуатации	80
2.3.11. Журналирование	80
2.3.12. Задачи администрирования	81
2.4. Актуальность наших примеров.....	81
2.5. Создание микросервиса с использованием Spring Boot и Java	82
2.5.1. Подготовка окружения	83
2.5.2. Начало создания проекта	83
2.5.3. Запуск приложения <i>Spring Boot</i> : класс инициализации	88
Итоги	90

3

Создание микросервисов с использованием Spring Boot..... 91

3.1. Точка зрения архитектора: проектирование микросервисной архитектуры.....	92
3.1.1. Декомпозиция бизнес-задачи	92

3.1.2. Детализация служб	95
3.1.3. Определение интерфейсов служб	98
3.2. Когда не следует использовать микросервисы	99
3.2.1. Сложность распределенных систем	99
3.2.2. Беспорядочный рост виртуальных серверов или контейнеров	99
3.2.3. Тип приложения	100
3.2.4. Транзакции и согласованность данных	100
3.3. Точка зрения разработчика: создание микросервиса с использованием Spring Boot и Java	100
3.3.1. Встраивание дверного проема в микросервис: контроллер Spring Boot	101
3.3.2. Добавление интернационализации в службу лицензий	112
3.3.3. Реализация Spring HATEOAS для отображения связанных ссылок	115
3.4. Точка зрения инженера DevOps: сборка выполняемых артефактов	118
3.4.1. Сборка службы: упаковка и развертывание микросервисов	120
3.4.2. Инициализация службы: управление конфигурацией микросервисов	122
3.4.3. Регистрация и обнаружение службы: взаимодействие клиентов с микросервисами	123
3.4.4. Мониторинг состояния микросервиса	124
3.5. Объединение точек зрения	127
Итоги	128

4	<i>Добро пожаловать в Docker</i>	129
4.1.	Контейнеры или виртуальные машины?	130
4.2.	Что такое Docker?	132
4.3.	Файлы Dockerfile	135
4.4.	Docker Compose	136
4.5.	Интеграция Docker с микросервисами	138
4.5.1.	Создание образа Docker	138
4.5.2.	Создание образов Docker со Spring Boot	144
4.5.3.	Запуск служб с помощью Docker Compose	147
	Итоги	148

5	<i>Управление конфигурациями с использованием Spring Cloud Configuration Server</i>	150
5.1.	Об управлении конфигурациями (и сложностью)	151
5.1.1.	Архитектура управления конфигурацией	152

5.1.2. Варианты реализации.....	154
5.2. Настройка Spring Cloud Configuration Server	156
5.2.1. Настройка класса инициализации Spring Cloud Config ...	161
5.2.2. Использование Spring Cloud Config Server с файловой системой	161
5.2.3. Создание конфигурационных файлов для службы	163
5.3. Интеграция Spring Cloud Config с клиентом Spring Boot.....	168
5.3.1. Настройка зависимостей Spring Cloud Config Service в службе лицензий.....	170
5.3.2. Настройка службы лицензий для взаимодействий с Spring Cloud Config.....	170
5.3.3. Подключение к источнику данных с использованием Spring Cloud Config Server	175
5.3.4. Чтение настроек с использованием @ConfigurationProperties.....	179
5.3.5. Обновление настроек с использованием Spring Cloud Config Server	180
5.3.6. Использование Spring Cloud Configuration Server с Git	182
5.3.7. Интеграция Vault со службой Spring Cloud Config.....	183
5.3.8. Пользовательский интерфейс Vault	184
5.4. Защита конфиденциальных настроек в конфигурации	187
5.4.1. Настройка симметричного шифрования.....	187
5.4.2. Шифрование и дешифрование настроек.....	188
5.5. Заключительные мысли	190
Итоги	190

6

Обнаружение служб.....	191
6.1. Где моя служба?	193
6.2. Обнаружение служб в облаке	195
6.2.1. Архитектура механизма обнаружения служб.....	196
6.2.2. Обнаружение служб с использованием Spring и Netflix Eureka.....	200
6.3. Создание службы Spring Eureka.....	202
6.4. Регистрация служб в Spring Eureka	207
6.4.1. REST API Eureka.....	211
6.4.2. Панель управления Eureka	212
6.5. Использование механизма обнаружения служб	214
6.5.1. Поиск экземпляров служб с Spring Discovery Client.....	216
6.5.2. Вызов служб с использованием шаблона Spring REST с поддержкой Load Balancer	218
6.5.3. Вызов служб с использованием Netflix Feign.....	220
Итоги	222

7	<i>Когда случаются неприятности: шаблоны устойчивости с использованием Spring Cloud и Resilience4j</i>	223
7.1.	Шаблоны устойчивости на стороне клиента	225
7.1.1.	Балансировка нагрузки на стороне клиента	226
7.1.2.	Размыкатель цепи	226
7.1.3.	Резервная реализация	227
7.1.4.	Герметичные отсеки	227
7.2.	Почему устойчивость клиента важна	228
7.3.	Реализация с Resilience4j	232
7.4.	Подготовка службы лицензий к использованию Spring Cloud и Resilience4j	233
7.5.	Реализация размыкателя цепи	234
7.5.1.	Добавление размыкателя цепи для обработки вызовов службы организаций	240
7.5.2.	Настройка размыкателя цепи	240
7.6.	Использование резервной реализации	241
7.7.	Реализация шаблона герметичных отсеков	244
7.8.	Реализация шаблона повторных попыток	248
7.9.	Реализация шаблона ограничителя частоты	249
7.10.	ThreadLocal и Resilience4j	252
	Итоги	257

8	<i>Маршрутизация служб с использованием Spring Cloud Gateway</i>	259
8.1.	Что такое сервисный шлюз?	260
8.2.	Введение в Spring Cloud Gateway	263
8.2.1.	Настройка проекта шлюза Spring Boot	264
8.2.2.	Настройка Spring Cloud Gateway для взаимодействий с Eureka	266
8.3.	Настройка маршрутов в Spring Cloud Gateway	268
8.3.1.	Автоматическое отображение маршрутов с помощью механизма обнаружения служб	268
8.3.2.	Отображение маршрутов вручную с помощью механизма обнаружения служб	270
8.3.3.	Динамическая загрузка настроек маршрутизации	273
8.4.	Настоящая мощь Spring Cloud Gateway: фабрики предикатов и фильтров	274
8.4.1.	Встроенные фабрики предикатов	275
8.4.2.	Встроенные фабрики фильтров	276
8.4.3.	Добавление своих фильтров	278

8.5. Создание предварительного фильтра	281
8.6. Использование идентификатора корреляции в службах...	284
8.6.1. <i>UserContextFilter: перехват входящих HTTP-запросов</i>	286
8.6.2. <i>UserContext: обеспечение доступности HTTP-заголовков в службах</i>	287
8.6.3. <i>RestTemplate и UserContextInterceptor: обеспечение передачи идентификатора корреляции нижестоящим службам</i>	289
8.7. Создание заключительного фильтра, добавляющего идентификатор корреляции	290
Итоги	293

9

Безопасность микросервисов	294
9.1. Что такое OAuth2?	295
9.2. Введение в Keycloak	297
9.3. Начнем с малого: использование Spring и Keycloak для защиты единственной конечной точки	299
9.3.1. <i>Добавление Keycloak в Docker</i>	299
9.3.2. <i>Настройка Keycloak</i>	300
9.3.3. <i>Регистрация клиентского приложения</i>	303
9.3.4. <i>Настройка пользователей O-stock</i>	308
9.3.5. <i>Аутентификация пользователей приложения O-stock</i>	310
9.4. Защита службы организаций с использованием Keycloak....	314
9.4.1. <i>Добавление в службы JAR-файлов Spring Security и Keycloak</i>	314
9.4.2. <i>Настройка связи службы с сервером Keycloak</i>	315
9.4.3. <i>Определение пользователей, кому разрешено обращаться к службе</i>	315
9.4.4. <i>Передача токена доступа</i>	320
9.4.5. <i>Анализ нестандартного поля в JWT</i>	326
9.5. Некоторые заключительные рассуждения о безопасности микросервисов	328
9.5.1. <i>Используйте HTTPS/Secure Sockets Layer (SSL) для взаимодействий между службами</i>	329
9.5.2. <i>Используйте шлюз для организации доступа к микросервисам</i>	329
9.5.3. <i>Разделите свои службы на общедоступные и закрытые</i>	330
9.5.4. <i>Ограничьте поверхность атаки на ваши микросервисы, заблокировав ненужные сетевые порты</i>	330
Итоги	331

10	Событийно-ориентированная архитектура и Spring Cloud Stream	332
10.1.	Обмен сообщениями, событийно-ориентированная архитектура и микросервисы	333
10.1.1.	Передача событий об изменении состояния с использованием синхронного подхода запрос/ответ	334
10.1.2.	Передача событий об изменении состояния с использованием сообщений	337
10.1.3.	Недостатки архитектуры на основе сообщений	339
10.2.	Введение в Spring Cloud Stream	340
10.3.	Простые издатель и получатель сообщений	342
10.3.1.	Настройка Apache Kafka и Redis в Docker	343
10.3.2.	Публикация сообщений в службе организаций	344
10.3.3.	Получение сообщений в службе лицензий	351
10.3.4.	Тестирование передачи сообщений между службами	355
10.4.	Пример использования Spring Cloud Stream: распределенное кеширование	356
10.4.1.	Использование Redis в роли кеша	357
10.4.2.	Определение собственных каналов	363
	Итоги	366
11	Распределенная трассировка с использованием Spring Cloud Sleuth и Zipkin	367
11.1.	Spring Cloud Sleuth и идентификатор корреляции	369
11.1.1.	Подключение Spring Cloud Sleuth к службам лицензий и организаций	370
11.1.2.	Особенности трассировки в Spring Cloud Sleuth	370
11.2.	Агрегирование журналов и Spring Cloud Sleuth	372
11.2.1.	Интеграция Spring Cloud Sleuth и стека ELK	374
11.2.2.	Настройка Logback в службах	376
11.2.3.	Определение и запуск приложений ELK в Docker	380
11.2.4.	Настройка Kibana	383
11.2.5.	Поиск идентификаторов трассировки Spring Cloud Sleuth в Kibana	386
11.2.6.	Добавление идентификатора корреляции в HTTP-ответ с помощью Spring Cloud Gateway	388
11.3.	Распределенная трассировка с использованием Zipkin	390
11.3.1.	Настройка зависимостей Spring Cloud Sleuth и Zipkin	391

11.3.2. Настройка в службах ссылки на сервер Zipkin	391
11.3.3. Настройка сервера Zipkin.....	392
11.3.4. Настройка уровней трассировки.....	393
11.3.5. Использование Zipkin для трассировки транзакций	394
11.3.6. Визуализация более сложных транзакций	397
11.3.7. Трассировка операций обмена сообщениями.....	398
11.3.8. Добавление дополнительных операций.....	400
Итоги	403
12 Развертывание микросервисов	404
12.1. Архитектура конвейера сборки/развертывания	406
12.2. Настройка базовой инфраструктуры для O-stock в облаке	410
12.2.1. Создание базы данных PostgreSQL с использованием Amazon RDS	413
12.2.2. Создание кластера Redis в Amazon	416
12.3. После подготовки инфраструктуры: развертывание O-stock и ELK	418
12.3.1. Создание экземпляра EC2 с помощью ELK	418
12.3.2. Развертывание стека ELK в экземпляре EC2.....	422
12.3.3. Создание кластера EKS.....	423
12.4. Конвейер сборки/развертывания в действии	430
12.5. Создание конвейера сборки/развертывания.....	432
12.5.1. Настройка GitHub.....	433
12.5.2. Сборка наших служб в Jenkins	434
12.5.3. Создание сценария конвейера.....	439
12.5.4. Создание сценариев для конвейера развертывания Kubernetes.....	441
12.6. Заключительные рассуждения о конвейере сборки/развертывания	442
Итоги	444
Приложение А.....	445
Модель зрелости Ричардсона	446
Spring HATEOAS.....	448
Внешняя конфигурация	448
Непрерывная интеграция и непрерывная доставка	449
Мониторинг	450
Журналирование.....	451
API-шлюзы	451

Приложение В	453
Тип разрешения: пароль	454
Тип разрешения: учетные данные клиента	455
Тип разрешения: код авторизации.....	456
Тип разрешения: неявный.....	459
Как обновляются токены.....	461
Приложение С	463
С.1. Введение в мониторинг с использованием Spring Boot Actuator.....	464
С.1.1. Добавление зависимостей Spring Boot Actuator.....	464
С.1.2. Включение конечных точек Spring Boot Actuator.....	464
С.2. Настройка Micrometer и Prometheus	465
С.2.1. Введение в Micrometer и Prometheus.....	466
С.2.2. Интеграция с Micrometer и Prometheus	467
С.3. Настройка Grafana	469
С.4. Итоги обсуждения	474
Предметный указатель	475