

Java微服务测试：基于Arquillian、Hoverfly、AssertJ、JUnit、Selenium与Mocki



[Java微服务测试：基于Arquillian、Hoverfly、AssertJ、JUnit、Selenium与Mocki_下载链接1](#)

著者:【美】亚历克斯·索托·布埃诺 (Alex Soto Bueno)

出版者:电子工业出版社

出版时间:2019-7

装帧:平装

isbn:9787121367106

《Java微服务测试：基于Arquillian、Hoverfly、AssertJ、JUnit、Selenium与Mockito》

从实战出发，介绍微服务架构所带来的测试方面的挑战，以及如何利用新的技术来应对这些挑战。通过本书，读者可以学会如何编写微服务架构下的单元测试、组件测试、集成测试以及契约测试。在此过程中还会用到 Arquillian、ShrinkWrap、Pact、Selenium、Docker、Hoverfly 等多个帮助测试的工具和框架。书中涵盖大量的代码和样例，可以帮助读者快速上手，并在自己的实际工作中应用这些技术。

《Java微服务测试：基于Arquillian、Hoverfly、AssertJ、JUnit、Selenium与Mockito》适合有一定 Java 基础的开发和测试人员，对使用其他编程语言的开发者也会有一定的帮助。

作者介绍:

Alex Soto Bueno 是一位 Java 大师，也是一位在Red Hat工作的软件工程师，主要研发提升测试体验的工具。他十分喜爱Java、软件自动化，并且深信开源软件。Alex 是NoSQLUnit 项目的创建者以及JSR374（JSON 处理的Java API）专家组的一员，并且是一位国际讲师。

Alex 从ZX Spectrum（使用POKE命令的遥远年代）就开启了编程之路，之后使用过多台不同的计算机，例如80286（感谢他的父母 Mili 和 Ramon 买下了它）。从La Salle大学计算机工程专业毕业后，他加入了Aventia，开始了自己的职业生涯，在此期间开发了一个生成并校验电子信号的平台。之后他加入了Grifols，并开发了一种医疗诊断设备，严格的测试成为其软件生涯中重要的一部分。之后他加入了Everis 的银行部门，其间开发了电子投票系统及 CloudBees。

工作之余，Alex 会陪伴他的妻子Jessica 以及两个女儿Ada和Alexandra，Ada和Alexandra 是他的掌上明珠。

Andy Gumbrecht 是 Tomitribe 的高级软件工程师。从12岁起他就对任何与计算机相关的事情感兴趣，并在那时幸运地拥有了一台高达 1KB 内存的计算机Sinclair ZX81。那时候许多早期的程序示例都是表单上的二进制码，需要手动输入到计算机中。这非常容易出错，因此Andy很快就找到他的兄弟John，让他作为自己的测试者来保证输入的正确性。这也是他第一次意识到测试代码的重要性。

接触机器码和BASIC 使Andy在代码优化方面的能力持续提升。在成为英国军队的皇家工程师之后不久，他回到德国的大学继续深造。之后他进入 PROVOX Sytemplanung GmbH 实习并在那里工作了很多年，主要开发政府软件。

Andy 从 2007 年开始进行开源软件的研发并在 2009 年加入了 Apache OpenEJB和Apache TomEE Application Server 项目，现在他已经成为项目管理委员会的一员。

Jason Porter 从12岁起就开始不断提升自己的软件开发技术。在那之前的几年间，他通过查看一个在古老的 80286 上用BASIC编写的游戏开启了自己的计算机和编程之旅。在编程方面的兴趣带领他之后又接触了Java 和 C/C++。他在 Netscape Navigator 和IE 浏览器时期成为一名早期的Web 开发者，和DHTML 以及层级元素的“斗争”占据了他的大量时间。15 岁时，Jason 在当地的一家Web 开发公司找到了一份工作，开始开发网站并利用Perl 来编写CGI

脚本（然而，他很想忘记那段岁月）。在那之后，他在多个领域工作过，使用过Java、PHP、Ruby、C# 以及JavaScript。

他认为自己主要是一名后端开发者，但是其实他拥有应对任何编程领域问题的能力。在Red Hat, Jason参与过多个不同的框架、网站和集成工作。他在美国和世界其他地方做过的演讲可以帮助人们更好地理解编程的概念或新的技术。Jason 现在和他的妻子以及5个孩子居住在犹他州，对于Jason 来说，带孩子可比从事计算机编程工作要困难多了。

目录: 第1章 微服务概述1

1.1 什么是微服务，为什么要使用微服务2

1.1.1 为什么要使用微服务 3

1.1.2 什么是微服务 5

1.1.3 持续集成、持续部署和Docker 6

1.2 微服务网络及其功能 7

1.2.1 微服务网络 7

1.2.2 微服务特性 8

1.3 微服务架构 9

1.3.1 资源组件 10

1.3.2 业务领域组件 10

1.3.3 远程资源组件 11

1.3.4 持久化组件 11

1.4 微服务单元测试 12

1.4.1 孤立型单元测试 12

1.4.2 联合型单元测试 12

总结 13

第2章 测试下的应用14

2.1 准备开始 15

2.2 准备工作 16

2.2.1 Java Development Kit 16

2.2.2 构建工具 17

2.2.3 环境变量 17

2.2.4 集成开发环境（IDE） 18

2.3 架构 18

2.3.1 游戏服务 19

2.3.2 评论服务 22

2.3.3 视频服务 24

2.3.4 聚合服务 26

2.3.5 整体架构 27

2.4 应用设计模式 28

2.4.1 解构 28

2.4.2 ECB 模式 38

2.4.3 其他模式 40

2.5 设计决策 42

总结 42

第3章 微服务单元测试44

3.1 单元测试技术 44

3.1.1 联合型单元测试 45

3.1.2 测试替身 46

3.1.3 孤立型单元测试 48

3.1.4 微服务的单元测试 49

3.2 工具 51

3.2.1 JUnit 51

- 3.2.2 AssertJ 52
- 3.2.3 Mockito 53
- 3.2.4 构建脚本的修改 54
- 3.3 编写Gamer应用的单元测试 55
 - 3.3.1 YouTubeVideoLinkCreator 测试 55
 - 3.3.2 YouTubeLink 测试 56
 - 3.3.3 Games 测试 58
 - 3.3.4 GamesService 测试 60
 - 3.3.5 GamesResource 测试 63
- 练习 68
- 总结 68
- 第4章 微服务组件测试69
 - 4.1 Arquillian测试框架 70
 - 4.2 @RunWith(Arquillian.class)注解72
 - 4.3 ShrinkWrap 工具类 75
 - 4.3.1 利用ShrinkWrap 构建归档文件 76
 - 4.3.2 给ShrinkWrap 归档文件增加内容 77
 - 4.3.3 添加资源 79
 - 4.3.4 增加库和依赖 80
 - 4.3.5 利用Maven 解析器增加复杂依赖 80
 - 4.3.6 增加服务实现 82
 - 4.4 写一次代码，然后复用 83
 - 4.5 构建脚本的修改 84
 - 4.5.1 定义Maven 依赖 85
 - 4.5.2 定义Gradle 依赖 86
 - 4.6 覆盖Arquillian的默认配置 87
 - 4.6.1 容器定义 89
 - 4.6.2 指定容器属性 89
 - 4.7 使用Arquillian REST扩展 90
 - 4.7.1 Arquillian REST 客户端扩展 90
 - 4.7.2 Warp REST 扩展91
 - 4.8 使用Arquillian测试Spring应用 93
 - 4.8.1 Arquillian Spring 框架扩展 94
 - 4.8.2 测试Spring Boot 应用 96
 - 4.9 更复杂的Arquillian测试 98
 - 4.9.1 测试远程组件 98
 - 4.9.2 测试资源组件 101
 - 4.9.3 测试领域组件 103
 - 4.9.4 测试持久化组件 103
 - 练习 106
 - 总结 106
- 第5章 微服务集成测试108
 - 5.1 微服务架构下的集成测试 109
 - 5.1.1 网关组件层 110
 - 5.1.2 数据映射器和仓库 111
 - 5.2 使用Arquillian持久化扩展进行持久化测试 111
 - 5.2.1 描述式方法 114
 - 5.2.2 过程式方法 116
 - 5.2.3 使用NoSQLUnit 进行持久化测试 121
 - 5.2.4 使用Arquillian 多部署的持久化测试 125
 - 5.2.5 使用Arquillian 序列的持久化测试 126
 - 5.2.6 构建脚本的修改 127
 - 5.3 Gamer应用的集成测试 128
 - 5.3.1 测试Comments 类 128

- 5.3.2 测试CommentsGateway 类 131
- 练习 137
- 总结 137
- 第6章 契约测试138
 - 6.1 理解契约 138
 - 6.1.1 契约和单体应用 139
 - 6.1.2 契约和微服务应用 140
 - 6.1.3 使用集成测试进行验证 145
 - 6.1.4 什么是契约测试 146
 - 6.1.5 谁负责契约 146
 - 6.2 工具 149
 - 6.2.1 Pact 150
 - 6.2.2 JVM 语言中的Pact 152
 - 6.2.3 利用Algeron 将Pact JVM 与Arquillian 生态系统集成 161
 - 6.3 构建脚本的修改 168
 - 6.3.1 使用Pact JVM 进行契约测试 168
 - 6.3.2 使用Arquillian Algeron 进行契约测试168
 - 6.4 为Gamer应用编写消费者驱动的契约测试 170
 - 6.4.1 评论服务的消费者端 170
 - 6.4.2 评论服务的生产者端 173
 - 6.5 契约类型总结 175
- 练习 176
- 总结 176
- 第7章 端到端测试177
 - 7.1 端到端测试在整个测试体系中的位置 178
 - 7.2 端到端测试技术 178
 - 7.2.1 垂直测试 179
 - 7.2.2 水平测试 179
 - 7.3 端到端测试工具介绍 180
 - 7.3.1 Arquillian Cube 180
 - 7.3.2 Arquillian Drone 180
 - 7.3.3 Arquillian Graphene 2 181
 - 7.3.4 JMeter 181
 - 7.3.5 Cukes in Space 181
 - 7.4 端到端测试样例 181
 - 7.4.1 构建微服务 182
 - 7.4.2 增加构建依赖和配置 182
 - 7.4.3 给测试增加@Deployment 注解和@TargetsContainer 注解 189
 - 7.4.4 跨域资源共享 191
 - 7.4.5 使用@ClassRule 处理混合环境 192
 - 7.4.6 使用@OperateOnDeployment 注解操作部署 197
 - 7.4.7 引入@Drone、页面对象、@Location 和WebDriver 198
 - 7.4.8 在测试中操作页面对象 200
 - 7.4.9 运行测试 202
- 练习 203
- 总结 203
- 第8章 Docker和测试 204
 - 8.1 Docker生态系统下的工具 205
 - 8.1.1 Docker 205
 - 8.1.2 Docker Machine 207
 - 8.1.3 Docker Compose 208
 - 8.2 Arquillian Cube 210
 - 8.2.1 设置Arquillian Cube 211
 - 8.2.2 编写容器测试 214

- 8.2.3 编写集成测试 217
- 8.2.4 编写端到端测试 221
- 8.3 REST API 224
- 8.4 Arquillian Drone和Graphene 225
 - 8.4.1 Arquillian Cube 和Arquillian Drone 的集成 225
 - 8.4.2 Arquillian Cube 和Arquillian Graphene 的集成 228
- 8.5 并行测试 230
- 8.6 Arquillian Cube和Algeron 231
- 8.7 使用容器对象模式 233
- 8.8 部署测试和Kubernetes 240
- 8.9 构建脚本的修改 245
 - 8.9.1 Arquillian Cube Docker 245
 - 8.9.2 Arquillian Cube Docker JUnit 规则246
 - 8.9.3 Arquillian Cube Kubernetes 246
 - 8.9.4 Arquillian Cube OpenShift 247
- 8.10 测试视频服务的Dockerfile 247
- 练习 249
- 总结 249
- 第9章 服务虚拟化 250
 - 9.1 什么是服务虚拟化 250
 - 9.1.1 为什么要使用服务虚拟化 251
 - 9.1.2 什么时候使用服务虚拟化 252
 - 9.2 使用Hoverfly模拟服务响应 253
 - 9.2.1 Hoverfly 模式 253
 - 9.2.2 JUnit Hoverfly 255
 - 9.2.3 配置Hoverfly .257
 - 9.3 构建脚本的修改 258
 - 9.4 对Gamer应用使用服务虚拟化 258
- 总结 260
- 第10章 微服务下的持续交付 262
 - 10.1 什么是持续交付 263
 - 10.2 微服务架构和持续交付 264
 - 10.3 持续交付的编排 266
 - 10.3.1 使用Jenkins 267
 - 10.3.2 Jenkins 流水线 267
 - 10.3.3 确定性部署 271
 - 10.4 Jenkins 272
 - 10.4.1 定义一个流水线 276
 - 10.4.2 Jenkins 流水线示例 278
- 总结 280
- 附录A 使用Arquillian Chameleon屏蔽不同类型的容器281
 - • • • • (收起)

[Java微服务测试：基于Arquillian、Hoverfly、AssertJ、JUnit、Selenium与Mocki 下载链接1](#)

标签

测试

Java

评论

标准实践类的框架，省略太多了，所以内容比较水，也就是提供个思路吧……

[Java微服务测试：基于Arquillian、Hoverfly、AssertJ、JUnit、Selenium与Mocki_下载链接1](#)

书评

[Java微服务测试：基于Arquillian、Hoverfly、AssertJ、JUnit、Selenium与Mocki_下载链接1](#)