首本讲解基于Java标准规范实现REST的专著，阿里巴巴Java技术专家12年开发经验结晶，3位业内著名技术专家联袂推荐！

深刻解读JAX-RS标准和API设计，Jersey的使用要点和实现原理，以及基于REST的Web服务的设计思想、原则和特点。

Java 核心技术系列

HZ BOOKS 华章科技

Java RESTful Web Service实战

Java RESTful Web Services in Action

韩陆 著

机械工业出版社
China Machine Press

Java 核心技术系列

Java RESTful Web Service实战

Java RESTful Web Services in Action

韩陆 著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Java RESTful Web Service 实战 / 韩陆著 . —北京：机械工业出版社，2014.10
（Java 核心技术系列）

ISBN 978-7-111-47888-1

I. J… II. 韩… III. JAVA 语言 – 程序设计 IV. TP312

中国版本图书馆 CIP 数据核字（2014）第 207228 号

Java RESTful Web Service 实战

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码：100037）
责任编辑：高婧雅　　责任校对：殷　虹
印　　刷：北京市荣盛彩色印刷有限公司　　版　　次：2014 年 10 月第 1 版第 1 次印刷
开　　本：186mm × 240mm　1/16　　印　　张：19.75
书　　号：ISBN 978-7-111-47888-1　　定　　价：69.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991　88361066　　投稿热线：（010）88379604
购书热线：（010）68326294　88379649　68995259　　读者信箱：hzjsj@hzbook.com

Foreword 推荐序一

——REST 开发的理想与现实

REST 是一种分布式应用的架构风格，也是一种大流量分布式应用的设计方法论。REST 是由（构成了 Web 基础架构的）HTTP、URI 等规范的主要设计者 Roy Fileding 博士在其 2000 年的博士论文（中文版名为《架构风格与基于网络应用软件的架构设计》）中提出的。到目前为止，关于 REST 最系统、最全面的论述，仍然是 Fielding 的博士论文。

REST 就是 Web（World Wide Web，简称 Web 或者 WWW）本身的架构风格，是设计、开发 Web 相关规范、Web 应用、Web 服务的指导原则。不符合 REST 风格要求的架构和技术，很难在 Web 这个生态系统中得到繁荣发展。在我看来，Roy Fielding 博士就是 15 年以来对于分布式应用架构设计理论贡献最大的人。Fielding 在 HTTP 规范的设计过程中，并没有采用当时大行其道的 DO（Distributed Object，分布式对象）风格，而是自出机杼、另辟蹊径，提出了一整套新的设计方法论。Fielding 的开创性工作，极大地推动了分布式应用设计理论的发展。

有趣的是，其实基于 SOAP/WSDL 的“大 Web Service”（以下简称 Web Service），几乎是与 REST 同时发展起来的。虽然在 Web Service 中也使用了对象，但是 Web Service 其实是 RPC 风格的，而不是 DO 风格的。Web Service 在最初几年发展很快，很大原因是它解决了 DO 风格难以解决的异构系统（不同的硬件系统、不同操作系统、不同的编程语言，等等）之间互操作性的问题。

然而遗憾的是，设计 Web Service 协议栈的核心人员，几乎都是来自于企业应用阵营的，尤其是来自于 IBM 和微软两家公司的人。这些企业应用的专家们没有充分认识到 Web 基础架构的巨大优点，甚至可以说并没有理解 HTTP 协议究竟是用来做什么的、为何要如此设计。在 Web Service 协议栈的设计之中，仍然有深深的企业应用痕迹。Web Service 虽然宣称能够很好地支持互操作，然而因为协议栈的复杂性很高，在实战中互操作性并不好（例如升

级过程困难而且复杂）。此外，Web Service 仅仅将 HTTP 协议当做一种传输协议来使用，还依赖 XML 这种冗余度很高的文本格式，这导致 Web Service 应用性能低下。很多开发团队宁可使用 Hessian 等轻量级的 RPC 协议，也不愿意使用 Web Service。在面向互联网的大流量 Web 应用（包括 Web 服务在内）这种运行环境中，Web Service 在复杂性、互操作性、性能、可伸缩性等方面的短板更加突出。因此，设计今日面向互联网的 API，已经很少有人会考虑 Web Service。这使得 Web Service 的使用被局限在企业应用运行环境之中，其名称中的“Web”更像是一个笑话（除了都使用 HTTP 协议，基本上与 Web 没什么关系）。假如在 2000 年，设计 Web Service 规范的专家们，能够认真读一下 Fielding 的博士论文，或者找 HTTP、URI 等 Web 基础架构规范的核心设计人员深入交流一下，Web Service 很可能就不是现在这个样子了。不过，历史是无法假设的。

在 Java 世界中，与大 Web Service 相对应的规范是 JAX-WS。在大 Web Service 已经成为明日黄花之后，Java 世界急需一套新的规范来取代 JAX-WS。这套新的规范就是 JAX-RS：Java 世界开发 RESTful Web Service（与 RESTful API 含义相同，可混用）的规范。虽然起步很晚，毕竟走上了正确的道路。

从 Java EE 6 开始，JAX-RS 在 Java EE 版图中，作为最重要的组成部分之一，逐步取代了 JAX-WS 的地位。在所有 Java EE 相关规范中，JAX-RS 是优点很突出的一个。例如，完全基于 POJO、很容易做单元测试、将 HTTP 作为一种应用协议而不是可替代的传输协议（因此提高了性能）、优秀的 IDE 集成，等等。可以说，在大多数场合，JAX-RS 完全可以取代 JAX-WS，作为 Java Web Service 开发的主要技术。JAX-RS 同样也可以完全取代 Hessian 等基于 HTTP 协议的 RPC 风格远程调用协议。毕竟 HTTP 本身就是一种 REST 风格的应用协议，以 REST 风格来使用 HTTP，才是最高效的使用方式。

Jersey、CXF 等支持 JAX-RS 规范的 REST 开发框架还支持输出 WADL。WADL 支持客户端代码自动生成，还可以将 WADL 导入到 SoapUI 等测试工具中，然后做自动化集成测试。从开发 Java 企业应用、取代 JAX-WS 的角度来看，JAX-RS 已经做得非常棒了。

尽管如此，不可不提的是，JAX-RS 这套规范，仍然存在着很多遗憾。需要特别指出的是，JAX-RS 规范并不等于 REST 架构风格本身，REST 的内涵要比 JAX-RS 广泛得多。学会了使用 JAX-RS 了，并不等于就完全理解了 REST，开发者仍然需要下工夫认真学习一下本源的 REST 究竟是什么。

例如，JAX-RS 规范对于应该如何定义一个资源，以及应该如何使用 HTTP 作为一个统一接口来操作资源，显然缺乏必要的指导。有很多开发者只是简单地将以前 JAX-WS 中的一个 endpoint 接口转换成一个资源接口。接口的方法太多，导致映射到的 HTTP 方法不够用，这也难不倒他们，在 URI 路径中加一些字符串就能够解决（例如，三个接口方法都映射到

POST，但是其 PATH 不同）。这样的开发方式非常常见，虽然开发者使用了 JAX-RS 规范，但是开发方式完全是 RPC 风格的，可以说与 REST 风格没有任何关系。

此外，JAX-RS 规范目前尚不支持 HATEOAS（将超文本作为应用状态的引擎，REST 风格的核心特征之一），从著名的 Richardson 成熟度模型（由《 RESTful Web APIs 》的作者 Richardson 提出）来衡量，基于 JAX-RS 规范实现的 RESTful API 仅仅能够达到成熟度模型的第二级，即支持资源抽象、统一接口的“CRUD 式 Web 服务”。

可以这样说，JAX-RS 规范与真正的 REST 风格，覆盖的范围其实是不同的。JAX-RS 覆盖的是简单基于 HTTP 协议（没有使用 SOAP/WSDL）的各种远程调用需求，很多需求对于可伸缩性、松耦合的要求并不高，仅仅是希望使用 HTTP 本身来取代大 Web Service 作为一种轻量级、容易测试的远程调用协议。REST 架构风格的严格要求，在这些场合并不是非常重要。慵懒是人类的天性，大多数开发者写代码只是为了解决手头的问题，JAX-WS 并不好用，JAX-RS 解救了他们。

如果按照 Roy Fielding 博士的严格要求（REST APIs must be hyper-text driven），那么包括 JAX-RS 规范在内都不能算是真正的 RESTful。然而，从实战角度，我认为革命不分先后，只要能够达到 Richardson 成熟度模型第一级，即有清晰的资源抽象，就可以认为是 RESTful API 了。如果连第一级都达不到，所设计的架构根本就不是面向资源的，那八成还是 RPC 风格的，就没有必要非要往 RESTful API 阵营里面挤了。从来没有人说过 RPC 就是万恶的，RPC 在企业应用的大多数场合其实都非常有效，只是不适合面向互联网的大流量 Web 应用而已。

因此，能够完美支持 HATEOAS，攀登到成熟度模型第三级，是一种理想情况（当然也是值得追求的）。而通过部分拥抱 REST 风格的要求，来更好地解决手头的问题，是更多开发者所面对的现实情况。JAX-RS 反映的正是这种现实情况，从实战的角度，它是一套非常有用也很好用的规范。

韩陆兄的新著《 Java RESTful Web Service 实战》是 JAX-RS 规范方面的专著，也是国内第一本 REST 开发的原创著作。这本书的实战性非常强，全面介绍了 JAX-RS 2.0 的方方面面，可以作为一线 Java 分布式应用开发者的案头必备书。如同我在前面所指出的，JAX-RS 规范并不等于 REST 架构风格本身，它们有着不同的覆盖范围。在本书中，作者也介绍了很多设计 RESTful API 的最佳实践，这些内容假如读者不理解 REST，甚至在亲自阅读了 JAX-RS 规范之后也未必能够总结出来。读者在阅读本书的过程中，不应该仅仅满足于掌握了 JAX-RS 开发的基本方法、解决了手头的问题、用其完全取代 JAX-WS，更重要的是，读者还应该就 REST 架构风格本身做更多的学习。幸运的是，除了本书之外，目前 REST 设计和开发方面的图书资料已经非常多了。

本书的内容非常严谨，有非常好的系统性，对于设计开发大流量 Web 服务会面临的各种问题都有涉及。特别是在自动化测试方面着墨颇多，在我看来是本书的一大亮点。RESTful API 的自动化测试非常重要，需要在设计之初就充分考虑到。本书是一本难得的原创佳作，值得所有 Java 分布式应用的开发者购买。

理想富丽丰满，现实贫瘠骨感，追求理想和注重解决现实问题其实并不矛盾。JAX-RS 规范的发展，反映出了 Java 社区在更好地开发 RESTful Web Service 方面的求索。尽管存在争议，在我看来，规范化是推动 RESTful Web Service 取得更大发展的必由之路。目前对于优秀的 RESTful API 有哪些判断标准，Web 开发者社区已经达成了高度共识，也积累了大量非常有价值的成果。JAX-RS 规范的发展，离不开 Web 开发者社区的这些成果。在未来的 JAX-RS 3.0 规范中，我们将会看到更多令人兴奋的成果被规范化。JAX-RS 2.0 已经做得不错了，但是在 RESTful Web Service 规范化的道路上，其实才刚刚起步，任重而道远。

李锟 于上海

Forword 推荐序二

半年前初识韩陆的时候，我们就聊到他正在写的这本书，当得知我从2006年就参与了Apache CXF开发，他立即邀请我为他的新书写序，我也就欣然答应了。

Apache CXF作为JAXWS以及JAX-RS规范的实现框架，已经成为很多Web服务开发者必选的开发框架。作为这一框架的开发维护者之一，我的日常工作经常需要熟悉这些JSR规范，并实现JSR所定义的API，解决最终用户的使用问题。

熟悉Java的人大多都听说过JSR（Java Specification Requests）、JCP（Java Community Process），通过JSR可以就Java某一方面的应用定义一组标准的API或者服务。对于最终用户来说，他们的代码只需要调用JSR定义的标准API，不做任何修改就可以调用不同的JSR实现。这里常见的例子就是Java Servlet应用，用户开发的Web应用可以不做任何修改就部署到Tomcat、JBoss等不同的Web容器中。

JAXRS是JCP为Java RESTful Web Service定义的一套API。由于Web服务的描述模型与Java类和接口有一定的差距，JAX-RS定义了很多annotation，通过这些annotation我们可以很方便地将Java类描述成为相关的REST服务。由于RESTful Web Service通常需要部署到Web容器中，JAX-RS也定义了相关服务来发现部署到容器中的JAX-RS应用。

读过JSR规范的朋友或多或少都会有这样的体会，JSR作为规范文档，其目标是将API定义以及实现功能描述清楚、完备，其目标读者是相关API的实现人员，或者是相关API的高级使用人员。如果读者对相关的背景知识还不熟悉的话，JSR文档读起来会比较晦涩而且难以理解。加之绝大部分JSR文档都没有相关的中文翻译，对于绝大多数初学者来说，通过阅读JSR文档来学习相关的API的知识是一个艰难的过程。

如果我们想要对JAX-RS规范有一个比较快速并且全面的了解应该怎么办呢？一般来我们可以通过JSR的相关参考实现入手，我们不但可以通过运行相关的参考实现的例子快速入门，还可以通过跟踪相关的代码对实现细节有一个全面的了解。韩陆的这本新作以JAX-RS

的参考实现 Jersey 为蓝本，由浅入深地向大家介绍了 JAX-RS 的由来，以及与 RESTful Web 服务开发的相关 API，并结合实例分享了作者的实战经验。

好了，现在打开你熟悉的 IDE 工具，加载 Jersey 代码库，沿着本书的指引去探索 Java RESTful Web Services 开发世界吧。

RedHat 姜宁

Preface 前　言

从我启动本书到完稿，历时一年有余。

此间，Java EE 7 得到了更多服务器软件的支持，Jersey 升级了 9 个小版本——我在动笔开始文字和示例代码编写的时候，Jersey 刚刚推出 2.0 版本，到本书完毕时，版本号是 2.9，这也是本书的最终版本。此后新版本带来的改变只有通过本书提供的源代码来同步更新。

此间，我积极参与和关注着 Jersey 项目的动向，通过关注 Jersey 官方文档、Jersey 在 GitHub 托管系统的源代码、Jersey 的 Jira 缺陷管理系统、Jersey 的 StackOverflow 问答系统，对其修复缺陷、引入新功能和如何使用 Jersey 等事宜不断跟进。

我之所以这么做，目的只有一个，即希望为读者呈现的是一本 Java 领域 REST 开发的好书。

为什么要写这本书

REST 式的 Web 服务有多流行，相信每一位翻阅本书的读者都很清楚，冒昧地猜测，你可能想要看到的是一本讲述如何使用 Java 语言和 Java EE 平台，来实现这一风格的服务或者应用的书。这也正是我这两年来努力写作的初心和原动力。我相信，读者希望看到的内容不单单是追逐流行、风靡一时的“快餐”。作为开发者，我知道拥有对新技术、新标准的敏锐嗅觉非常重要，但我认为更难能可贵的是把一个业内认可的标准学好和用好。Java EE 7 中包含了 JAX-RS 2.0 标准，是 Java 领域 REST 式的 Web 服务的规范；GlassFish 是 Java EE 7 的参考实现项目集，Jersey 是其子项目，是 JAX-RS 2.0 的参考实现。本书的目的就是要把 JAX-RS 2.0 说清楚，把如何用 Jersey 写好 REST 式的服务讲明白。

本书特色

- ❑ 第一本完整讲述使用 Java 标准规范实现 REST 的书籍。
- ❑ 第一本完整讲述以 JAX-RS 2.0 参考实现实践 Jersey 的书籍。

❑ 给出深度学习和实践 JAX-RS 的线路图和解决方案。

读者对象

本书从实践角度，完整地诠释了 JAX-RS 2.0（JSR 339），即 Jersey 2.0 的核心元素和 REST 开发过程。面向所有在 Java 领域学习和使用 REST 的读者。同样欢迎 REST 领域的其他语言的使用者通过本书了解 REST 的实现。

❑ 技术路线：架构师、技术主管、研发工程师、REST 小白（网络用语，本书指新手）；
❑ 管理路线：部门经理、项目经理、产品经理；
❑ 敏捷实践者。

如何阅读本书

本书收纳了笔者近三年的 RESTful 实战经验，将 REST 理论与 Java 实现相结合，循序渐进地将使用 Java 开发 REST 式的 Web 服务中遇到的知识点和经验呈现给读者。每个章节中的知识点都精心设计了相应的示例代码，便于读者更好地理解和更及时地进行实践。从第 11 章开始，笔者从敏捷角度为读者呈现了一个完整和相对复杂的 REST 式的 Web 服务实例，相信这个实例能让读者更好地理解相关内容，同时，可以对敏捷开发和自动化测试有新的认识。

全书分为 3 篇，共 11 章。

第一篇共 5 章（第 1 ~ 5 章），讲述 REST 的基本理论和 Jersey 的基本实践。完成第一篇的阅读和示例代码的实践，读者可以学会使用 Java 开发 REST 式的 Web 服务的基本能力。

第 1 章 分别阐述了 REST、REST 式的 Web 服务、JAX-RS 2.0 和 Jersey 2.x 的基本情况。

第 2 章 讲述了使用 Jersey 2.x 开发一个基于 JAX-RS 2.0 标准的应用的基本知识以及如何使用 Jersey 来集成 Spring 和 JPA 以快速开发一个 REST 式的 Web 服务。本章包含 10 个示例项目。

第 3 章 深入阐述了如何使用 Jersey 设计和实现 REST 式的 Web 服务的 API。本章包含 5 个示例项目。

第 4 章 深入阐述了 Jersey 的 Providers 对 REST 请求的处理。

第 5 章 讲述了 Jersey 的客户端开发的基本实践和常用配置。

第二篇共 5 章（第 6 ~ 10 章），讲述写好 REST 程序的必要知识点。完成第二篇的阅读和实践，读者可以全面了解如何写好一个完整的 REST 式的 Web 服务。

第 6 章 全面讲述了如何实现一个安全的 REST 式的 Web 服务。

第 7 章 讲述了 Jersey 的测试框架及其使用。

第 8 章 讲述了 Jersey 对 HTML5 的 SSE 的支持和异步请求处理。

第 9 章 讲述了 Jersey 1.x 迁移到 Jersey 2.x 的要素和经验分享。

第 10 章 分享了 REST 式的 Web 服务的性能调优的经验。

第三篇包含 1 章（第 11 章），分享了 5 年外企工作中，我对自动化测试和敏捷的体会。完成本部分内容的阅读和实践，读者可以更宏观地审视 REST 的应用场景，起到抛砖引玉的作用。

第 11 章 讲述一个完整的 REST 项目的全过程。

全文由三个小版组成："阅读指南"、"小白讲堂"（为某些在知识点上比较"小白"的同学介绍概念性的知识）和"宅人坑事"（技术宅最自豪和最担惊受怕的就是"踩坑"），旨在和读者分享基础知识和心得体会，只为交流，切勿对号入座。

第一篇推荐研发工程师和 REST 小白完整阅读，这部分包含了了解和使用 JAX-RS 2.0 完成学习和工作的必要章节。对于有基础的技术人员，可以作为实践的参考有选择地阅读。

第二篇推荐致力于提高自己的技术人员完整阅读，这部分包含了 JAX-RS 2.0 的高级功能。永不满足和持续学习的精神，会让你在关键时刻成为"关键先生"。架构师和项目经理在考虑安全、性能等问题时，可以参考相关章节。

第三篇推荐渴望实战指导的技术人员阅读和跟随实践。同时，该篇结合了笔者参与敏捷实践的体会，以 scrum 的方式进行开发管理。因此，敏捷实践者和相关的部门经理可以参考。

产品经理可以阅读与 REST 特性相关的章节，这样可帮你在设计应用方面有所提高。

源代码

章节	源代码地址
第 1 章 ~ 第 10 章	https://github.com/feuyeux/jax-rs2-guide
第 11 章	https://github.com/feuyeux/jax-rs2-atup

勘误和交流

作为开发者，非常欢迎读者能与我一起交流 JAX-RS 2.0 相关的技术。我的邮箱是：feuyeux@gmail.com，新浪微博是：六爷 1_1。

本书的勘误统计在 https://github.com/feuyeux/jax-rs2-guide/wiki 中，欢迎读者批评指正。

致谢

感谢我的夫人 Caroline。写作期间，没有时间陪你共度周末，甚至没有精力陪你共度以 2 开头的最后一个生日，也没有营造任何浪漫的气氛。我们的女儿从蹒跚学步到会唱儿歌，我没有更多的时间陪伴你们，而你一如既往地支持我，在此我对你表示深深的感谢。

感谢华章公司的杨福川。我最难忘的是最初谈到要出本书时与君一拍即合的快意，这为我此后动笔增添了无穷的动力和信心。同时，阁下给予我的信任和鼓励也极大地推动了本书的顺利完成。在此后一年多的交往中，我深深感受到了业内对杨福川专业水准的评价和他敬业的态度。

感谢华章公司编辑高婧雅专业和耐心的审阅和指正。我曾一度被这位可敬的东北妹子的严谨和勤奋打击，心生愧对华章精品精神的挫败感。但随着婧雅的不断鼓励和支持，我一遍遍改进着稿件，自信心得到了恢复，同时，内心得到了安慰，我没有偏离初心和那份原动力。非常感谢华章公司和婧雅。

感谢 RedHat 公司姜宁师兄的技术校对，感谢阿里巴巴公司许晓斌的敏捷校对。

最后，我要感谢 Technicolor 敏捷团队中的每一位成员，正是与各位度过的每一天，才让我有资本和信心与广大读者分享在 REST 式的开发中的经验和教训，并在本书中介绍自动化测试和敏捷实践的相关知识。

Contents 目　录

第二篇　全面掌握——JAX-RS 2.0 进阶