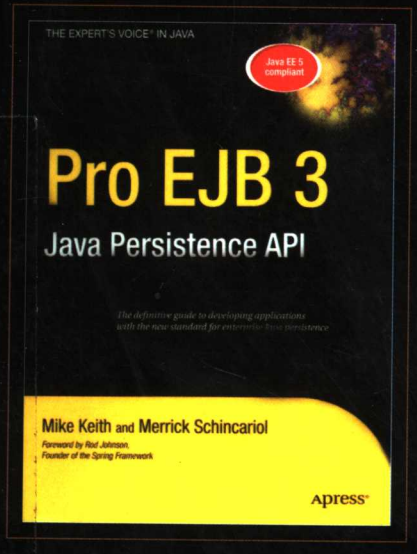
HZ BOOKS

Apress®

华章程序员书库

EJB 3.0 专家编程

Pro EJB 3 Java Persistence API



Mike Keith　Merrick Schincariol　著
赵 睿　赵俊明　等译

"Java Persistence API是Java EE 5平台修订版中最重要的提升特性，而本书让这个特性易于使用！"

Rod Johnson，Spring框架之父

机械工业出版社
China Machine Press

华章程序员书库

EJB 3.0 专家编程

Pro EJB 3 Java Persistence API

Mike Keith Merrick Schincariol 著
赵 睿 赵俊明 等译

机械工业出版社
China Machine Press

本书全面讲解如何在企业和桌面应用程序中使用持久化，并介绍Java Persistence API规范中的技术要点和实际应用。内容包括：应用组件模型、实体管理器、（高级）对象关系映射、查询和查询语言、XML映射文件、以及打包、部署、测试和移植。

本书并不是简单地对应JPA规范，罗列各种策略和术语，而是深入浅出地介绍规范中的技术要点，阐述其应用环境和最佳实践，并提供精选的示例和图解，对那些最常用或最经典的场景进行示范。

本书适合Java软件开发人员阅读。

Mike Keith, Merrick Schincariol: Pro EJB 3.0 Java Persistence API（ISBN：ISBN 1-59059-645-5）.

本书版权登记号：图字：01-2007-0692

图书在版编目（CIP）数据

EJB 3.0 专家编程 / 凯特（Keith, M.），斯琴塔瑞尔（Schincariol, M.）著；赵睿译. －北京：机械工业出版社，2008.1

书名原文：Pro EJB 3 Java Persistence API

ISBN 978-7-111-22489-1

Ⅰ.E… Ⅱ.① 凯… ② 斯… ③ 赵… Ⅲ.JAVA语言－程序设计 Ⅳ.TP312

中国版本图书馆CIP数据核字（2007）第154314号

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：周茂辉

北京京北制版厂印刷 · 新华书店北京发行所发行

2008年1月第1版第1次印刷

186mm × 240mm · 24.25印张

定价：49.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

译者序

Java Persistence API的发布，可能是最近几年来能让我为技术的进步感到兴奋的少有的几件事之一。作为一名体验过EJB之前多个版本、也在多种对象关系映射技术中挣扎过很久的开发人员，JPA的发布确实使我感到兴奋，同时心里产生跃跃欲试的感觉。我很荣幸能够成为这本书的译者，因为之前已经久仰过Mike Keith的大名，也拜读过他的文章聆听过他的演讲，所以知道他是一名注重实干的人，他的书质量肯定不低。

本书作为一本针对规范的书，并不是简单地与规范对应，罗列各种策略和术语，而是深入浅出地介绍规范中的技术要点，阐述其应用的上下文和已知的最佳实践，并提供精选的示例和图解，对那些最常用或最经典的场景进行示范，并比较各种手段、方式的优劣。

保持作者的原有风格、最大限度将作者的语气和情感体现在译作中，是我们的宗旨，但是为了符合国内读者的习惯，帮助读者能够更好地理解文中的内容，我们将一些隐喻或用语做了适当的调整（或者添加少量的译注）。在用词上我们尽量以简单、通用的词句为主，但是限于这是一本针对规范的著作，免不了要包含很多的术语或者业内用语。因此，希望读者能够在阅读时与规范相比较，并行阅读，以增强对其中内容的了解。另外，JPA作为一个规范，始终不能脱离其空中阁楼的地位，希望大家在阅读的时候，能够针对所熟悉的其中一种（或多种）持久化容器，将书中的实例在真实环境中运行一下，这样不仅能增强对规范的理解，同时也能对书中提到的不同持久化提供商的差异有所了解，在今后的工作中能够很好地运用规范。

尽管译者对持久化相关的技术和背景知识有较深了解，并具备多年的实践经验，但是鉴于个人水平，以及有限的时间，翻译过程中难免有所疏漏和错误，请广大读者予以指正，在此深表谢意。

参与本书翻译工作的有赵睿、赵俊明、窦巍、刘春霞、黄庆昆、马丽霞、杨斯雯、赵北扬、刘卓、杨磊、张楠、张敏、陈红、赵斌，并由赵睿统一全稿。机械工业出版社华章分社的编辑对全书进行了细致的审校，提供了大量宝贵的修改意见和指导，在此一并致谢。

序　言

我相信JPA（Java Persistence API，Java持久化API）是Java EE 5平台修订版中最重要的提升特性。它为ORM（Object-Relational Mapping，对象关系映射）提供了简单而强有力的标准。它领导持久化供应商联合起来，一起开发它，同时也使开发人员可以联合起来一起采用它。

直到EJB2.1，也包括EJB 2.1，持久化技术（实体Bean）是整个EJB规范里的最薄弱环节——既复杂又缺少一些重要的能力，例如继承建模，并且独立时不可使用。JPA现在是一个很好的模型，可以独立并应用于一定范围的环境中。在未来的更新版本中，JPA独立于它的运行时环境的能力可能是确定的，JPA会从EJB中分离出来，成为独立的规范。因此JPA与操作关系型数据库的所有Java开发人员息息相关，而不仅是那些完全使用EJB组件模型的人。

从2001年EJB 2.0的发布到现在，企业型Java向一个更为简单化的方向变革——基于POJO的编程模型。这一转换相当于在简单Web应用开发人员的财富500强中奠定了它的地位。持续关注的焦点是在Java语言上，而不是复杂的容器合同上。现代基础结构可以将企业服务应用到POJO上，而不会使它们的需求沾染上基础结构。

不像以前的实体Bean模型，JPA提供了持久化POJO的能力。JPA规范从跨越诸如TopLink、Kodo和Hibernate这类产品超过十年的集体经验中抽取而成。基于POJO的持久化提供了许多重要的优点，这些优点是之前的企业Bean模型所缺少的：简单、富有成效的开发体验，易于测试，以及构建和持久化捕获业务概念的真实域模型的能力。POJO持久化的崛起，将会逐渐地帮助我们更好地设计应用程序，在采用J2EE开发的传统过程风格的地方，更大地发挥面向对象的作用。

当然，ORM并不是一个新概念，很多开发人员已经成功地使用一些ORM产品工作了好多年。JPA的重要性在于，它将各个实现提供给用户选择，而不需要使他们屈从于一个最小公分母（译者注：意指所有产品的一个大杂烩）。所有处于领导地位的ORM产品都将会支持JPA，并附带有它们各自的专有API（或者其他标准，例如JDO）。在将来，选择转向JPA API的用户将不会被强迫更换持久化产品，但是却能够获取更大的灵活性。

作为开源项目Spring Framework的领导者，我尤其高兴，因为处于领导地位的JPA实现也完全是开源的。Spring Framework 2.0发布版本与JPA集成在一起，并且我们可以提供帮助，使Spring的用户更容易在任何环境中使用它。我们很激动，因为我们工作在开源JPA实现的社区中，这样有助于让Spring/JPA的体验越来越好。

因此这是一个重要的主题。希望你觉得应该读一本关于JPA的书，但是为什么应该读这一本呢？

Mike Keith绝对够资格来写这样一本书。他同时也是EJB 3.0规范的共同领导者，不仅因为他对规范的相关知识十分熟悉，还因为他在完成规范的过程中扮演了至关重要的角色。他在持久化引擎方面的经验要追溯到15年前。他还曾经在TopLink团队中作为一个关键成员工作了5年，在J2EE 1.3和1.4持久化的黑暗年代里，在被称为POJO的持久化技术变得流行起来之前，

TopLink一直是业界的冠军产品。

最重要的是，Mike和Merrick Schincariol已经将他们的经验转换成一部清晰的、极具可读性的书，提供了对JPA的完整介绍。而且你一定不会失望，因为你增加了在JPA方面的经验，并且在你寻找关于高级问题和最佳实践的有价值内容时，这本书绝不会让你觉得陷入到太过深入的困境中。作者将你如何能够有效地使用JPA这个大的场景清晰地传达给你。许多代码示例确保讨论能够切合实际，而且采用的是直接与开发人员相关的示例。

我建议读这本书，并且我相信你将会与我一样，对这本书有很高的评价。

Rod Johnson
Spring框架之父
Interface21公司CEO

前　言

JPA（Java Persistence API，Java持久化应用程序接口）宣称即将推出已经有一段时间了，甚至有些人会说它已经过时了。基于“POJO”开发模型的企业Java持久化标准的到来，填补了平台中早就应该被填补的一个空白。以前的尝试都忘了这一标记，反而鼓吹对于开发来说很笨拙的、且对于很多应用程序来说都太沉重的EJB实体Bean。它从来没有达到被业内许多部门广泛采用或普遍认可的程度。但是在缺乏标准的情况下，像JBoss Hibernate和Oracle TopLink这样的私有持久化产品在业内得以普及，而且已经取得成功。随着JPA的出现，现在，开发人员能够创建可移植的持久化代码，这些代码将能够运行在任何适应Java EE 5的服务器上，以及服务器以外独立的JVM中。

已经可以证实，等到持久化市场成熟起来以后，出现的高级标准是以产品和用户经验为基础的，而不是以理论和设计为基础。JPA中所包含的是所有持久化行家们根据对现有产品的经验而公认的基本概念和接口。这更容易使已经在使用这些产品的人们选择采用JPA，使初学持久化的开发人员选择学习这个API，并快速地学会它。

已经有为架构师和开发人员撰写的规范，但是它终究是个规范。很少有人愿意坐下来一字一句地去读规范来学会如何使用API。它并不探究应用这个API的复杂性，也不解释在开发过程中可能会碰到的任何外围问题。在这本书中，我们希望为这个主题提供更加实用的方法，并突出使用一些我们认为有价值的模式。

我们的本来想法是围绕整个EJB 3.0规范来写。同时我们也打算让这本书完成后能够适合放在笔记本电脑包中而不要过重。很快我们就意识到，为了能够提供足够的主题范围，编写一本对于一般的开发人员有意义的书，只需要把注意力集中在半个规范上即可。假设我们缺乏JPA以外的现有知识，那么就很容易做出选择了。

这本书从头到尾将会详细阐述这个API的所有要素。我们会解释概念，并举例说明如何在应用程序中应用它们。我们首先通过在Java SE环境中创建一个非常简单的应用程序，来快速浏览这个API。然后，提供EJB 3.0和Java EE 5标准中适用于企业应用的持久化特性的概述。

对象关系映射处于关系数据库中存储对象状态的中心，我们详细阐述了这个API支持的ORM技术，以及映射和特征。EntityManager是用来与实体交互的主要接口。我们探索了使用实体管理器的不同方面，揭示了它的内部实现，以帮助你理解一些重要的细微差别。我们还探索了从实体管理器中可以获得的查询，在各种可以获得的动态、静态或命名查询之间进行区分。评价了查询能力，它们是可以访问的，并介绍了关于在什么时候应该使用哪种查询。我们全面讨论了Java持久化查询语言（Java Persistence Query Language，JPQL），并举例说明它的所有特征。

接下来，我们讨论了一些关于ORM的中级和高级主题，例如继承，并且演示了如何将各种类层级结构与各种数据模式进行映射。我们还深入研究了锁定这个重要的主题，并解释了如何

在应用程序中最好地使用锁定策略。对于那些因为元数据而喜欢用XML的人，我们描述了XML映射是如何指定的，并解释了它们如何能够用来覆盖注解。然后，通过讨论如何在不同的企业应用组件中配置和打包持久化单元，结束开发的生命周期。

关于测试已经出了很多书，而且仍然继续在出这方面的书。这个API的另一个优势是，它支持单元测试和一些当前正在使用的其他现行测试方法和模式的能力。我们花一些时间讨论了可以用来测试实体和调用它们的应用逻辑的一些方法，包括应用服务器内部调用逻辑和外部调用逻辑。

最后，对于那些打算从现有的持久化系统改用这个API的人，我们用了一些时间来研究移植问题。关于为了使用JPA而移植不同架构应用程序的方法，我们通过使用一些通用的设计模式，提供了一些建议。

我们希望你能喜欢这本书，并学会怎样使用JPA。我们不可能在这本书中涵盖所有内容，但是希望能像Indiana Jones所说，我们“明智地选择了”内容。欢迎读者对本书提出任何建议或意见。

本书为谁而写

我们为想要在企业和桌面应用程序中使用持久化的任何开发人员编写本书。我们假设你没有任何持久化产品的经验，但是你要有一些Java编程经验，并了解J2EE平台。新的Java EE 5标准的经验可能会有所帮助，但当然不是必需的。也不需要EJB以前版本的相关知识。

要学习在关系数据库中映射对象并存储数据的持久化API，需要对数据库和SQL有一些基本了解。另外，因为这个API是在访问数据的 JDBC（Java Database Connectivity，Java数据库连接）API基础上实现的，所以JDBC API的任何相关知识也都是很有用的，但绝对不是必需的。

关于代码示例

有时，一段代码比大段文字更有价值。我们尝试使用更实用或更能够满足目的的内嵌代码作为示例。虽然我们更愿意从代码中学习，而不是去读大段的文字，但是我们发现，当一页一页读示例代码的时候，直到代码的结尾，可能已经忘了想从这段代码中学些什么。我们通过使用省略号省略例子中没有必要的部分，努力减少无关代码对阅读的影响。我们希望你能理解这样做使例子更有意义，而不会认为它们不完整。

关于持久状态的访问调节器，在一定程度上这个API是灵活的，因为它可以是包范围的、保护的或是私有的。我们定义了实体状态在示例中一贯是私有的，来强调状态应该怎样封装在实体中。对于记录，我们不会武断地认为其状态就是私有的。我们只是碰巧用这种方法开始，并且始终没有要修改它的烦恼。

为了保证集中精力理解技术，而不必苦苦思考样例域，我们采用尽可能最简单的、最流行的域模型——真实可靠的Employee模型。然而我们不得不承认，它无所不在，以至于我们可以打赌实际上这个星球上的每个开发人员都理解它并且都涉及到它。它包含所有模型化可变性（无可否认，尽管我们的确必须在某些情况下扩展它），而这正是举例说明这个API的概念和实践所必需的。

本书中的例子已经用Java EE 5应用服务器和JPA的官方参考实现（Reference Implementation，RI）实现了。Java EE 5 RI叫做“Glassfish”，它是在共同开发和分发许可（Common Development and Distribute License，CDDL）下能够获得并使用的完全公开的开源应用服务器。用于JPA的RI叫做“TopLink Essentials”，它是从商业的分布式Oracle TopLink企业数据集成栈中衍生出来的产品，它是开源的，并且可以免费获得。Glassfish和TopLink Essentials可以从java.net的Glassfish产品下载链接上获得，但是我们推荐你去持久化页面下载，其网址为http://glassfish.dev.java.net/javaee5/persistence。TopLink Essentials还可以从Oracle Technology Network上的TopLink页面获得，其网址为http://www. oracle.com/ technology/ products/ias/toplink。

可以从Apress网站上下载得到这些例子，其网址为http://www.apress.com。我们建议下载并在那里浏览。学习这个API的最好方法就是你自己亲自试用。采用现有的模型，并将它与例子结合是开始学习的非常好的方法。这个API本身就是最好的卖点。一旦用它进行开发，你将会发现持久化开发比以前的开发要容易得多！

联系方式

可以通过michael.keith@oracle.com和merrick.schincariol@oracle.com联系我们。

致谢

我想要感谢为EJB 3.0规范做出贡献的所有专家组成员。大量的电话会议，无数小时在电话中与Linda、Gavin、Patrick和其他人，以及数以千记的email得出的结果，我们希望它对于这两年来我们为之付出的时间来说是值得的。

我想要感谢4位D（Dennis Leung、Dan Lesage、Doug Clarke和Donald Smith），他们在这本书的各个阶段提供了支持和友谊。感谢Shahid对早期各章节草稿的审阅，以及许多其他临时审阅人员对不固定章节的审阅。我特别应该感谢3位伟大的朋友：Jason，因为他孜孜不倦地将所有示例变成代码（同时修改我的bug！），审阅，甚至还编写早期章节的部分草稿；Huyen，因为他远超过职责的、对稿件夜以继日地进行审阅，以满足这样紧的进度要求；当然还有Merrick，在我无法完成任务或者不能继续的时候，他能够与我合作并使我再次振奋。Apress的Tony、Julie、Hastings和Laura一直提供帮助，并且奇迹般地使本书得以出版。最后，也是最重要的，感谢我的妻子Darleen和我的孩子们Cierra、Ariana、Jeremy和Emma。我对他们的爱永无止境。是他们为这本书做出了最大的牺牲，他们耐心等待了好几个月，期间他们只看到一个疏远的丈夫和父亲，整天懒散地坐着并不断在笔记本电脑上敲着。

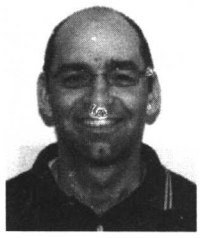
Mike Keith

写一本书所涉及的人远比列在这里的人名要多得多。在去年整整一年中，我一直被保佑着，所以能够得到如此多的人支持，他们提供建议、审查我的工作并且一直不断地鼓励我。最首要的，我要感谢Mike，给我提供了与他合作编写这本书的机会，从始至终一直都是一种真正的伙伴关系。但是，如果没有我的妻子Natalie爱的支持，以及我年幼的儿子Anthony不平凡的耐心——

他不得不容忍他的父亲常常躲在办公室里不停地写，这是不可能完成的。在Oracle，我要特别感谢Jason Haley，他在我工作于这个项目的时候，担负了远多于他自己的资深工程方面的责任，也感谢Dennis Leung、Rob Campbell和全部EJB容器团队的支持。在Apress方面，Julie Smith、Hastings Hart和Laura Esterman帮助度过了让事情按期完成的所有难关。最后，感谢很多审阅过这本书草稿的审阅者们。特别是Huyen Nguyen和Jason Haley帮助我们提高质量，使这本书更为精确和更具可读性。

Merrick Schincariol

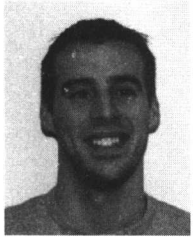
关于作者



Mike Keith是EJB 3.0的联合规范领导者，并且是Java EE 5专家组的一名成员。他持有加拿大卡尔顿大学（Carleton University）的计算机科学硕士学位，并在对象持久化方面有15年的执教、研究和实践经验。他曾为财富100强企业在很多的技术中实现过持久化系统，包括关系型和对象型数据库、XML、路径服务以及自定义数据格式。从EJB初出茅庐的时候，他就从事EJB实现和与多个应用服务器集成的工作。他写过各种各样的论文和文章，并在很多会议中演讲过关于EJB 3.0的内容。目前他受雇于Oracle公司，为一名持久化架构师。

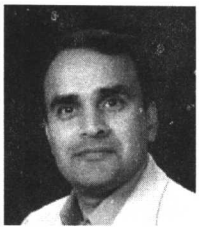
Merrick Schincariol 是Oracle资深工程师，并且是EJB 3.0规范的审核人员。他持有加拿大湖首大学（Lakehead University）的计算机科学学士学位，并且在业界有超过七年的经验。在开始转向编写Java和J2EE应用之前，他在预备采用Java 的企业和商务智能领域工作过。他具有大规模系统和数据仓库设计经验，使他在企业软件方面具有成熟和实践性的观点，这些后来促使他进入EJB容器的实现工作。他现在是Oracle公司EJB 3.0产品的领队工程师之一。

关于技术审阅者



Jason Haley 是一名Oracle资深工程师，也是EJB 3.0规范的审核人员之一。他拥有卡尔顿大学的计算机科学学士学位，并且在EJB内部，以及Java和J2EE应用方面有不止7年的经验。他也做过一些咨询和培训工作，但是他把大部分的时间花在设计和实现EJB容器基础结构的会话和持久化方面。他在BEA WebLogic Server和Oracle Application Server的内部运行方面有广泛的经验，并且设计了多会话和持久化管理器接口。他是一名Oracle公司的EJB容器的领队工程师。

Huyen Nguyen是Oracle服务器技术组的一名质量保证经理。他拥有滑铁卢大学系统设计工程专业的应用科学硕士学位。在对象持久化领域，他作为一名讲师和咨询师工作了11年，在过去4年里他是一名质量保证经理。



Shahid N. Shah 是Netspective Communications的创始人和CEO。Netspective是一个软件开发公司，为使用Java和.NET技术的面向服务系统提供必要的工具和技能。他最近构建了一个采用了EJB 3.0和Java 5.0的大型保健信息框架。Shahid曾经在大型企业担任过15年的技术VP、CTO、首席软件架构师以及领导工程师。Shahid的关键技术专业领域是面向服务架构、分布式对象服务、Java、J2EE、.NET、XML、UML以及面向对象和面向方面的软件开发等方面。Shahid有三个博客。在http://shahid.shah.org，他写的是关于架构问题的内容；在http://www.healthcareguy.com，他为在保健方面如何应用各种技术提供了非常有价值的观点；而在http://hitsphere.com，他作为汇集者给我们提供了一个对保健IT博客群体的快速预览。他的电子邮箱是：shahid@shah.org。

目　录

第1章 引 言

企业这个词在当今的软件开发中可以说是一个用得最过量的词。然而，当有些人声明他们正在开发企业应用时，在他们脑海里始终萦绕的是这样一个词：信息。企业应用根据他们需要搜集、转换和报告的大量信息来定义。当然，信息并不是简单地凭空存在的。数据存储和检索是一个数十亿美元的生意，近年来涌现的急速增长的企业集成系统（Enterprise Integration Systems，EIS）和企业应用集成（Enterprise Application Integration，EAI）公司已经证实了这一点。

许多年来，有很多持久化数据的方法来来去去，但是没有哪一个概念比关系型数据库具有更强的持久力。已经证实，目前世界上大量的企业数据都存储在关系数据库中。它们是所有企业应用的出发点，因为应用都带有预期生命期限，在应用慢慢退化后还能够持续很长时间。

了解关系数据是企业成功发展的关键。开发与数据库系统一起运行良好的应用程序已经成为软件开发的主要业务。尤其是对于Java来说，它的成功部分可以归功于它是建立企业数据库系统时被广泛采用的语言。从消费者网站到自动化网关，Java应用程序都是企业数据开发的中心。

尽管Java平台已经能够与数据库系统很好地配合了，但是它仍然存在一个问题：在数据库系统和Java应用的对象模型之间来回移动数据，比它需要的难得多。Java开发人员似乎要花费很多时间将行和列数据转换成对象，或者发现自己还是受约束于那些试图对开发者隐藏数据库的专有框架。

幸运的是，终于有一个解决办法即将到来。最近，由于Java Persistence API被来自跨领域的商业和开源势力标准化和支持，以至于它对我们处理Java内的持久化方式产生了重大的影响。开发人员第一次有了一个能拉近面向对象域模型和关系数据库系统之间距离的标准方法。

在这本书中，我们将介绍Java Persistence API，并探索它为开发人员提供的一切。不论你是在构建收集Swing应用中的表单数据的客户端/服务器（Client-Server，C/S）应用，还是在构建使用最新的应用框架的网站，Java Persistence API都是你能够使用的、实现持久化更加有效的框架。它的主要优势之一是，它能够被用在应用程序需要它处在的任何一层、级或框架中。

为了给Java Persistence API定级，本章首先回顾一下我们处于什么位置，以及我们正在试图解决什么问题。我们将看看规范的历史，并会提供它为开发人员带来价值的抽象视图。

1.1 Java对持久化的支持

Java平台为管理关系数据库的持久化提供了很好的支持。在最早期的平台上，编程界面已经存在，用来将网关提供给数据库，甚至抽象了许多供应商特有的业务应用持久化需求。在接下来的几个部分中，我们将看看目前关于持久化的Java标准集合，以及它们在企业应用中所扮演的角色。