开发专家之Sun ONE

精通Hibernate：Java对象持久化技术详解

Let Java objects
hibernate in the relational database.

孙卫琴 编著
飞思科技产品研发中心 监制

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

光盘内容为本书所有范例源程序，以及本书涉及的软件的最新版本的安装程序

开发专家之Sun ONE

精通Hibernate：Java对象持久化技术详解

hibernate

孙卫琴 编著
飞思科技产品研发中心 监制

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内容简介

Hibernate 是非常流行的对象-关系映射工具。本书详细介绍了运用目前最成熟的 Hibernate 2.1 版本进行 Java 对象持久化的技术。Hibernate 是连接 Java 对象模型和关系数据模型的桥梁，通过本书，读者不仅能掌握用 Hibernate 工具对这两种模型进行映射的技术，还能获得设计与开发 Java 对象模型和关系数据模型的先进经验。书中内容注重理论与实践相结合，列举了大量具有典型性和实用价值的 Hibernate 应用实例，并提供了详细的开发和部署步骤。随书附赠光盘内容为本书所有范例源程序，以及本书涉及的软件的最新版本的安装程序。

本书无论对于 Java 开发的新手还是行家来说，都是精通 Java 对象持久化技术的必备实用手册。

图书在版编目（CIP）数据

精通 Hibernate：Java 对象持久化技术详解 / 孙卫琴编著. —北京：电子工业出版社，2005.5
（开发专家之 Sun ONE）
ISBN 7-121-01136-0

Ⅰ. 精... Ⅱ. 孙... Ⅲ. Java 语言—程序设计 Ⅳ. TP 312

中国版本图书馆 CIP 数据核字（2005）第 036404 号

责任编辑：赵红梅
印　　刷：北京东光印刷厂
出版发行：电子工业出版社
　　北京海淀区万寿路 173 信箱　邮编：100036
经　　销：各地新华书店
开　　本：787×1092　1/16　印张：38.5　字数：985.6 千字
印　　次：2005 年 5 月第 1 次印刷
印　　数：6 000 册　　定价：59.00 元（含光盘 1 张）

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系电话：010-68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

在如今的企业级应用开发环境中，面向对象的开发方法已成为主流。众所周知，对象只能存在于内存中，而内存不能永久保存数据。如果要永久保存对象的状态，需要进行对象的持久化，即把对象存储到专门的数据存储库中。目前，关系数据库仍然是使用最广泛的数据存储库。关系数据库中存放的是关系数据，它是非面向对象的。

对象和关系数据其实是业务实体的两种表现形式。业务实体在内存中表现为对象，在数据库中表现为关系数据。内存中的对象之间存在关联和继承关系，而在数据库中，关系数据无法直接表达多对多关联和继承关系。因此，把对象持久化到关系数据库中，需要进行对象-关系的映射（Object/Relation Mapping，简称 ORM），这是一项繁琐耗时的工作。

在实际应用中，除了需要把内存中的对象持久化到数据库外，还需要把数据库中的关系数据再重新加载到内存中，以满足用户查询业务数据的需求。频繁地访问数据库，会对应用的性能造成很大影响。为了降低访问数据库的频率，可以把需要经常被访问的业务数据存放在缓存中，并且通过特定的机制来保证缓存中的数据与数据库中的数据同步。

在 Java 领域，可以直接通过 JDBC 编程来访问数据库。JDBC 可以说是访问关系数据库的最原始、最直接的方法。这种方式的优点是运行效率高，缺点是在 Java 程序代码中嵌入大量 SQL 语句，使得项目难以维护。在开发企业级应用时，可以通过 JDBC 编程来开发单独的持久化层，把数据库访问操作封装起来，提供简洁的 API，供业务层统一调用。但是，如果关系数据模型非常复杂，那么直接通过 JDBC 编程来实现持久化层需要有专业的知识。对于企业应用的开发人员，花费大量时间从头开发自己的持久化层不是很可行。

幸运的是，目前在持久化层已经有好多种现成的持久化中间件可供选用，有些是商业性的，如 TopLink；有些是非商业性的，如 JDO 和 Hibernate。Hibernate 是一个基于 Java 的开放源代码的持久化中间件，它对 JDBC 做了轻量级封装，不仅提供 ORM 映射服务，还提供数据查询和数据缓存功能，Java 开发人员可以方便地通过 Hibernate API 来操纵数据库。

现在，越来越多的 Java 开发人员把 Hibernate 作为企业应用和关系数据库之间的中间件，以节省和对象持久化有关的 30%的 JDBC 编程工作量。2005 年，Hibernate 作为优秀的类库和组件，荣获了第 15 届 Jolt 大奖。Hibernate 之所以能够流行，归功于它的以下优势：

（1）它是开放源代码的，允许开发人员在需要的时候研究源代码，改写源代码，定制客户化功能。

（2）具有详细的参考文档。

（3）对 JDBC 仅做了轻量级封装，必要的话，用户还可以绕过 Hibernate，直接访问 JDBC API。

（4）具有可扩展性。

（5）使用方便，容易上手。

（6）Hibernate 既适用于独立的 Java 程序，也适用于 Java Web 应用，而且还可以在 J2EE 架构中取代 CMP（Container-managered Persistence，由容器管理持久化），完成对象持久化

的重任，Hibernate 能集成到会话 EJB 和基于 BMP 的实体 EJB 中，BMP（Bean- managered Persistence）是指由实体 EJB 本身管理持久化。本书以 netstore 应用为例，介绍了把 Hibernate 集成到会话 EJB 中的方法。

（7）Hibernate 可以和多种 Web 服务器、应用服务器良好集成，并且支持几乎所有流行的数据库服务器。

本书结合大量典性的实例，详细介绍了运用目前最成熟的 Hibernate 2.1 版本进行 Java 对象持久化的技术。Hibernate 是连接 Java 对象模型和关系数据模型的桥梁，通过本书，读者不仅能掌握用 Hibernate 工具对这两种模型进行映射的技术，还能获得设计与开发 Java 对象模型和关系数据模型的先进经验。

本书的组织结构和主要内容

本书按照由浅入深、前后照应的顺序来安排内容，主要包含以下内容。

1．Hibernate 入门

第 1 章和第 2 章为入门篇。第 1 章概要介绍了和 Java 对象持久化相关的各种技术，详细阐述了中间件、Java 对象的持久化、持久化层、数据访问细节、ORM、域模型和关系数据模型等概念。

第 2 章以一个 Hibernate 应用实例——helloapp 应用为例，引导读者把握设计、开发和部署 Hibernate 应用的整体流程，理解 Hibernate 在分层的软件结构中所处的位置。

对于已经在 Java 对象持久化领域有一定工作经验的开发人员，可以从第 1 章入手，高屋建瓴地把握持久化领域的各种理论，对于新手，不妨先阅读第 2 章，以便快速获得开发 Hibernate 应用的实际经验。

2．Hibernate 工具

第 3 章和附录 C 介绍了 Hibernate 的一些代码转换工具的用法，例如，hbm2java 工具能根据映射文件自动生成 Java 源文件，hbm2ddl 功能能根据映射文件自动生成数据库 Schema。

3．对象-关系映射技术

本书重点介绍的内容就是如何运用 Hibernate 工具，把对象模型映射到关系数据模型，相关章节包括：

第 4 章：介绍对象-关系映射的基础知识。

第 5 章：介绍对象标识符的映射方法。

第 6 章：介绍一对多关联关系的映射方法。

第 8 章：介绍组成关系的映射方法。

第 9 章：介绍 Java 类型、SQL 类型和 Hibernate 映射类型之间的对应关系。

第 14 章：介绍继承关系的映射方法。

第 15 章：介绍了 Java 集合类的用法，这一章主要是为第 16 章做铺垫的。

第 16 章：介绍 Java 集合的映射方法。

第 17 章：介绍一对一和多对多关联关系的映射方法。

4．通过 Hibernate API 操纵数据库

第 7 章介绍了运用 Hibernate API 来保存、更新、删除、加载或查询 Java 对象的方法，并介绍了 Java 对象在持久化层的三种状态：临时状态、持久化状态和游离状态。深入理解 Java 对象的三种状态及状态转化机制，是编写健壮的 Hibernate 应用程序的必要条件。

5．Hibernate 的检索策略和检索方式

第 10 章介绍了 Hibernate 的各种检索策略，对每一种检索策略，都介绍了它的适用场合。第 11 章详细介绍了 HQL 查询语句的语法，以及 QBC API 的使用方法。合理运用 Hibernate 的检索策略及检索技巧，是提高 Hibernate 应用性能的重要手段。

6．数据库事务、并发、缓存与性能优化

第 12 章先介绍了数据库事务的概念，接着介绍了运用 Hibernate API 来声明事务边界的方法，接着介绍在并发环境中出现的各种并发问题，然后介绍了采用 Hibernate 的悲观锁，以及版本控制功能来避免并发问题的方法。

第 13 章介绍了 Hibernate 的二级缓存机制，并介绍了如何根据实际需要来配置 Hibernate 的第二级缓存，以提高应用的性能。

7．Hibernate 高级配置

第 18 章介绍了 Hibernate 应用的两种运行环境：受管理环境与不受管理环境，然后介绍了在这两种环境中配置数据库连接池及 SessionFactory 实例的方法。

8．综合实例

第 19 章和第 20 章介绍了一个名为 netstore 应用的电子商务网站的实例，netstore 应用利用 Struts 作为 Java Web 框架，用 Hibernate 来完成对象持久化的任务，并且分别用普通的 JavaBean 及 EJB 组件来实现业务逻辑。

9．附录

本书的附录介绍了标准 SQL 语言的主要用法、Java 的反射机制、XDoclet 工具的用法，以及 netstore 应用的发布和运行过程。在介绍标准 SQL 语言和 Java 反射机制时，都不是泛泛而谈，而是有针对性地介绍了与 Hibernate 紧密相关的知识，如 SQL 连接查询，以及运用 Java 反射机制来实现持久化中间件的基本原理。

本书的范例程序

为了使读者不但能掌握用 Hibernate 来持久化 Java 对象的理论，并且能迅速获得开发 Hibernate 应用的实际经验，彻底掌握并会灵活运用 Hibernate 技术，本书为每一章都提供了

完整的 Hibernate 应用范例，在本书附赠光盘中包含了所有范例源文件。

为了方便初学者顺利地运行本书的范例，光盘上提供的所有范例程序都是可运行的。读者只要把它们拷贝到本地机器上，就能够运行，不需要再做额外的配置。此外，在每个范例的根目录下还提供了 ANT 工具的工程文件 build.xml，它用于编译和运行范例程序。

本书最后还提供了一个完整的 netstore 应用例子，它实现了一个购物网站，更加贴近实际应用。本书以 netstore 应用为例，介绍了软件的 MVC 框架，控制层与模型层之间通过游离对象来传输数据的方式，以及模型层采用合理的检索策略来控制检索出来的对象图的深度，从而优化应用的性能的技巧。

这本书是否适合您

把 Java 对象持久化到关系数据库，几乎是所有企业 Java 应用必不可少的重要环节，因此本书适用于所有从事开发 Java 应用的读者。Hibernate 是 Java 应用和关系数据库之间的桥梁，阅读本书，要求读者具备 Java 语言和关系数据库的基础知识。

如果您是开发 Hibernate 应用的新手，建议按照本书的先后顺序来学习。您可以先从简单的 Hibernate 应用实例下手，把握开发 Hibernat 应用的大致流程，然后逐步深入地了解把对象模型映射到关系数据模型的各种细节。

如果您已经在开发 Hibernate 应用方面有着丰富的经验，则可以把本书作为实用的 Hibernate 技术参考资料。本书深入探讨了把复杂的对象模型映射到关系数据模型的各种映射方案，详细介绍了 Hibernate 的 HQL 查询语言的用法，并且介绍了优化 Hibernate 应用性能的各种手段，如选择恰当的检索策略和事务隔离级别，以及运用版本控制和 Hibernate 的第二级缓存等。灵活运用本书介绍的 Hibernate 最新技术，将使您开发 Hibernate 应用更加得心应手。

实践是掌握 Hibernate 的好方法。为了让读者彻底掌握并学会灵活运用 Hibernate，本书为每一章都提供了典型的范例，在本书配套光盘上提供了完整的源代码，以及软件安装程序。建议读者在学习 Hibernate 技术的过程中，善于将理论与实践相结合，达到事半功倍的效果。

光盘使用说明

本书配套光盘包含以下目录。

1．software 目录

在该目录下包含了本书内容涉及的所有软件的最新版本的安装程序，包括：

（1）Hibernate 软件包（Hibernate 2.1.7）。

（2）Hibernate 的扩展软件包（Hibernate-extensions 2.1.3）。

（3）MySQL 服务器的安装软件（MySQL 5.0.2）。

（4）MySQL 的驱动程序（mysql-connector-java-3.1.7）。

（5）ANT 的安装软件（Ant 1.5.4）。
（6）Tomcat 的安装软件（Tomcat 5.0.24）。
（7）Struts 软件（Struts 1.1）。
（8）JBoss 与 Tomcat 的集成软件（Jboss-3.2.1_tomcat-4.1.24）。
（9）XDoclet 软件包（XDoclet1.2.2）。

2．sourcecode 目录

在该目录下提供了本书所有的源程序。

写作规范

为了节省文章的篇幅，在本书中显示范例的源代码时，有时做了一些省略。对于 Java 类，省略显示 package 语句和 import 语句。除了 netstore 应用外，本书其他范例创建的 Java 类都位于 mypack 包下。对于持久化类，还省略显示了属性的 getXXX()和 setXXX()方法。对于对象-关系映射文件，省略显示开头的<?xml>和<!DOCTYPE>元素。在配套光盘中可获得完整的源代码。

在本书提供的 SQL 语句中，表名和字段名都采用大写形式，而 SQL 关键字，如 select、from、insert、update 和 delete 等，都采用小写形式。

在本书中，有时把运用了 Hibernate 技术的 Java 应用简称为 Hibernate 应用。此外，对象和实例是相同的概念；覆盖方法、重新定义方法，以及重新实现方法是相同的概念；继承和扩展是相同的概念；表的记录和表的数据行是相同的概念；表的字段和表的数据列是相同的概念；查询与检索是相同的概念；持久化类和 POJO 都是指其实例需要被持久化的基于 JavaBean 形式的实体域对象；对象-关系映射文件和映射文件是相同的概念；本书中的应用服务器主要指 J2EE 服务器。

本书在编写过程中得到了 Hibernate 软件组织和 SUN 公司在技术上的大力支持，飞思科技产品研发中心负责监制工作，此外孙璐为本书的编写提供了有益的帮助，在此表示衷心的感谢！尽管我们尽了最大努力，但本书难免会有不妥之处，欢迎各界专家和读者朋友批评指正。

我们的联系方式如下：
咨询电话：（010）68134545　　68131648
答疑邮件：support@fecit.com.cn
服务网址：http://www.fecit.com.cn　　http://www.fecit.net
通用网址：计算机图书、FECIT、飞思教育、飞思科技、飞思

编　著　者
飞思科技产品研发中心

第 1 章　Java 对象持久化技术概述

Hibernate 是什么？从不同的角度有不同的解释：

- 它是连接 Java 应用程序和关系数据库的中间件。
- 它对 JDBC API 进行了封装，负责 Java 对象的持久化。
- 在分层的软件架构中它位于持久化层，封装了所有数据访问细节，使业务逻辑层可以专注于实现业务逻辑。
- 它是一种 ORM 映射工具，能够建立面向对象的域模型和关系数据模型之间的映射。

这样的解释不一定会让初学者满意，因为这随之会带来一系列新的疑问：什么是中间件？什么是 Java 对象的持久化？是什么是持久化层？什么是数据访问细节？什么是 ORM？什么是域模型？什么是关系数据模型？

由此可见，在介绍 Hibernate 之前，有必要先解释和 Java 对象持久化相关的各种技术和术语。本章首先介绍了分层的应用程序结构，探讨了为软件分层的基本原理，然后介绍了 Hibernate 在分层软件中所处的位置：它位于持久化层。

本章接着介绍了软件的三种模型：概念模型、域模型和数据模型，然后介绍了 Java 对象的持久化概念，并介绍了实现对象持久化的几种模式：

- 业务逻辑和数据访问耦合
- 主动域对象模式
- ORM 模式
- JDO 模式
- CMP 模式

1.1　应用程序的分层体系结构

纵观 40 多年来计算机应用软件的演变过程，可以看出，应用程序逐渐由单层体系结构发展为多层体系结构。最初的应用软件只是在大型机上的单层应用程序，许多程序采用文件系统来存储数据。20 世纪 70 年代数据库得到普及，20 世纪 80 年代 PC 和局域网的出现使数据库技术飞速发展，原来的单层应用发展为双层应用，如图 1-1 所示。

在双层应用中，数据库层存放持久性业务数据，应用程序作为单独的一层，在这个层中负责生成用户界面的代码和负责业务逻辑的代码混合在一起。例如，在同一个 JSP 文件中，既包含生成动态网页的代码，还包含响应用户请求，完成相应业务逻辑的代码。由于界面代码与业务逻辑代码掺杂在一起，使程序结构不清晰，而且维护很困难。对于大型复杂的应用软件，这一问题显得尤为突出。在这种环境下，三层结构应运而生，它把原来的应用程序层划分为表述层和业务逻辑层，如图 1-2 所示。

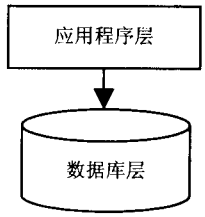

图 1-1 双层应用程序

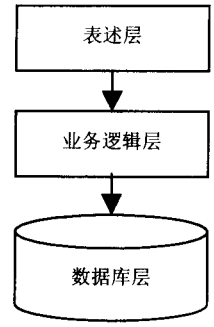

图 1-2 三层应用程序

三层结构是目前典型的一种应用软件的结构。

- 表述层：提供与用户交互的界面。GUI（图形用户界面）和 Web 页面是表述层的两个典型的例子。
- 业务逻辑层：实现各种业务逻辑。例如当用户发出生成订单的请求时，业务逻辑层负责计算订单的价格、验证订单的信息，以及把订单信息保存到数据库中。
- 数据库层：负责存放和管理应用的持久性业务数据。例如对于电子商务网站应用，在数据库中保存了客户、订单和商品等业务数据。关系数据库依然是目前最流行的数据库。

1.1.1 区分物理层和逻辑层

软件的分层包含两种含义：一种是物理分层，即每一层都运行在单独的机器上，这意味着创建分布式的软件系统；一种是逻辑分层，指的是在单个软件模块中完成特定的功能。例如在图 1-3 中，业务逻辑层和数据库层运行在同一台机器上，这台机器既是应用服务器，又是数据库服务器，因此整个系统物理上为两层结构，而逻辑上为三层结构。

在本书中，如果没有特别说明，软件分层均指的是逻辑分层。

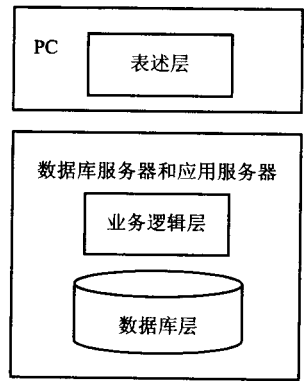

图 1-3 区分物理层和逻辑层

1.1.2　软件层的特征

由于每个软件都有自身的特点，因此不可能提供一个适合于所有软件的体系结构。但总的说来，软件的层必须符合以下特征：

- 每个层由一组相关的类或组件（如 EJB）构成，共同完成特定的功能。
- 层与层之间存在自上而下的依赖关系，即上层组件会访问下层组件的 API，而下层组件不应该依赖上层组件。例如：表述层依赖于业务逻辑层，而业务逻辑层依赖于数据库层。
- 每个层对上层公开 API，但具体的实现细节对外透明。当某一层的实现发生变化，只要它的 API 不变，不会影响其他层的实现。

软件分层的一个基本特征就是层与层之间存在自上而下的依赖关系，例如以图 1-4 所示把电子商务网站系统按照业务功能划分为客户管理模块、订单管理模块和库存管理模块。这几个模块之间为并列关系，不存在自上而下的依赖关系，因此不算分层的结构。但是每个模块都可划分为表述层、业务逻辑层和数据库层，这两种划分方式是正交的。

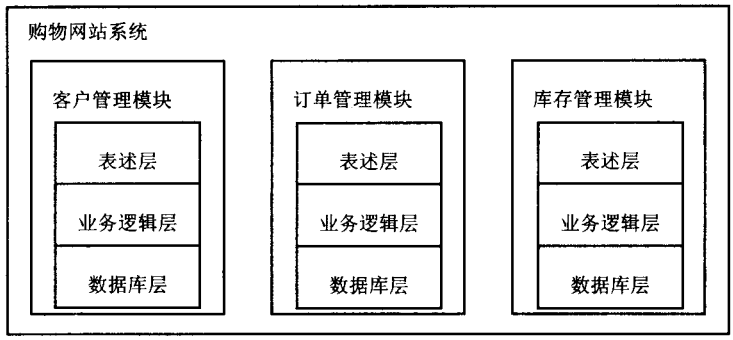

图 1-4　按业务功能划分应用的模块

每个软件层都向上公开接口，封装实现细节，如图 1-5 所示。

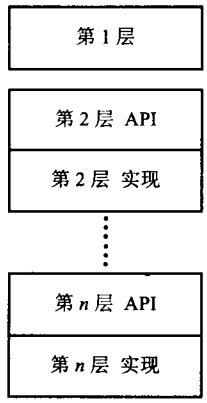

图 1-5　软件层向上公开接口，封装实现细节

由于软件上层总是依赖软件下层，因此也可以把软件上层称为客户程序。例如表述层