对J2EE中的DAO组件编写单元测试 PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/252/2021_2022__E5_AF_B9J2EE_E4_B8_AD_c104_252495.htm 单元测试作为保证软件质量及重构的基础，早已获得广大开发人员的认可。单元测试是一种细粒度的测试，越来越多的开发人员在提交功能模块时也同时提交相应的单元测试。对于大多数开发人员来讲，编写单元测试已经成为开发过程中必须的流程和最佳实践。 对普通的逻辑组件编写单元测试是一件容易的事情，由于逻辑组件通常只需要内存资源，因此，设置好输入输出即可编写有效的单元测试。对于稍微复杂一点的组件，例如Servlet，我们可以自行编写模拟对象，以便模拟HttpRequest和HttpResponse等对象，或者，使用EasyMock之类的动态模拟库，可以对任意接口实现相应的模拟对象，从而对依赖接口的组件进行有效的单元测试。 在J2EE开发中，对DAO组件编写单元测试往往是一件非常复杂的任务。和其他组件不通，DAO组件通常依赖于底层数据库，以及JDBC接口或者某个ORM框架（如Hibernate），对DAO组件的测试往往还需引入事务，这更增加了编写单元测试的复杂性。虽然使用EasyMock也可以模拟出任意的JDBC接口对象，或者ORM框架的主要接口，但其复杂性往往非常高，需要编写大量的模拟代码，且代码复用度很低，甚至不如直接在真实的数据库环境下测试。不过，使用真实数据库环境也有一个明显的弊端，我们需要准备数据库环境，准备初始数据，并且每次运行单元测试后，其数据库现有的数据将直接影响到下一次测试，难以实现“ 即时运行，反复运行 ”单元测试的良好实

践。 本文针对DAO组件给出一种较为合适的单元测试的编写策略。在JavaEE开发网的开发过程中，为了对DAO组件进行有效的单元测试，我们采用HSQLDB这一小巧的纯Java数据库作为测试时期的数据库环境，配合Ant，实现了自动生成数据库脚本，测试前自动初始化数据库，极大地简化了DAO组件的单元测试的编写。 在Java领域，JUnit作为第一个单元测试框架已经获得了最广泛的应用，无可争议地成为Java领域单元测试的标准框架。本文以最新的JUnit 4版本为例，演示如何创建对DAO组件的单元测试用例。 JavaEEdev的持久层使用Hibernate 3.2，底层数据库为MySQL。为了演示如何对DAO进行单元测试，我们将其简化为一个DAOTest工程： 由于将Hibernate的Transaction绑定在Thread上，因此，HibernateUtil类负责初始化SessionFactory以及获取当前的session： public class HibernateUtil { private static final SessionFactory sessionFactory. static { try { sessionFactory = new AnnotationConfiguration() .configure() .buildsessionFactory(). } catch(Exception e) { throw new ExceptionInInitializerError(e). } } public static Session getCurrentsession() { return sessionFactory.getCurrentsession(). } } 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com