在Weblogic上配置Hibernate为JNDI 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_9C_A8Weblogi_c104_252493.htm 一、首先需要把Hibernate 用到的jar包和配置文件都放到Weblogic能够搜索到的CLASSPATH路径上。单单这一步就有很多人很迷茫，其实去仔细看看Weblogic的启动脚本文件startWeblogic.cmd和startWLS.cmd，我想大部分人都知道该怎么配置了。 我机器上的有个Hibernate的项目，在D:\test\Oracle目录下，该目录下的结构是：D:\test\Oracle\lib 放置hibernate的所有jar包D:\test\Oracle\src 放置源代码D:\test\Oracle\classes 编译好的代码和hibernate的配置文件(hibernate.properties, log4j.properties, cache.ccf) 现在需要把D:\test\oracle\lib目录下那些jar文件和D:\test\Oracle\classes目录都放置到Weblogic的 CLASSPATH里面去，所以修改mydomain里面的Weblogic启动脚本startWeblogic.cmd，在启动Weblogic之前，插入设置CLASSPATH的命令，如下：@rem set hibernate classpathset HIBERNATE_LIB=D:\test\Oracle\libset HIBERNATE_CLASSES=D:\test\Oracle\classesset CLASSPATH=%CLASSPATH%.%HIBERNATE_LIB%\cglib-asm.jar.%HIBERNATE_LIB%\commons-beanutils.jar.%HIBERNATE_LIB%\commons-collections.jar.%HIBERNATE_LIB%\commons-lang.jar.%HIBERNATE_LIB%\commons-logging.jar.%HIBERNATE_LIB%\dom4j-full.jar.%HIBERNATE_LIB%\hibernate2.jar.%HIBERNATE_LIB%\jcs.jar.%HIBERNATE_LIB%\log4j-1.2.8.jar.%HIBERNATE_LIB%\odmg.jar.%HIBERNATE_LIB%\jta.jar.%HIBE

RNATE_CLASSES%. 下面一行，就是本来脚本里面的启动命令：@rem Call Weblogic Servercall "C:\bea\weblogic700\server\bin\startWLS.cmd" 二、在Weblogic上配置 Oracle数据库的连接池。这一步本来和Hibernate无关，但是如果你想要使用EJB，想要使用JTA，那么必须使用Weblogic提供的连接池，而不能使用Hibernate自带的连接池，或者其它第三方连接池，否则容器将无法管理数据库事务。这一步很简单，就是在Weblogic Console里面配置Connection Pool和TxData Source，我的TxDataSource取名称为“ mypool ”三、修改hibernate.properties。使用Weblogic的连接池，而不是自带的连接池。我修改的是D:\test\Oracle\classes\hibernate.properties，增加如下行：hibernate.dialect net.sf.hibernate.dialect.OracleDialecthibernate.connection.datasource mypoolhibernate.connection.provider_class net.sf.hibernate.connection.DatasourceConnectionProviderhibernate.session_factory_name hibernate.session_factory 注意最后一行，这是使用 Hibernate来绑定JNDI给JNDI起的名称，本来应该是hibernate/session_factory，但是Weblogic要求改为. 号，不过在程序中lookup的时候还是要写hibernate/session_factory 另外提到一点的是hibernate.jdbc.fetch_size 50hibernate.jdbc.batch_size 25 分别对数据库查询和插入有很大的性能影响，调节这两个选项可以得到最好的性能。 为了保证sessionFactory实例的预创建，使用Weblogic的T3StartUpDef接口创建一个StartUp类，在Weblogic启动的时候运行：package com.Javaeye.import Java.util.Hashtable.import weblogic.common.T3StartupDef.import

weblogic.common.T3ServicesDef.import net.sf.hibernate.cfg.Configuration.import net.sf.hibernate.sessionFactory.public class HibernateStartUp implements T3StartupDef { public void setServices(T3ServicesDef services) {} public String startup(String name, Hashtable args) throws Exception { SessionFactory sf = new Configuration().configure().buildsessionFactory(). return "Hibernate Startup completed successfully". } } 代码非常简单，其实就是确保预先运行SessionFactory sf = new Configuration().configure().buildsessionFactory(). 把sf创建出来，而Hibernate会自行调用一系列类方法，把sf绑定到Weblogic的的JNDI树下的hibernate/session_factory路径中。 4、编译HibernateStartUp.Java 编译这个源代码的时候需要注意的是，要把weblogic.jar包和Hibernate所有的相关包和配置文件导入。我是把这个源代码放到D:\test\oracle\src目录下的，用早已编写好的ant脚本运行一下就编译好了，并且编译好的 class文件被放置到D:\test\Oracle\classes目录下，该目录已经被加入到Weblogic的CLASSPATH里面，因此很省事。 五、配置StartUp类 启动Weblogic，打开Console控制台，在左边的Applet树上找到StartUp &amp. Shutdown，然后在右边点击“ Configure a new Startup Class... ”，在Name框里面随便填写，在ClassName里面填写你编写的StartUp类，我填写的是com.Javaeye.HibernateStartUp，然后点击“ Apply ”。然后切换到Target这选项卡，在Target-Server左边的 Avaiable框里面选择“ myserver ”，点击右箭头，把它挪到右边的“ Chosen ”框里面去，最后再点击一下“ Apply"按钮。如果此时 Weblogic

的DOS窗口里面没有出错信息，那么应该已经配置成功了。六、现在关闭Weblogic，再重新运行 startWelogic.cmd，启动Weblogic，观察DOS窗口的输出信息，可以看到Hibernate的初始化信息一屏屏的滚动输出，证明已经配置成功。现在再打开Console控制台，点击左边Applet树中的Servers|myserver，然后可以在右边最下面找到“ View JNDI tree ”，点击它，会打开一个浏览器窗口，显示JNDI树，这时你可以看到一个名称为hibernate的JNDI对象，在左边的Applet树中点击它，看右边的详细信息，我的机器上的信息如下：Bind Name: hibernateObject Class: net.sf.hibernate.impl.sessionFactoryImplObject Hash Code: 454492Object To String: net.sf.hibernate.impl.sessionFactoryImpl@6ef5c 完全正确！ 最后你可以随意在EJB或者Servlet/JSP里面使用JND查找来获得sessionFactory了。 例如：Context ctx = new InitialContext().SessionFactory sf = (SessionFactory) ctx.lookup("hibernate/session_factory"). 请注意：上述代码只能在WebLogic容器内运行，而不能在WebLogic容器外运行。因为SessionFactory并没有实现序列化接口，因此当客户端程序(在另一个单独的JVM中运行)远程访问WebLogic JNDI，企图将SessionFactory序列化到本地，肯定会失败。但即使sessionFactory实现序列化接口，由于它不是一个可以支持RMI的对象，仍然无法在WebLogic容器外正常调用。 与此不同的是，WebLogic本身的DataSource，EJB，JMS等等都是支持RMI的(前提条件是WebLogic相应的jar要有)，所以你可以在WebLogic外面lookup，并且使用它。 100Test 下载频道开通