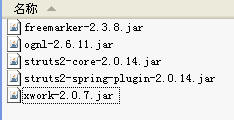
[第一步: 配置Struts 1](#_Toc248824643)

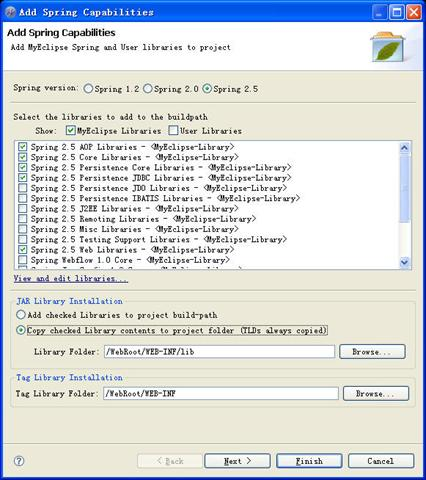
[第二部: 配置Spring 1](#_Toc248824644)

[第三步: 配置Hibernate 3](#_Toc248824645)

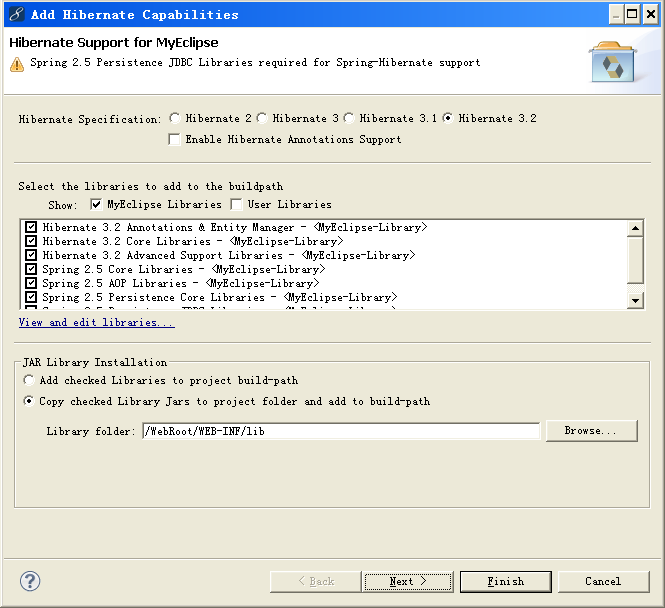
# 第一步: 配置Struts

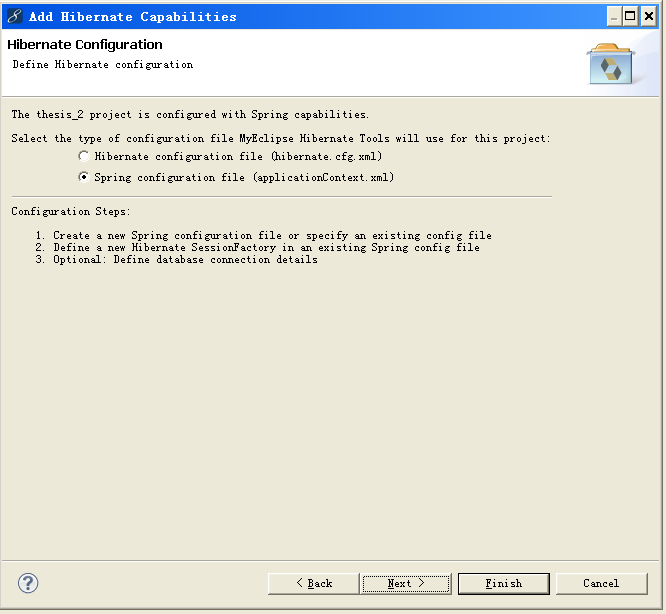


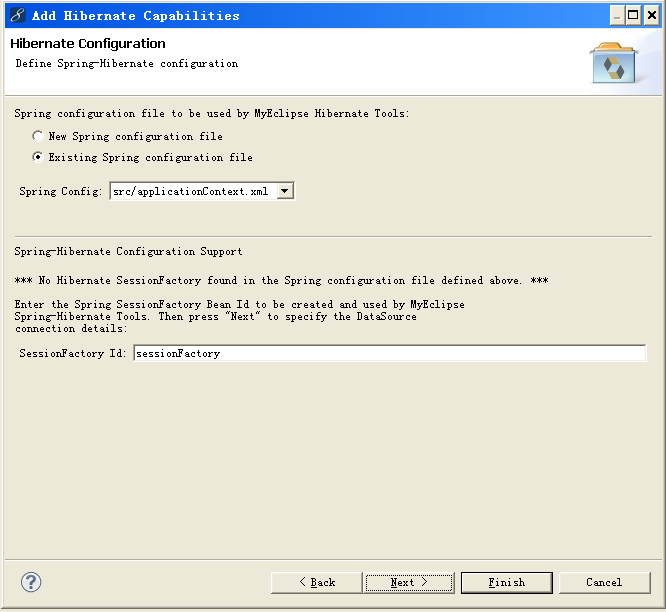
# 第二部: 配置Spring

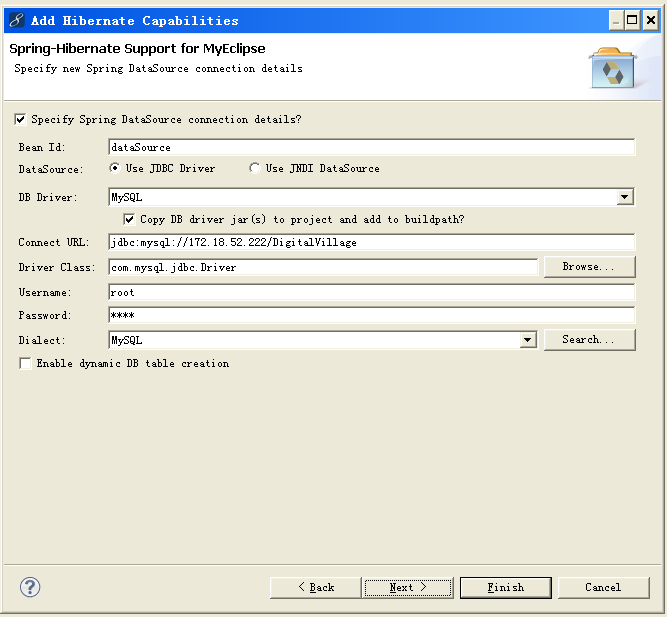


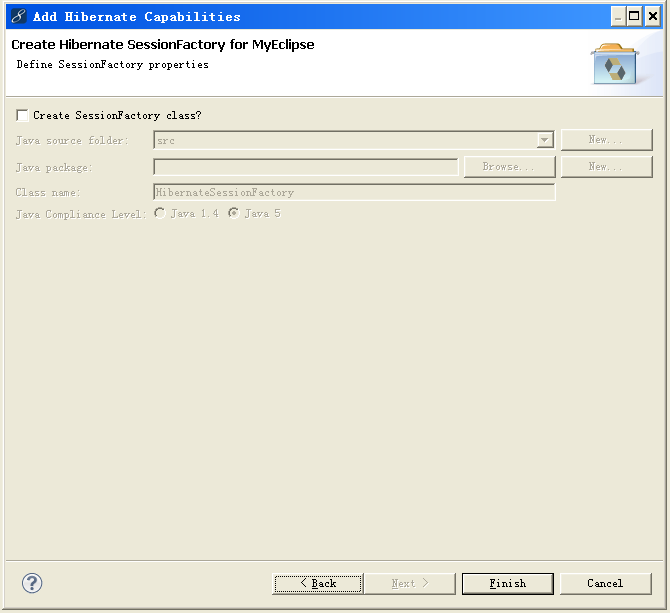
# 第三步: 配置Hibernate











# 第四步: 编写配置文件

创建struts.xml文件，在src包下新建一个struts.xml文件，该文件内容如下：

<!DOCTYPE struts PUBLIC  
        "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"  
        "<http://struts.apache.org/dtds/struts-2.0.dtd>">

<struts>

<!--下述语句是将struts2交给spring管理 -->

<constant name="struts.objectFactory" value="spring" />

<!--导入struts-default.xml文件 -->

<include file="struts-default.xml"></include>

<!--将action内容放在package元素下，package元素的name值与extends值可以使用默认值，如下所示 -->

<package name="default" extends="struts-default">

<!--例如与用户登录相关的LoginAction，配置如下，其中，class值与applicationContext.xml文件中该action的id值相对应，name值作为标识-->

   <action name="loginAction" class="LoginAction">

<!--result元素设置页面跳转值，其中name值为目标页面的字符串标识，如下，登录失败时回到login.jsp页面，成功时，转向success.jsp页面 -->

    <result name="fail">/login.jsp </result>

    <result name="success">/success.jsp</result>

   </action>

</package>

</struts>

6.配置web.xml文件，其内容如下：

<?xml version="1.0" encoding="UTF-8"?>  
<web-app version="2.5"   
xmlns="<http://java.sun.com/xml/ns/javaee>"   
xmlns:xsi="<http://www.w3.org/2001/XMLSchema-instance>"   
xsi:schemaLocation="<http://java.sun.com/xml/ns/javaee>   
<http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd>">

<context-param>  
<param-name>contextConfigLocation</param-name>  
<param-value>/WEB-INF/applicationContext.xml</param-value>  
</context-param>

<filter>  
<filter-name>struts2</filter-name>  
<filter-class>org.apache.struts2.dispatcher.FilterDispatcher</filter-class>  
</filter>

<filter-mapping>  
<filter-name>struts2</filter-name>  
<url-pattern>/\*</url-pattern>  
</filter-mapping>

<welcome-file-list>  
<welcome-file>index.jsp</welcome-file>  
</welcome-file-list>

<listener>  
<listener-class>  
   org.springframework.web.context.ContextLoaderListener  
</listener-class>  
</listener>

</web-app>

7.反向生成实体类文件

在src包下新建一个entity包，将前面数据源xiaoqi所示数据库bookshop中的user表反向生成Hibernate实体类文件。

8.创建数据操作文件

在src包下创建dao包，在该包下新建类文件PersonManage.java，其内容如下所示：

package dao;

import java.util.\*;

import org.springframework.orm.hibernate3.support.HibernateDaoSupport;

import entity.\*;

public class PersonManage extends HibernateDaoSupport {

//普通用户登录验证  
@SuppressWarnings("unchecked")  
public boolean checkUser(String userName,String userPassword){  
   boolean flag = false;  
   String hql = "from User as user where user.userName = '"+userName+"' and user.userPassword = '"+userPassword+"'";  
   List<User> userList = this.getHibernateTemplate().find(hql);  
   if(userList.size()>0){  
    flag = true;  
   }  
   return flag;  
}

}

9.创建Action

在src包下新建一个包，名为"com.xiaoqi.action"（名字仅为示例，只写一个com也可以，示例写法是采取struts中的action包名写法），在该包下新建一个action类LoginAction.java，其内容如下：

package com.xiaoqi.action;

import com.opensymphony.xwork2.ActionSupport;

import dao.PersonManage;

public class LoginAction extends ActionSupport{

private String userName;  
private String userPassword;  
private PersonManage personManage;  
public String getUserName() {  
   return userName;  
}  
public void setUserName(String userName) {  
   this.userName = userName;  
}  
public String getUserPassword() {  
   return userPassword;  
}  
public void setUserPassword(String userPassword) {  
   this.userPassword = userPassword;  
}  
public void setPersonManage(PersonManage personManage) {  
   this.personManage = personManage;  
}  
public String execute(){  
   String page = "fail";  
   boolean flag = false;  
   flag = personManage.checkUser(userName, userPassword);  
   if(flag){  
    page = "success";  
   }  
   return page;  
}  
}

10.创建页面，创建登录页面login.jsp，内容如下：

<%@ page language="java" pageEncoding="gb2312"%>

<!-- 导入struts2标签库 -->  
<[%@taglib](mailto:%25@taglib) uri="/struts-tags" prefix="s"%>

<html>   
<head>  
   <title>用户登录</title>  
</head>  
<body>  
<center>

   <div id="login">  
    <s:form action="loginAction">  
     <s:textfield label="用户名" name="userName"></s:textfield>  
     <s:password label="密码" name="userPassword"></s:password>  
     <s:submit value="登录"></s:submit>  
    </s:form>  
   </div>

</center>  
</body>  
</html>

页面中，先导入struts2标签库，用来创建form表单。form表单提交给与登录相关的LoginAction.java文件，而LoginAction.java在struts.xml文件中相应的name值为"loginAction”，所以这里的action值为"loginAction"。至于用户名文本框与密码文本框的name值则与LoginAction.java文件中的userName与userPassword完全匹配。

11.配置applicationContext.xml文件：

<?xml version="1.0" encoding="UTF-8"?>  
<beans  
xmlns="<http://www.springframework.org/schema/beans>"  
xmlns:xsi="<http://www.w3.org/2001/XMLSchema-instance>"  
xsi:schemaLocation="<http://www.springframework.org/schema/beans> <http://www.springframework.org/schema/beans/spring-beans-2.5.xsd>">

<!-- 以下为数据源配置，自动生成 -->

<bean id=*"candy"* class=*"org.apache.commons.dbcp.BasicDataSource"*>

<property name=*"driverClassName"* value=*"com.mysql.jdbc.Driver"*>

</property>

<property name=*"url"* value=*"jdbc:mysql://localhost:3306/bookstore"*>

</property>

<property name=*"username"* value=*"root"*></property>

<property name=*"password"* value=*"sa"*></property>

</bean>

<!-- 以下为sessionFactory配置，自动生成 -->

<bean id=*"sessionFactory"*

class=*"org.springframework.orm.hibernate3.LocalSessionFactoryBean"*>

<property name=*"dataSource"*>

<ref bean=*"candy"* />

</property>

<property name=*"hibernateProperties"*>

<props>

<prop key=*"hibernate.dialect"*>

org.hibernate.dialect.MySQLDialect

</prop>

</props>

</property>

<property name=*"mappingResources"*>

<list>

<value>entity/User.hbm.xml</value>

</list>

</property>

</bean>

<!-- 配置PersonManage.java类文件 -->

<bean id=*"personManage"* class=*"dao.PersonManage"* abstract=*"false"*

lazy-init=*"default"* autowire=*"default"* dependency-check=*"default"*>

<property name=*"sessionFactory"*>

<ref local=*"sessionFactory"* />

</property>

</bean>

<!-- 配置LoginAction.java类文件 -->

<bean id=*"LoginAction"* class=*"com.xiaoqi.action.LoginAction"*>

<property name=*"personManage"*>

<ref bean=*"personManage"* />

</property>

</bean>

</beans>