# SSH框架开发的基本回顾

## 技术分析之SSH三大框架需要的jar包

1. Struts2框架

\* struts-2.3.24\apps\struts2-blank\WEB-INF\lib\\*.jar -- Struts2需要的所有jar包

\* struts2-spring-plugin-2.3.24.jar ---Struts2整合Spring的插件包

2. Hibernate框架

\* hibernate-release-5.0.7.Final\lib\required\\*.jar -- Hibernate框架需要的jar包

\* slf4j-api-1.6.1.jar -- 日志接口

\* slf4j-log4j12-1.7.2.jar -- 日志实现

\* mysql-connector-java-5.1.7-bin.jar -- MySQL的驱动包

3. Spring框架

\* IOC核心包

\* AOP核心包

\* JDBC模板和事务核心包

\* Spring整合JUnit测试包

\* Spring整合Hibernate核心包

\* Spring整合Struts2核心包

## SSH三大框架需要的配置文件

1. Struts2框架

\* 在web.xml中配置核心的过滤器

<filter>

<filter-name>struts2</filter-name>

<filter-class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter</filter-class>

</filter>

<filter-mapping>

<filter-name>struts2</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

\* 在src目录下创建struts.xml，用来配置Action

2. Hibernate框架

\* 在src目录创建hibernate.cfg.xml配置文件

\* 在JavaBean所在的包下映射的配置文件

3. Spring框架

\* 在web.xml配置整合WEB的监听器

<listener>

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

</listener>

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:applicationContext.xml</param-value>

</context-param>

\* 在src目录下创建applicationContext.xml

\* 在src目录下log4j.proerties

## Spring框架整合Struts2框架

1. 导入CRM项目的UI页面，找到添加客户的页面，修改form表单，访问Action

2. 编写CustomerAction接收请求，在struts.xml中完成Action的配置

<package name="crm" extends="struts-default" namespace="/">

<action name="customer\_\*" class="com.qf.web.action.CustomerAction" method="{1}">

</action>

</package>

3. 在Action中获取到service（开发不会使用，因为麻烦）

\* 可以通过 WebApplicationContextUtils.getWebApplicationContext(ServletActionContext.getServletContext()); 来获取，但是这种方式编写代码太麻烦了！！

4. Spring整合Struts2框架的第一种方式（Action由Struts2框架来创建）

\* 因为导入的struts2-spring-plugin-2.3.24.jar 包自带一个配置文件 struts-plugin.xml ，该配置文件中有如下代码

\* <constant name="struts.objectFactory" value="spring" /> 开启一个常量，如果该常量开启，那么下面的常量就可以使用

\* struts.objectFactory.spring.autoWire = name，该常量是可以让Action的类来自动装配Bean对象！！

5. Spring整合Struts2框架的第二种方式（Action由Spring框架来创建）（推荐大家来使用的）

\* 把具体的Action类配置文件applicatonContext.xml的配置文件中，但是注意：struts.xml需要做修改

\* applicationContext.xml

\* <bean id="customerAction" class="com.qf.web.action.CustomerAction" scope="prototype">

\* struts.xml中的修改，把全路径修改成ID值

\* <action name="customer\_\*" class="customerAction" method="{1}">

\* 第二种方式需要有两个注意的地方

\* Spring框架默认生成CustomerAction是单例的，而Struts2框架是多例的。所以需要配置 scope="prototype"

\* CustomerService现在必须自己手动注入了

## Spring框架整合Hibernate框架

（带有hibernate.cfg.xml的配置文件。强调：不能加绑定当前线程的配置）

1. 编写CustomerDaoImpl的代码，加入配置并且在CustomerServiceImpl中完成注入

2. 编写映射的配置文件，并且在hibernate.cfg.xml的配置文件中引入映射的配置文件

3. 在applicationContext.xml的配置文件，配置加载hibernate.cfg.xml的配置

<bean id="sessionFactory" class="org.springframework.orm.hibernate5.LocalSessionFactoryBean">

<property name="configLocation" value="classpath:hibernate.cfg.xml"/>

</bean>

4. 在CustomerDaoImpl中想完成数据的添加，Spring框架提供了一个HibernateDaoSupport的工具类，以后DAO都可以继承该类！！

public class CustomerDaoImpl extends HibernateDaoSupport implements CustomerDao {

public void save(Customer c) {

System.out.println("持久层...");

this.getHibernateTemplate().save(c);

}

}

<bean id="customerDao" class="com.qf.dao.CustomerDaoImpl">

<property name="sessionFactory" ref="sessionFactory"/>

</bean>

5. 开启事务的配置

\* 先配置事务管理器，注意现在使用的是Hibernate框架，所以需要使用Hibernate框架的事务管理器

<bean id="transactionManager" class="org.springframework.orm.hibernate5.HibernateTransactionManager">

<property name="sessionFactory" ref="sessionFactory"/>

</bean>

\* 开启注解事务

<tx:annotation-driven transaction-manager="transactionManager"/>

\* 在Service类中添加事务注解

@Transactional

## Spring框架整合Hibernate框架

（不带有hibernate.cfg.xml的配置文件）\*\*

1. Hibernate配置文件中

\* 数据库连接基本参数（4大参数）

\* Hibernate相关的属性

\* 连接池

\* 映射文件

2. 开始进行配置

\* 先配置连接池相关的信息

<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">

<property name="driverClass" value="com.mysql.jdbc.Driver"/>

<property name="jdbcUrl" value="jdbc:mysql:///xxx"/>

<property name="user" value="root"/>

<property name="password" value="root"/>

</bean>

\* 修改 LocalSessionFactoryBean 的属性配置，因为已经没有了hibernate.cfg.xml的配置文件，所以需要修改该配置，注入连接池

<bean id="transactionManager" class="org.springframework.orm.hibernate5.HibernateTransactionManager">

<property name="dataSource" ref="dataSource"/>

</bean>

\* 继续在 LocalSessionFactoryBean 中配置，使用hibernateProperties属性继续来配置其他的属性，注意值是properties属性文件

<!-- 配置其他的属性 -->

<property name="hibernateProperties">

<props>

<prop key="hibernate.dialect">org.hibernate.dialect.MySQLDialect</prop>

<prop key="hibernate.show\_sql">true</prop>

<prop key="hibernate.format\_sql">true</prop>

<prop key="hibernate.hbm2ddl.auto">update</prop>

</props>

</property>

<!-- 配置映射 -->

<property name="mappingResources">

<list>

<value>com/qf/domain/Customer.hbm.xml</value>

</list>

</property>

## Hibernate的模板的常用的方法

1. 增删改的操作:

\* 添加:

\* save(Object obj);

\* 修改:

\* update(Object obj);

\* 删除:

\* delete(Object obj);

2. 查询的操作:

\* 查询一条记录:

\* Object get(Class c,Serializable id);

\* Object load(Class c,Serializable id);

3. 查询多条记录:

\* List find(String hql,Object... args);

## 延迟加载问题

1. 使用延迟加载的时候，再WEB 层查询对象的时候程序会抛出异常！

\* 原因是延迟加载还没有发生SQL语句，在业务层session对象就已经销毁了，所以查询到的JavaBean对象已经变成了托管态对象！

\* 注意：一定要先删除javassist-3.11.0.GA.jar包（jar包冲突了）

2. 解决办法非常简单，Spring框架提供了一个过滤器，让session对象在WEB层就创建，在WEB层销毁。只需要配置该过滤器即可

\* 但是：要注意需要在struts2的核心过滤器之前进行配置

<filter>

<filter-name>OpenSessionInViewFilter</filter-name>

<filter-class>org.springframework.orm.hibernate5.support.OpenSessionInViewFilter</filter-class>

</filter>

<filter-mapping>

<filter-name>OpenSessionInViewFilter</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>