**Maven多模块整合S2SH**

**一、搭建s2sh-parent模块**

**1、新建s2sh-parent模块**

新建s2sh-parent项目，打包方式为pom。删除src目录。

C:\Users\zqq\AppData\Local\Temp\1569464719(1).png

**2、POM配置jar包管理及依赖**

整体框架需要用到的主jar包，在父项目中管理版本。

三个框架都要用的jar包在父项目中依赖—主要是junit和spring的jar。

Pom.xml如下：

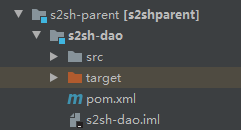
*<?*xml version="1.0" encoding="UTF-8"*?>*<**project** xmlns="http://maven.apache.org/POM/4.0.0"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <**modelVersion**>4.0.0</**modelVersion**>  
 <**groupId**>com.zqq.s2sh</**groupId**>  
 <**artifactId**>s2sh-parent</**artifactId**>  
 <**version**>1.0-SNAPSHOT</**version**>  
 <**packaging**>pom</**packaging**>  
 <**properties**>  
 <**struts2.version**>2.5.8</**struts2.version**>  
 <**spring.version**>5.0.7.RELEASE</**spring.version**>  
 <**hibernate.version**>5.3.1.Final</**hibernate.version**>  
 </**properties**>  
 <**dependencyManagement**>  
 <**dependencies**>  
 <**dependency**>  
 <**groupId**>junit</**groupId**>  
 <**artifactId**>junit</**artifactId**>  
 <**version**>4.12</**version**>  
 <**scope**>test</**scope**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.apache.struts</**groupId**>  
 <**artifactId**>struts2-core</**artifactId**>  
 <**version**>${struts2.version}</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.apache.struts</**groupId**>  
 <**artifactId**>struts2-spring-plugin</**artifactId**>  
 <**version**>${struts2.version}</**version**>  
 </**dependency**>  
 <!--配置struts2注解需要的jar包-->  
 <**dependency**>  
 <**groupId**>org.apache.struts</**groupId**>  
 <**artifactId**>struts2-convention-plugin</**artifactId**>  
 <**version**>${struts2.version}</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-web</**artifactId**>  
 <**version**>${spring.version}</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-context</**artifactId**>  
 <**version**>${spring.version}</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-test</**artifactId**>  
 <**version**>${spring.version}</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-aspects</**artifactId**>  
 <**version**>${spring.version}</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-orm</**artifactId**>  
 <**version**>${spring.version}</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.hibernate</**groupId**>  
 <**artifactId**>hibernate-core</**artifactId**>  
 <**version**>${hibernate.version}</**version**>  
 </**dependency**>  
 </**dependencies**>  
 </**dependencyManagement**>  
 <!--几个模块都要用的jar-->  
 <**dependencies**>  
 <**dependency**>  
 <**groupId**>junit</**groupId**>  
 <**artifactId**>junit</**artifactId**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-context</**artifactId**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-test</**artifactId**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-aspects</**artifactId**>  
 </**dependency**>

<**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-orm</**artifactId**>  
 </**dependency**>

</**dependencies**>  
</**project**>

**二、搭建s2sh-dao模块**

**1、新建s2sh-dao模块**



**2、配置pom依赖**

<**parent**>  
 <**artifactId**>s2sh-parent</**artifactId**>  
 <**groupId**>com.zqq.s2sh</**groupId**>  
 <**version**>1.0-SNAPSHOT</**version**>  
 </**parent**>  
 <**modelVersion**>4.0.0</**modelVersion**>  
 <**artifactId**>s2sh-dao</**artifactId**>  
 <**dependencies**>  
 <**dependency**>  
 <**groupId**>org.hibernate</**groupId**>  
 <**artifactId**>hibernate-core</**artifactId**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>mysql</**groupId**>  
 <**artifactId**>mysql-connector-java</**artifactId**>  
 <**version**>6.0.6</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>com.mchange</**groupId**>  
 <**artifactId**>c3p0</**artifactId**>  
 <**version**>0.9.5.2</**version**>  
 </**dependency**>  
 </**dependencies**>

**3、准备实体类和dao类**

User.java和UserDao.java

**4、编辑orm映射配置文件**

User.hbm.xml

<**hibernate-mapping**>  
 <**class** name="com.zqq.s2sh.domain.User" table="user">  
 <**id** name="userId" column="userid">  
 <**generator** class="native"/>  
 </**id**>

<**property** name="userName" column="username"/>  
 <**property** name="password" column="password"/>  
 </**class**>  
</**hibernate-mapping**>

**5、配置hibernate.cfg.xml**

<**hibernate-configuration**>  
 <**session-factory**>  
 <**property** name="hibernate.dialect">org.hibernate.dialect.MySQL5Dialect</**property**>  
 <**property** name="hibernate.show\_sql">true</**property**>  
 <**property** name="hibernate.format\_sql">false</**property**>  
 <**property** name="hibernate.hbm2ddl.auto">update</**property**>  
 <**property** name="hibernate.current\_session\_context\_class">org.springframework.orm.hibernate5.SpringSessionContext</**property**>  
 <**mapping** resource="User.hbm.xml"/>

</**session-factory**>  
</**hibernate-configuration**>

**6、配置applicationContext.dao.xml**

主要配置dao和jdbc的配置。后面hibernate的配置也可以全部合过来

<**bean** id="userDao" class="com.zqq.s2sh.dao.impl.UserDaoImpl">  
 <**property** name="hibernateTemplate" ref="hibernateTemplate"/>  
 </**bean**>  
 <**bean** id="hibernateTemplate" class="org.springframework.orm.hibernate5.HibernateTemplate">  
 <**property** name="sessionFactory" ref="sessionFactory"/>  
 </**bean**>  
 <**bean** id="sessionFactory" class="org.springframework.orm.hibernate5.LocalSessionFactoryBean">  
 <**property** name="dataSource" ref="dataSource"/>  
 <**property** name="configLocation" value="classpath:hibernate.cfg.xml"/>  
 </**bean**>  
 <!--数据源 也可以配置到hibernate中，后面都要和过来-->  
 <**bean** id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">  
 <**property** name="driverClass" value="com.mysql.cj.jdbc.Driver"/>  
 <**property** name="jdbcUrl" value="jdbc:mysql://127.0.0.1:3306/zqqdb?serverTimezone=UTC"/>  
 <**property** name="user" value="zqq"/>  
 <**property** name="password" value="123"/>  
 </**bean**>

**7、测试dao部署情况**

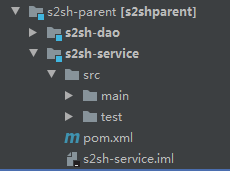
@RunWith(**SpringJUnit4ClassRunner**.**class**)  
@ContextConfiguration("classpath:application.dao.xml")  
**public class DaoTest** {  
 @Resource(name="userDao")  
 **private UserDaoImpl** userDao ;  
 @Test  
 **public void** test(){  
 //userDao.saveUser(**new** User());

System.out.printlin(userDao);  
 }  
}

由于还没有配置事务，此时只能测试spring的注入和数据源的配置是否通。

**三、搭建s2sh-service模块**

**1、新建s2sh-service模块**



**2、配置pom依赖**

<**parent**>  
 <**artifactId**>s2sh-parent</**artifactId**>  
 <**groupId**>com.zqq.s2sh</**groupId**>  
 <**version**>1.0-SNAPSHOT</**version**>  
 </**parent**>  
 <**modelVersion**>4.0.0</**modelVersion**>  
 <**artifactId**>s2sh-service</**artifactId**>  
 <**dependencies**>  
 <**dependency**>  
 <**groupId**>com.zqq.s2sh</**groupId**>  
 <**artifactId**>s2sh-dao</**artifactId**>  
 <**version**>${project.version}</**version**>  
 </**dependency**>  
 </**dependencies**>

**3、准备service类**

UserService.java

**4、配置spring（此时需要配置事务）**

配置事务：1、配置事务管理器；2、配置事务通知、3、配置切点关联

<**import** resource="classpath:applicationContext.dao.xml"/>  
 <**bean** id="userService" class="com.zqq.s2sh.service.impl.UserServiceImpl">  
 <**property** name="userDao" ref="userDao"/>  
 </**bean**>  
 <!--配置事务-->  
 <**bean** id="hibernateTransactionManager" class="org.springframework.orm.hibernate5.HibernateTransactionManager">  
 <**property** name="sessionFactory" ref="sessionFactory"/>  
 </**bean**>  
 <**tx:advice** id="txAdvice" transaction-manager="hibernateTransactionManager">  
 <**tx:attributes**>  
 <**tx:method** name="\*" propagation="REQUIRED" read-only="false"/>  
 <**tx:method** name="find\*" propagation="SUPPORTS" read-only="true"/>  
 </**tx:attributes**>  
 </**tx:advice**>  
<**aop:config**>  
 <**aop:pointcut** id="servicePoint" expression="execution(\*com.zqq.s2sh.service.impl.\*.\*(..))"/>  
 <**aop:advisor** advice-ref="txAdvice" pointcut-ref="servicePoint"/>  
</**aop:config**>  
</**beans**>

**5、测试事务配置情况**

@RunWith(**SpringJUnit4ClassRunner**.**class**)  
@ContextConfiguration("classpath:applicationContext.service.xml")  
**public class ServiceTest** {  
 @Autowired  
 **private** IUserService userService;  
 @Test  
 **public void** test(){  
 **User** user = **new** User();  
 user.setUserName("zqq");  
 user.setPassword("123");  
 userService.saveUser(user);  
 }  
}

**四、搭建s2sh-web模块**

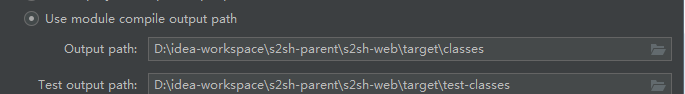
**1、新建s2sh-web模块**

使用webapp模板创建s2sh-web模块。手动修改如下：

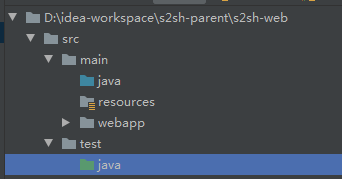
（1）修改pom文件依赖父模块

<**parent**>  
 <**artifactId**>s2sh-parent</**artifactId**>  
 <**groupId**>com.zqq.s2sh</**groupId**>  
 <**version**>1.0-SNAPSHOT</**version**>  
</**parent**>  
<**modelVersion**>4.0.0</**modelVersion**>  
<**artifactId**>s2sh-dao</**artifactId**>

（2）修改compile output path路径



（3）新增java目录



**2、添加pom依赖**

需添加struts2框架jar包依赖，及struts2-spring-plugin和spring整合必须的包

<**dependencies**>  
 <**dependency**>  
 <**groupId**>com.zqq.s2sh</**groupId**>  
 <**artifactId**>s2sh-service</**artifactId**>  
 <**version**>${project.version}</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.apache.struts</**groupId**>  
 <**artifactId**>struts2-core</**artifactId**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.apache.struts</**groupId**>  
 <**artifactId**>struts2-spring-plugin</**artifactId**>  
 </**dependency**>  
 <!--配置struts2注解需要的jar包-->  
 <**dependency**>  
 <**groupId**>org.apache.struts</**groupId**>  
 <**artifactId**>struts2-convention-plugin</**artifactId**>  
 </**dependency**>  
</**dependencies**>

<**dependency**>  
 <**groupId**>jstl</**groupId**>  
 <**artifactId**>jstl</**artifactId**>  
 <**version**>1.2</**version**>  
</**dependency**>

**3、准备页面和action类**

UserAciont.java和register.jsp

**4、配置struts.xml**

<**struts**>  
 <**constant** name="struts.devMode" value="true"/>

<!—将action的创建交给spring -->  
 <**constant** name="struts.objectFactory" value="spring"/>  
 <**package** name="zqqs2sh" namespace="/user" extends="json-default">  
 <**action** name="register" class="userAction" method="register">  
 <**result** name="success" type="redirect">/s2sh.jsp</**result**>  
 </**action**>  
 <**action** name="findAllUser" class="userAction" method="findAllUser">  
 <**result** name="users" type="json">  
 <**param** name="root">users</**param**>  
 </**result**>  
 </**action**>  
 </**package**>  
</**struts**>

**5、配置applicationContext.web.xml**

<**import** resource="classpath:applicationContext.service.xml"/>  
 <!--多例action-->  
 <**bean** id="userAction" class="com.zqq.s2sh.web.action.UserAction" scope="prototype">  
 <**property** name="userService" ref="userService"/>  
 </**bean**>

**6、配置web.xml**

修改web.xml版本

*<?*xml version="1.0" encoding="UTF-8"*?>*<**web-app** 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\_3\_0.xsd"  
 version="3.0">

(1)struts2核心过滤器

<!--struts2核心过滤器-->  
<**filter**>  
 <**filter-name**>struts2</**filter-name**> <**filter-class**>org.apache.struts2.dispatcher.filter.StrutsPrepareAndExecuteFilter</**filter-class**>  
</**filter**>  
<**filter-mapping**>  
 <**filter-name**>struts2</**filter-name**>  
 <**url-pattern**>\*</**url-pattern**>

(2)servlectContextListener的实现类ContextLoaderListener监听器

<!--配置spring配置文件启动时加载，使用ServletContextLinsener-->  
<**listener**>  
<**listener-class**>org.springframework.web.context.ContextLoaderListener</**listener-class**>  
</**listener**>

(3)配置spring配置文件位置

<!--Spring配置文件位置-->  
<**context-param**>  
 <**param-name**>contextConfigLocation</**param-name**>  
 <**param-value**>classpath:applicationContext.web.xml</**param-value**>  
</**context-param**>

**7、修改struts.xml将action交给spring管理**

正常Action由objectFactory创建，由以下配置修改为spring管理

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

**8、测试及修改**

（1）由于struts-spring-plugin及spring-web包引用了其他版本的spring，导致依赖冲突，需显示依赖spring-beans和spring-core。在s2sh-parent模块依赖

<**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-core</**artifactId**>  
</**dependency**>  
<**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-beans</**artifactId**>  
</**dependency**>

（2）新增struts-json-plugin的依赖，实现struts自动转换为json返回。s2sh-web模块依赖：

<**dependency**>  
 <**groupId**>org.apache.struts</**groupId**>  
 <**artifactId**>struts2-json-plugin</**artifactId**>  
</**dependency**>

（3）在struts2使用转发时，由于命名空间的名称与要访问的jsp目录不一致，导致css和jsp文件加载失败

**解决方式：1、命名空间与jsp目录保持一致**

**2、使用重定向的方式访问jsp**

**3、不使用命名空间。**

**五、变更为注解+XML配置S2SH**

**1、s2sh-dao变更**

（1）hibernate改注解，并将主配置文件写入spring中。

修改实体类：

@Entity  
@Table(name = "user")  
**public class User implements** Serializable {  
 @Id  
 @Column(name = "userid")  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 **private Long** userId;  
 @Column(name = "username")  
 **private String** userName;  
 @Column(name = "password")  
 **private String** password;

删除User.hbm.xml、hibernate.cfg.xm。将hibernate配置写入applicationContext.dao.xml.

<**bean** id="sessionFactory" class="org.springframework.orm.hibernate5.LocalSessionFactoryBean">  
 <!--数据源-->  
 <**property** name="dataSource" ref="dataSource"/>  
 <!--hibernate可选配置-->  
 <**property** name="hibernateProperties">  
 <**props**>  
 <**prop** key="dialect">org.hibernate.dialect.MySQL5Dialect</**prop**>  
 <**prop** key="hibernate.show\_sql">true</**prop**>  
 <**prop** key="hibernate.format\_sql">false</**prop**>  
 <**prop** key="hibernate.current\_session\_context\_class">org.springframework.orm.hibernate5.SpringSessionContext</**prop**>  
 <**prop** key="hibernate.hbm2ddl.auto">update</**prop**>  
 </**props**>  
 </**property**>  
 <!--hibernate映射 mappingLocations指定映射文件位置  
 packageToScan指定实体类所在的包-->  
 <**property** name="packagesToScan">  
 <**array**>  
 <**value**>com.zqq.s2sh.domain</**value**>  
 </**array**>  
 </**property**>

</**bean**>

<**bean** id="hibernateTemplate" class="org.springframework.orm.hibernate5.HibernateTemplate">  
 <**property** name="sessionFactory" ref="sessionFactory"/>  
</**bean**>

<!--数据源-->  
<**bean** id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">  
 <**property** name="driverClass" value="com.mysql.cj.jdbc.Driver"/>  
 <**property** name="jdbcUrl" value="jdbc:mysql://127.0.0.1:3306/zqqdb?serverTimezone=UTC"/>  
 <**property** name="user" value="zqq"/>  
 <**property** name="password" value="123"/>  
</**bean**>

<**context:component-scan** base-package="com.zqq.s2sh.dao.impl"/>

（2）配置dao实现类使用spring注解

@Repository("userDao")  
**public class UserDaoImpl implements** IUserDao {  
 @Resource(name="hibernateTemplate")  
 **private HibernateTemplate** hibernateTemplate;

**2、s2sh-service变更**

修改applicationContext.service.xml配置文件：

<!--配置IOC需要扫描的包-->  
<**context:component-scan** base-package="com.zqq.s2sh.service.impl"/>

<!--配置事务管理器-->  
<**bean** id="hibernateTransactionManager" class="org.springframework.orm.hibernate5.HibernateTransactionManager">  
<**property** name="sessionFactory" ref="sessionFactory"/>  
</**bean**>  
<!--配置事务支持注解-->  
<**tx:annotation-driven** transaction-manager="hibernateTransactionManager"/>

配置service实现类使用IOC和tx注解：

@Service("userService")  
@Transactional(readOnly = **false**,propagation = Propagation.REQUIRED)  
**public class UserServiceImpl implements** IUserService {  
 @Resource(name = "userDao")  
 **private** IUserDao userDao ;  
 @Override  
 **public void** saveUser(**User** user) {  
 userDao.saveUser(user);  
 }  
 @Override  
 @Transactional(readOnly = **true**,propagation = Propagation.SUPPORTS)  
 **public** List<**User**> findAllUser() {  
 **return** userDao.findAllUser();  
 }  
}

**3、s2sh-web变更**

删除struts.xml中action的配置，在action实现类中配置注解：

@ParentPackage("json-default")  
@Namespace("user")  
@Controller("userAction")@Scope("prototype")此行写了没有效果，依然是struts创建action  
**public class UserAction extends ActionSupport implements** ModelDriven<**User**> {  
 **private User** user = **new** User();  
 @Resource(name = "userService")  
 **private** IUserService userService;  
 **private** List<**User**> users;  
 @Action(value = "register",results = {  
 @Result(name = "success",type = "redirect",location = "/s2sh.jsp")  
 })  
 **public String** register() {  
 userService.saveUser(user);  
 **return** SUCCESS;  
 }  
 @Action(value = "findAllUser",results = {  
 @Result(name = "users",type = "json",params = {"root","users"})})  
 **public String** findAllUser() {  
 users = userService.findAllUser();  
 **return** "users";  
 }  
 @Override  
 **public User** getModel() {  
 **return** user;  
 }  
 **public** List<**User**> getUsers() {  
 **return** users;  
 }  
 **public void** setUsers(List<**User**> users) {  
 **this**.users = users;  
 }  
}

修改struts.xml

<**struts**>  
 <**constant** name="struts.devMode" value="true"/>  
 <!--action交给spring来创建,下面第二个配置始终使用自动注入,解决aop拦截时没有注入参数的错误-->  
 <**constant** name="struts.objectFactory" value="spring"/>  
 <**constant** name="struts.objectFactory.spring.autoWire.alwaysRespect" value="true"/>  
</**struts**>

修改applicationContext.web.xml:

<**import** resource="classpath:applicationContext.service.xml"/>  
 <**context:component-scan** base-package="com.zqq.s2sh.web.action"/>