

# 1. SSH框架整合思想

1 三大框架应用在javaee三层结构



2 struts2框架和spring整合

(1) struts2的action在spring配置

- \* **struts2框架和spring整合**
- \*\* **把struts2的action创建 交给spring配置**
- \*\*\* **配置action多实例的**

3 spring框架和hibernate框架整合

(1) hibernate的sessionFactory交给spring配置

(2) 把hibernate数据库配置交给spring配置

- spring框架 和 hibernate整合**
- \*\* **( 1 ) 把hibernate的sessionFactory交给spring管理**
- \*\* **( 2 ) 把hibernate里面的数据库信息配置 , 交给spring进行管理**

## 2. 整合struts2和spring框架

1 把struts2的action交给spring管理

2 实现过程

第一步 导入struts2的jar包

(1) 导入用于整合的jar包



第二步 创建action

第三步 创建struts2核心配置文件，配置action

(1) 位置在src下面，名称是struts.xml

第四步 配置struts2过滤器

第五步 导入spring的jar包

第六步 创建spring配置文件

(1) 引入约束

(2) 配置spring监听器

(3) 指定spring配置文件位置

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

```
<context-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>classpath:bean1.xml</param-value>
</context-param>
```

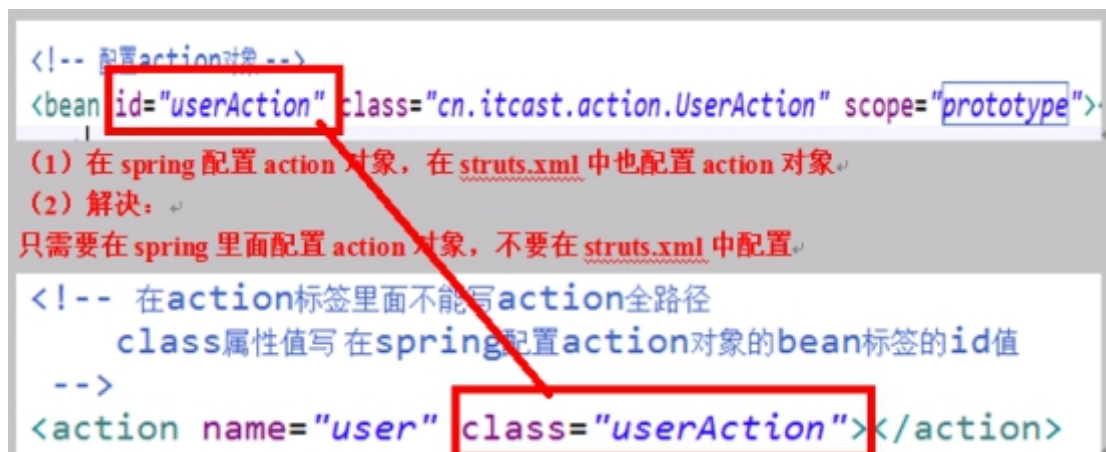
第七步 把action交给spring进行配置 (\*\*\*)

```
<!-- 配置action对象 -->
<bean id="userAction" class="cn.itcast.action.UserAction" scope="prototype">
```

(1) 在spring配置action对象，在struts.xml中也配置action对象

(2) 解决：只需要在spring里面配置action对象，不要在struts.xml中配置

```
<!-- 在action标签里面不能写action全路径
      class属性值写 在spring配置action对象的bean标签的id值
-->
<action name="user" class="userAction"></action>
```



### 3. Spring框架整合hibernate框架

1 把hibernate核心配置文件中配置数据库信息，把数据库信息在spring进行配置

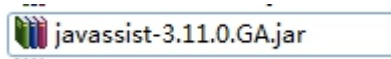
2 把hibernate里面的sessionFactory创建交给spring管理

#### 3-1. 具体实现

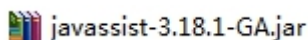
第一步 导入hibernate的jar包

(1) 导入struts2和hibernate的jar包时候有jar冲突问题

在struts2里面有jar包



在hibernate里面有jar包



删除低版本的jar包

(2) 导入spring整合持久化层框架需要导入jar包



## 第二步 搭建hibernate环境搭建

- 1 创建实体类
- 2 配置实体类映射关系
- 3 创建核心配置文件

## 第三步 把hibernate核心配置文件数据库配置，在spring进行配置

(1) 把hibernate核心文件中数据库配置去掉了，在spring配置

```
<!-- 配置连接池 -->
<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">
    <property name="driverClass" value="com.mysql.jdbc.Driver"></property>
    <property name="jdbcUrl" value="jdbc:mysql:///spring_day04"></property>
    <property name="user" value="root"></property>
    <property name="password" value="root"></property>
</bean>
```

## 第四步 把hibernate的sessionFactory交给spring配置

- (1) 服务器启动时候，加载spring配置文件，把配置文件中对象创建
- (2) 把sessionFactory对象创建在spring配置
- (3) 因为创建sessionFactory代码不是new出来的，而是多行代码实现的

```
//加载核心配置文件
cfg = new Configuration();
cfg.configure();
//创建sessionFactory
sessionFactory = cfg.buildSessionFactory();
```

(4) spring里面针对上面情况，封装类，配置类对象可以创建sessionFactory

```
<!-- 配置sessionFactory创建 -->
<bean id="sessionFactory" class="org.springframework.orm.hibernate5.LocalSessionFactoryBean">
    <!-- 指定数据库信息 -->
    <property name="dataSource" ref="dataSource"></property>
    <!-- 指定使用hibernate核心配置文件位置 -->
    <property name="configLocations" value="classpath:hibernate.cfg.xml"></property>
</bean>
```

## 第五步 在dao里面使用hibernateTemplate

Hibernate5.0	org.springframework.orm.hibernate5.HibernateTemplate
--------------	--

### (1) 在dao得到hibernateTemplate的对象

```
private HibernateTemplate hibernateTemplate;  
public void setHibernateTemplate(HibernateTemplate hibernateTemplate) {  
    this.hibernateTemplate = hibernateTemplate;  
}
```

```
<bean id="userDao" class="cn.itcast.dao.UserDao">  
    <!-- 注入HibernateTemplate -->  
    <property name="hibernateTemplate" ref="hibernateTemplate"></property>  
</bean>  
  
<!--创建hibernate模板对象-->  
<bean id="hibernateTemplate" class="org.springframework.orm.hibernate5.HibernateTemplate">  
    <property name="sessionFactory" ref="sessionFactory"></property>  
</bean>
```

### (2) 调用hibernate模板里面save方法添加

```
User user = new User();  
user.setUsername("lucy");  
user.setAddress("日本");
```

```
hibernateTemplate.save(user);
```

没有配置事务，做操作时候，出现异常

**COMMIT/AUTO or remove 'readOnly' marker from transaction**

## 第六步 配置事务

```
<!-- 1 配置事务管理器 -->  
<bean id="transactionManager" class="org.springframework.orm.hibernate5.HibernateTransactionManager"  
    <property name="sessionFactory" ref="sessionFactory"></property>  
</bean>
```

```
<!-- 开启事务注解 -->  
<tx:annotation-driven transaction-manager="transactionManager"/>
```

```
@Transactional
public class UserService {
```

## 4. HibernateTemplate介绍

1 HibernateTemplate对hibernate框架进行封装,  
直接调用HibernateTemplate里面的方法实现功能

2 HibernateTemplate常用的方法

- Serializable save(Object entity) : 添加操作
- void update(Object entity) : 修改操作
- void delete(Object entity) : 删除操作
- <T> T get(Class<T> entityClass, Serializable id) : 根据id查询
- <T> T load(Class<T> entityClass, Serializable id) : 根据id查询

```
// get方法: 根据id查询
User user = hibernateTemplate.get(User.class, 2);
System.out.println(user.getUsername()+"::"+user.getAddress());
```

- List find(String queryString, Object... values) : 查询操作的方法

```
find(String queryString, Object... values) : List<?> -
```

(1) 第一个参数是 hql语句

(2) 语句参数值

```
//2 find方法查询所有记录
List<User> list = (List<User>) hibernateTemplate.find("from User");
for (User user : list) {
    System.out.println(user.getUsername()+"::"+user.getAddress());
}
```

```
findByCriteria(DetachedCriteria criteria, int firstResult, int maxResults) : List<?>
```

## 5. SSH框架整合过程

第一步 导入jar包

```
Spring_day04 ▶ 资料 ▶ 01-相关jar包 ▶ ssh整合jar包
```



## 第二步 搭建struts2环境

- (1) 创建action, 创建struts.xml配置文件, 配置action
- (2) 配置struts2的过滤器

## 第三步 搭建hibernate环境

- (1) 创建实体类
- (2) 配置实体类和数据库表映射关系
- (3) 创建hibernate核心配置文件
  - 引入映射配置文件

## 第四步 搭建spring环境

- (1) 创建spring核心配置文件
- (2) 让spring配置文件在服务器启动时候加载
  - 配置监听器

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

- 指定spring配置文件位置

```
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath:bean1.xml</param-value>
</context-param>
```

## 第五步 struts2和spring整合

- (1) 把action在spring配置 (action多实例的)

```
<!-- 配置action的对象 -->
<bean id="userAction" class="cn.itcast.action.UserAction" scope="prototype">
```

- (2) 在struts.xml中action标签class属性里面写 bean的id值

```
<!-- class属性里面不写action全路径了, 因为写, action对象创建两次
    写spring配置的action的bean的id值
-->
<action name="userAction" class="userAction"></action>
```

## 第六步 spring和hibernate整合

- (1) 把hibernate核心配置文件中数据库配置, 在spring里面配置

```

<!-- 配置c3p0连接池 -->
<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">
    <!-- 注入属性值 -->
    <property name="driverClass" value="com.mysql.jdbc.Driver"></property>
    <property name="jdbcUrl" value="jdbc:mysql:///spring_day04"></property>
    <property name="user" value="root"></property>
    <property name="password" value="root"></property>
</bean>

```

## (2) 把hibernate的sessionFactory在spring配置

```

<!-- sessionFactory创建交给spring管理 -->
<bean id="sessionFactory" class="org.springframework.orm.hibernate5.LocalSessionFactoryBe
    <!-- 因为在hibernate核心配置文件中，没有数据库配置，数据库配置在spring里面配置，注入dataSource -->
    <property name="dataSource" ref="dataSource"></property>

    <!-- 指定使用hibernate核心配置文件 -->
    <property name="configLocations" value="classpath:hibernate.cfg.xml"></property>
</bean>

```

## 第七步 在dao里面使用hibernateTemplate

### (1) 在dao注入hibernateTemplate对象

### (2) 在hibernateTemplate对象中注入sessionFactory

```

<!-- 创建实现类对象 -->
<bean id="userDaoImpl" class="cn.itcast.dao.UserDaoImpl">
    <property name="hibernateTemplate" ref="hibernateTemplate"></property>
</bean>

<!-- 创建hibernateTemplate对象 -->
<bean id="hibernateTemplate" class="org.springframework.orm.hibernate5.HibernateTempla
    <!-- 注入sessionFactory -->
    <property name="sessionFactory" ref="sessionFactory"></property>
</bean>

```

## 第八步 配置事务

```

<!-- 第一步 配置事务管理器 -->
<bean id="transactionManager" class="org.springframework.orm.hibernate5.HibernateTransactionl
    <!--注入sessionFactory-->
    <property name="sessionFactory" ref="sessionFactory"></property>
</bean>

<!-- 第二步 开启事务注解 -->
<tx:annotation-driven transaction-manager="transactionManager"/>

```

@Transactional

```
public class UserService {
```

出现异常时候，

第一行 异常信息

下面 caused by :



## 6. 整合其他方式

1 spring整合hibernate时候, 可以不写hibernate核心配置文件

(1) 把hibernate核心配置文件中, 基本信息配置和映射引入都放到spring配置

```
<bean id="sessionFactory" class="org.springframework.orm.hibernate5.LocalSessionFactoryB
  <property name="dataSource" ref="dataSource"></property>
  <!-- <property name="configLocations" value="classpath:hibernate.cfg.xml"></property>
  <!-- 如果不写hibernate核心配置文件, -->
  <!-- 1把hibernate基本信息配置 -->
  <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>
  <!-- 2把映射文件引入配置 -->
  <property name="mappingResources">
    <list>
      <value>cn/itcast/entity/User.hbm.xml</value>
    </list>
  </property>
</bean>
```

## 7. Spring分模块开发

1 在spring里面配置多个内容, 造成配置混乱, 不利用维护























2 把spring核心配置文件中, 一部分配置放到单独的配置文件中, 在spring核心配置文件中引入单独配置文件

```
<!-- 引入其他spring配置文件 -->
<import resource="classpath:user.xml"/>
```

## 8. SSH整合所需引入的jar包总结

8-1.文件路径 F:\Develop\ssh整合jar包

8-2. 所有jar包截图

Name	Date modified	Type	Size
 antlr-2.7.7.jar	11/22/2016 3:54 AM	Executable Jar File	435 KB
 aopalliance-1.0.jar	11/22/2016 3:54 AM	Executable Jar File	5 KB
 asm-3.3.jar	11/22/2016 3:54 AM	Executable Jar File	43 KB
 asm-commons-3.3.jar	11/22/2016 3:54 AM	Executable Jar File	38 KB
 asm-tree-3.3.jar	11/22/2016 3:54 AM	Executable Jar File	21 KB
 aspectjweaver-1.8.7.jar	11/22/2016 3:54 AM	Executable Jar File	1,822 KB
 c3p0-0.9.2.1.jar	11/22/2016 3:54 AM	Executable Jar File	414 KB
 commons-fileupload-1.3.1.jar	11/22/2016 3:54 AM	Executable Jar File	68 KB
 commons-io-2.2.jar	11/22/2016 3:54 AM	Executable Jar File	170 KB
 commons-lang3-3.2.jar	11/22/2016 3:54 AM	Executable Jar File	376 KB
 commons-logging-1.2.jar	11/22/2016 3:54 AM	Executable Jar File	61 KB
 dom4j-1.6.1.jar	11/22/2016 3:54 AM	Executable Jar File	307 KB
 freemarker-2.3.22.jar	11/22/2016 3:54 AM	Executable Jar File	1,271 KB
 geronimo-jta_1.1_spec-1.1.1.jar	11/22/2016 3:54 AM	Executable Jar File	16 KB
 hibernate-commons-annotations-5.0.1.Final.jar	11/22/2016 3:54 AM	Executable Jar File	74 KB
 hibernate-core-5.0.7.Final.jar	11/22/2016 3:54 AM	Executable Jar File	5,453 KB
 hibernate-entitymanager-5.0.7.Final.jar	11/22/2016 3:54 AM	Executable Jar File	585 KB
 hibernate-jpa-2.1-api-1.0.0.Final.jar	11/22/2016 3:54 AM	Executable Jar File	111 KB
 jandex-2.0.0.Final.jar	11/22/2016 3:54 AM	Executable Jar File	184 KB
 javassist-3.18.1-GA.jar	11/22/2016 3:54 AM	Executable Jar File	698 KB
 jboss-logging-3.3.0.Final.jar	11/22/2016 3:54 AM	Executable Jar File	66 KB
 log4j-1.2.16.jar	1/23/2018 12:52 PM	Executable Jar File	471 KB

## 8-3. applicationContext.xml 所有约束汇总

```

<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:aop="http://www.springframework.org/schema/aop"
    xmlns:tx="http://www.springframework.org/schema/tx"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context.xsd
        http://www.springframework.org/schema/aop
        http://www.springframework.org/schema/aop/spring-aop.xsd
        http://www.springframework.org/schema/tx
        http://www.springframework.org/schema/tx/spring-tx.xsd">
</beans>

```

