ssh整合注意事项有hibernate配置文件版

ssh三大框架整合

Struts2，spring，hibernate三大框架整合。

三大框架知识点：

hibernate框架：

1.hibernate核心配置文件：

如果单独使用hibernate框架，核心配置文件名称hibernate.cfg.xml和位置src是固定的。

如果hibernate和spring整合一起使用的时候hibernate核心配置文件的名称和位置是没有固定要求的。

2.hibernate映射配置文件

1)实体类和数据库表映射关系一致，使用orm思想。

3.hibernate操作的步骤

1）spring框架的hibernate框架进行封装，使用hibernateTemplate

Structs2框架

1.action部分

1.Action创建三种方式

继承类ActionSupport，

2.配置action访问路径

创建structs.xml配置文件，文件固定在src目录下面

3.配置访问action的多个方法

使用通配符配置方式

4.在action获取表单提交数据

获取request对象

属性封装

模型驱动

5.action中操作域对象

使用ServletActionContext获取域对象

6.配置structs2的过滤器

2.值栈

1向值栈中存放数据

set方法

push方法

定义变量，生成get方法

2在jsp中使用structs2标签+ognl表达式

<s:property>

<s:iterator>

3.拦截器

1.aop和责任链模式

2自定义拦截器

继承MethodFilterInterceptor

重写类里面的方法

配置拦截器和action关联

spring框架

1.spring核心配置文件

1）名称和位置没有固定要求

2）在Spring核心配置文件中添加schema约束

2.创建对象

1) xml配置方式，<bean id="" class="" scope=""/>

2)注解方式，四个注解

@Component ：

@Controller :web层

@Service ：业务层

@Respository :持久层

3.注入属性

xml配置方式

注解方式

创建两个类，并在userService类中添加UserDao类的属性，在属性字段上添加@Autowired注解

@Autowired自动注入，根据类名去找属性

@Resource(name="属性字段的value值")

（@Autowired不用配置属性字段的值，@Resource可以指定对象）

```
@Component(value="dao")
public class UserDao {
 public void add(){
  System.out.println("add Dao!");
 }
}

public class UserService {
 //得到对象
 //使用注解不需要set方法
 //在属性上添加注解@Autowired
 @Autowired
 private UserDao userDao;

 public void add(){
  System.out.println("add Service!");
  userDao.add();
 }
}
```

4.使用servletContext对象和监听器实现

在服务器启动的时候加载spring配置文件，创建对象。

配置spring监听器

指定spring配置文件位置

指定spring配置文件位置

5.jdbcTemplate

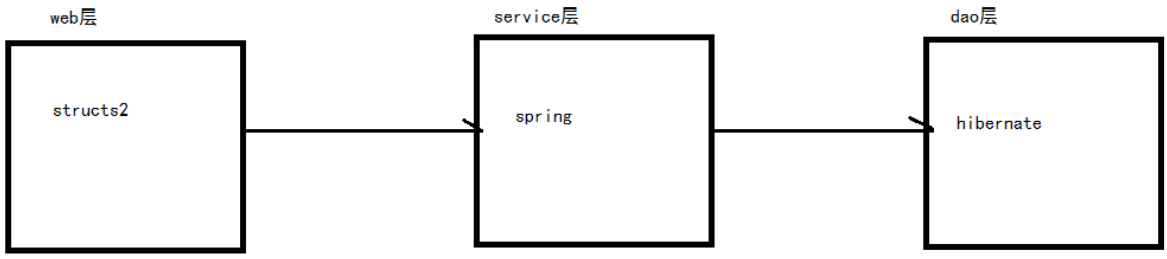
6.spring事务

xml配置方式

注解配置方式

ssh整合思想

ssh整合思想



*structs2框架和spring整合

*把structs2的action对象交个spring进行管理

<bean id="" class="" scope="prototype" />

*spring框架和hibernate进行整合

把hibernate核心配置文件中的数据库配置信息，直接写在spring配置文件中

第一次访问时，把sessionFactory对象的创建交给spring管理

整合structs2和spring框架

1.把struts2的action交给spring管理

2.实现过程

第一步： 导入struts和spring的jar包

```
<properties>
 <org.springframework-version>4.3.6.RELEASE</org.springframework-version>
</properties>
<dependencies>
 <!-- spring相关jar包 -->
 <dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-beans</artifactId>
  <version>${org.springframework-version}</version>
 </dependency>
 <dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context</artifactId>
  <version>${org.springframework-version}</version>
 </dependency>
 <dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-expression</artifactId>
  <version>${org.springframework-version}</version>
 </dependency>
 <dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-core</artifactId>
  <version>${org.springframework-version}</version>
 </dependency>
 <dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-aop</artifactId>
  <version>${org.springframework-version}</version>
 </dependency>
 <dependency>
  <groupId>org.springframework</groupId>
```

```
 <artifactId>spring-aspects</artifactId>
 <version>${org.springframework-version}</version>
</dependency>
<dependency>
 <groupId>org.springframework</groupId>
 <artifactId>spring-web</artifactId>
 <version>${org.springframework-version}</version>
</dependency>
<dependency>
 <groupId>org.springframework</groupId>
 <artifactId>spring-jdbc</artifactId>
 <version>${org.springframework-version}</version>
</dependency>
<dependency>
 <groupId>org.springframework</groupId>
 <artifactId>spring-tx</artifactId>
 <version>${org.springframework-version}</version>
</dependency>
<dependency>
 <groupId>org.springframework</groupId>
 <artifactId>spring-web</artifactId>
 <version>${org.springframework-version}</version>
</dependency>
<!-- spring相关jar包配置结束 -->

<!-- spring aop编程依赖jar包 -->
<dependency>
 <groupId>org.aspectj</groupId>
 <artifactId>aspectjweaver</artifactId>
 <version>1.8.10</version>
</dependency>
<dependency>
 <groupId>aopalliance</groupId>
 <artifactId>aopalliance</artifactId>
 <version>1.0</version>
</dependency>
<!-- spring aop编程依赖jar包 配置结束 -->

<!-- struts2-spring整合jar包 -->
<dependency>
 <groupId>org.apache.struts</groupId>
 <artifactId>struts2-spring-plugin</artifactId>
 <version>2.5.10</version>
</dependency>
<!-- struts2-spring整合jar包 结束 -->

<!-- struts2相关jar包 -->
<dependency>
 <groupId>asm</groupId>
 <artifactId>asm</artifactId>
 <version>3.3.1</version>
</dependency>
```

```
<dependency>
 <groupId>asm</groupId>
 <artifactId>asm-commons</artifactId>
 <version>3.3.1</version>
</dependency>
<dependency>
 <groupId>asm</groupId>
 <artifactId>asm-tree</artifactId>
 <version>3.3.1</version>
</dependency>
<dependency>
 <groupId>commons-fileupload</groupId>
 <artifactId>commons-fileupload</artifactId>
 <version>1.3.2</version>
</dependency>
<dependency>
 <groupId>commons-io</groupId>
 <artifactId>commons-io</artifactId>
 <version>2.5</version>
</dependency>
<dependency>
 <groupId>commons-lang</groupId>
 <artifactId>commons-lang</artifactId>
 <version>2.6</version>
</dependency>
<dependency>
 <groupId>org.freemarker</groupId>
 <artifactId>freemarker</artifactId>
 <version>2.3.25-incubating</version>
</dependency>
<dependency>
 <groupId>org.javassist</groupId>
 <artifactId>javassist</artifactId>
 <version>3.22.0-CR1</version>
</dependency>
<dependency>
 <groupId>javax.servlet</groupId>
 <artifactId>jstl</artifactId>
 <version>1.2</version>
</dependency>
<dependency>
 <groupId>ognl</groupId>
 <artifactId>ognl</artifactId>
 <version>3.2</version>
</dependency>

<dependency>
 <groupId>taglibs</groupId>
 <artifactId>standard</artifactId>
 <version>1.1.2</version>
</dependency>
<dependency>
```

```
 <groupId>org.apache.struts</groupId>
 <artifactId>struts2-core</artifactId>
 <version>2.5.10</version>
</dependency>
<dependency>
 <groupId>org.apache.struts.xwork</groupId>
 <artifactId>xwork-core</artifactId>
 <version>2.3.31</version>
</dependency>
<!-- struts2相关jar包结束 -->
<!-- 日志文件jar包 -->
<dependency>
 <groupId>commons-logging</groupId>
 <artifactId>commons-logging</artifactId>
 <version>1.2</version>
</dependency>
<dependency>
 <groupId>log4j</groupId>
 <artifactId>log4j</artifactId>
 <version>1.2.17</version>
</dependency>
<!-- 日志文件jar包配置结束 -->

<!-- junit测试jar包结束 -->
<dependency>
 <groupId>junit</groupId>
 <artifactId>junit</artifactId>
 <version>4.12</version>
</dependency>
<!-- junit测试jar包 -->

<!-- mysql数据库连接驱动jar包 -->
<dependency>
 <groupId>mysql</groupId>
 <artifactId>mysql-connector-java</artifactId>
 <version>5.1.40</version>
</dependency>
<!-- mysql数据库连接驱动jar包结束 -->

<!-- c3p0连接处jar包及依赖 -->
<dependency>
 <groupId>c3p0</groupId>
 <artifactId>c3p0</artifactId>
 <version>0.9.1.2</version>
</dependency>
<dependency>
 <groupId>com.mchange</groupId>
 <artifactId>mchange-commons-java</artifactId>
 <version>0.2.12</version>
</dependency>
<!-- c3p0连接处jar包及依赖结束 -->
</dependencies>
```

第二步：创建action类

```
public class UserAction extends ActionSupport{
 @Override
 public String execute() throws Exception {
  System.out.println("action.....running!");
  return NONE;
 }
}
```

第三步：创建struts2核心配置文件，配置action

1）位置src下面，名称struts.xml

struts配置文件：

```
<!DOCTYPE struts PUBLIC
   "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
   "http://struts.apache.org/dtds/struts-2.0.dtd">
<struts>
 <package name="demo1" extends="struts-default" namespace="/">
  <!-- classpath属性里面不写action全路径了，因为写，action对象会创建两次 写spring配置里action bean的id值 -->
  <action name="userAction" class="userAction"></action>
 </package>
</struts> "
```

2）spring配置文件：

```
<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:tx="http://www.springframework.org/schema/tx"
  xmlns:aop="http://www.springframework.org/schema/aop" 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/tx http://www.springframework.org/schema/tx/spring-tx.xsd
   http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop.xsd">


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

  <!-- action配置 -->
  <bean id="userAction" class="com.wmr.ssh.action.UserAction" scope="prototype"></bean>
 </beans>
```

spring整合hibernate

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中进行配置

2.把hibernate；里面的sessionFactory创建交给spring管理

web.xml文件：

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
id="WebApp_ID" version="2.5">
  <display-name>sshtest1</display-name>
  <!-- 配置struts2核心拦截器 -->
 <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>
 </filter-mapping>

 <!-- 配置spring监听器项目启动时加载spring配置文件 -->
 <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>
</web-app>
```

struts.xml文件：

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
    "http://struts.apache.org/dtds/struts-2.0.dtd">
<struts>
 <package name="demo1" extends="struts-default" namespace="/">
  <!-- classpath属性里面不写action全路径了，因为写，action对象会创建两次 写spring配置里action bean的id值 -->
  <action name="userAction" class="userAction"></action>
 </package>
</struts>
```

spring配置文件：springContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<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:tx="http://www.springframework.org/schema/tx"
  xmlns:aop="http://www.springframework.org/schema/aop" 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/tx http://www.springframework.org/schema/tx/spring-tx.xsd
    http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop.xsd">
```

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

<!-- 创建sessionFactory -->
<bean id="sessionFactory" class="org.springframework.orm.hibernate5.LocalSessionFactoryBean" >
 <!-- 注入数据库的配置 -->
 <property name="dataSource" ref="dataSource"></property>
 <!-- 指定实用的hibernate核心配置文件 -->
 <property name="configLocations" value="classpath:hibernate.cfg.xml"></property>

</bean>

<!-- action配置 -->
<!-- action inject service  -->
<bean id="userAction" class="com.wmr.ssh.action.UserAction"  scope="prototype">
 <property name="userService" ref="userService"/>
</bean>

<!-- service inject dao -->
<bean id="userService" class="com.wmr.ssh.service.UserService">
 <property name="userDao" ref="userDao"/>
</bean>

<!-- dao inject hibernateTemplate -->
<bean id="userDao" class="com.wmr.ssh.dao.UserDaoImpl">
 <property name="hibernateTemplate" ref="hibernateTemplate"/>
</bean>

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

<!-- 配置事务 -->
<!-- 第一步：配置事务管理器 -->
<bean id="hibernateTransactionManager" class="org.springframework.orm.hibernate5.HibernateTransactionManager">
 <!-- 注入sessionFactory -->
 <property name="sessionFactory" ref="sessionFactory" />
</bean>
<!-- 第二步开启事务注解 -->
<tx:annotation-driven transaction-manager="hibernateTransactionManager"/>

</beans>
```

hibernate配置文件：hibernate.cfg.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
 <session-factory>
  <!-- 输出底层sql语句 -->
  <property name="hibernate.show.show_sql">true</property>
  <!-- 输出底层sql语句的格式 -->
  <property name="hibernate.format_sql">true</property>
  <!-- hibernate创建表，需要配置之后配置 update:如果已经有表，更新，如果没有创建 -->
  <property name="hibernate.hbm2ddl.auto">update</property>

  <!-- 配置数据库方言 在mysql里面实现分页，关键字limit，只能使用mysql里面 在Oracle中实现分页，rownum 让hibernate框架识别不同的数据库自己的特有语言 -->
  <property name="dialect">org.hibernate.dialect.MySQLDialect</property>
  <!-- 配置hibernate核心配置文件中的配置 -->
  <property name="hibernate_current_session_context_class">thread</property>
  <mapping class="com.wmr.ssh.entity.User" />

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