表现层：SpringMVC

包：SpringMVC的jar

配置文件：spring-mvc.xml

业务层：Spring

包：spring-ioc

Spring-aop

Spring-tx

Spring-test

配置文件：applicationContext.xml

添加关于mybatis的配置

持久层：mybatis

包：mybatis自身核心包

Mysql数据库驱动包

C3p0连接池

Mybatis-spring的包

配置文件：sqlMapConfig.xml

Customer.xml

1. 单独使用mybatis
2. Mybatis整合spring - 有Mapper实现类
3. Mybatis整合spring – 没有Mapper实现类
4. Mybatis整合spring – mapper接口扫描
5. Mybatis 整合spring – 整合jdbc事务
6. 整合springmvc
7. 单独使用mybatis
8. 导入必须的包：
   1. Mybatis自身核心jar：mybatis-3.4.4.jar
   2. Log4j日志jar:log4j-1.2.17.jar
   3. Oracle驱动jar：odbc14-10.2.0.4.0.jar
9. 建立数据库和表
10. 建立实体类

也可以直接用HashMap

1. 建立mapper接口:

|  |
| --- |
| 1. package com.ways.app.common.dao; 2. import java.util.List; 3. import java.util.Map; 4. public interface CommonMapper { 5. public List<Map<String, Object>> getUserList(Map<String,Object> paramsMap); 6. } |

1. 建立sql映射文件

|  |
| --- |
| 1. <?xml version=*"1.0"* encoding=*"UTF-8"*?> 2. <!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd"> 3. <mapper namespace=*"com.ways.app.common.dao.CommonMapper"*> 4. <select id=*"getUserList"* parameterType=*"java.util.HashMap"* resultType=*"java.util.HashMap"*> 5. select \* from da\_user 6. </select> 7. </mapper> |

1. 建立sqlMapConfig.xml

|  |
| --- |
| 1. <?xml version=*"1.0"* encoding=*"UTF-8"*?> 2. <!DOCTYPE configuration 3. PUBLIC "-//mybatis.org//DTD Config 3.0//EN" 4. "http://mybatis.org/dtd/mybatis-3-config.dtd"> 6. <configuration> 7. <!-- 和spring整合后 environments配置将废除--> 8. <environments default=*"development"*> 9. <environment id=*"development"*> 10. <!-- 使用jdbc事务管理 --> 11. <transactionManager type=*"JDBC"*/> 12. <!-- 数据库连接池 --> 13. <dataSource type=*"POOLED"*> 14. <property name=*"driver"* value=*"oracle.jdbc.OracleDriver"*/> 15. <property name=*"url"* value=*"jdbc:oracle:thin:@172.16.1.37:1521:DBFM"*/> 16. <property name=*"username"* value=*"FAWVW"*/> 17. <property name=*"password"* value=*"FAWvw419"*/> 18. </dataSource> 19. </environment> 20. </environments> 22. <!-- 查找sql映射文件 --> 23. <mappers> 24. <mapper resource=*"com/ways/app/common/sqlmap/oracle/commonMapper.xml"*/> 25. </mappers> 26. </configuration> |

1. 编写测试类

|  |
| --- |
| 1. **package** ssmTest; 2. **import** java.io.IOException; 3. **import** java.io.InputStream; 4. **import** java.util.HashMap; 5. **import** java.util.List; 6. **import** java.util.Map; 7. **import** org.apache.ibatis.io.Resources; 8. **import** org.apache.ibatis.logging.Log; 9. **import** org.apache.ibatis.logging.LogFactory; 10. **import** org.apache.ibatis.session.SqlSession; 11. **import** org.apache.ibatis.session.SqlSessionFactory; 12. **import** org.apache.ibatis.session.SqlSessionFactoryBuilder; 13. **import** org.junit.Test; 14. **import** com.ways.app.common.dao.CommonMapper; 15. **public** **class** MybatisTest { 16. @Test 17. **public** **void** test() **throws** IOException { 18. //1.创建SqlSessionFactoryBuilder 19. SqlSessionFactoryBuilder builder = **new** SqlSessionFactoryBuilder(); 20. // 加载sqlMapConfig.xml文件 21. InputStream ins = Resources.*getResourceAsStream*("sqlMapConfig.xml");  24. //2.创建SqlSessionFactory 25. SqlSessionFactory factory = builder.build(ins); 27. //3.打开SqlSession 28. SqlSession sqlSession = factory.openSession(); 30. //4.获取mapper接口的对象 31. CommonMapper commonMapper = sqlSession.getMapper(CommonMapper.**class**); 33. //操作 34. Map<String, Object> paramsMap = **new** HashMap<String, Object>(); 35. List<Map<String, Object>> userList = commonMapper.getUserList(paramsMap); 36. **for**(**int** i = 0; i < userList.size(); i++) { 37. Map<String, Object> userMap = userList.get(i); 38. System.***out***.println("用户名: " + userMap.get("USER\_NAME")); 39. } 40. //6.提交事务 41. sqlSession.commit(); 43. //7.关闭资源 44. sqlSession.close(); 46. } 47. } |

补充：maven pom.xml

|  |
| --- |
| <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.ways</groupId>  <artifactId>ssmTest</artifactId>  <version>0.0.1-SNAPSHOT</version>  <packaging>war</packaging>  <dependencies>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <version>4.12</version>  <scope>test</scope>  </dependency>  <dependency>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-compiler-plugin</artifactId>  <version>3.8.1</version>  <scope>provided</scope>  </dependency>  <dependency>  <groupId>org.apache.tomcat</groupId>  <artifactId>tomcat-servlet-api</artifactId>  <version>7.0.94</version>  <scope>provided</scope>  </dependency>  <dependency>  <groupId>org.apache.tomcat</groupId>  <artifactId>tomcat-jsp-api</artifactId>  <version>7.0.94</version>  <scope>provided</scope>  </dependency>  <dependency>  <groupId>org.ow2.asm</groupId>  <artifactId>asm</artifactId>  <version>7.1</version>  <scope>provided</scope>  </dependency>  <dependency>  <groupId>org.mybatis</groupId>  <artifactId>mybatis</artifactId>  <version>3.4.4</version>  </dependency>  <dependency>  <groupId>log4j</groupId>  <artifactId>log4j</artifactId>  <version>1.2.17</version>  </dependency>  <dependency>  <groupId>ways.oracle</groupId>  <artifactId>ojdbc14</artifactId>  <version>10.2.0.4.0</version>  </dependency>  </dependencies>        <build>  <plugins>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-compiler-plugin</artifactId>  <version>3.8.1</version>  <configuration>  <source>1.7</source>  <target>1.7</target>  <compilerArguments>  <extdirs>src\main\webapp\WEB-INF\lib</extdirs>  <bootclasspath>C:\Program Files\Java\jdk1.8.0\_172\jre\lib\rt.jar</bootclasspath>  </compilerArguments>  </configuration>  </plugin>  <plugin>  <groupId>org.apache.tomcat.maven</groupId>  <artifactId>tomcat7-maven-plugin</artifactId>  <version>2.2</version>  <configuration>  <port>8080</port>  <path>/</path>  </configuration>  </plugin>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-war-plugin</artifactId>  <version>2.6</version>  <!-- <configuration>  <webResources>  <webResource>  <directory>${project.basedir}/lib</directory>  <targetPath>WEB-INF/lib</targetPath>  <filtering>false</filtering>  <includes>  <include>\*\*/\*.jar</include>  </includes>  </webResource>  </webResources>  </configuration> -->  </plugin>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-jar-plugin</artifactId>  <version>3.0.2</version>  </plugin>  </plugins>    <resources>  <resource>  <directory>src/main/java</directory>  <includes>  <include>\*\*/\*.properties</include>  <include>\*\*/\*.xml</include>  </includes>  <filtering>false</filtering>  </resource>  <resource>  <directory>src/main/resources</directory>  <includes>  <include>\*\*/\*.properties</include>  <include>\*\*/\*.xml</include>  </includes>  <filtering>false</filtering>  </resource>  </resources>  </build>    <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  </properties>  </project> |

web.xml：

|  |
| --- |
| <?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"*  xmlns:jsp=*"http://java.sun.com/xml/ns/javaee/jsp"*  xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"*  version=*"3.0"*  >  <display-name>ssmTest</display-name>  </web-app> |

二、Mybatis整合spring - 有Mapper实现类

1.导入必须的包：

Mybatis-spring

Spring自身的包：

spring-ioc

spring-aop

spring-tx

spring-context

2.编写Mapper的实现类：

|  |
| --- |
| **package** com.ways.app.common.dao.impl;  **import** java.util.List;  **import** java.util.Map;  **import** org.mybatis.spring.support.SqlSessionDaoSupport;  **import** com.ways.app.common.dao.CommonMapper;  **public** **class** CommonMapperImpl **extends** SqlSessionDaoSupport **implements** CommonMapper {  @Override  **public** List<Map<String, Object>> getUserList(Map<String, Object> paramsMap) {  **return** **this**.getSqlSession().selectList("com.ways.app.common.dao.CommonMapper.getUserList", paramsMap);  //不需要事务提交 mybatis帮忙提交了  }  } |

3.编写applicationContext.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:aop=*"http://www.springframework.org/schema/aop"*  xmlns:tx=*"http://www.springframework.org/schema/tx"*  xmlns:jee=*"http://www.springframework.org/schema/jee"*    xsi:schemaLocation=*"http://www.springframework.org/schema/beans*  *http://www.springframework.org/schema/beans/spring-beans-3.0.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*  *http://www.springframework.org/schema/jee*  *http://www.springframework.org/schema/jee/spring-jee-2.0.xsd"*>    <!-- 读取jdbc.properties -->  <context:property-placeholder location=*"classpath:jdbc.properties"*/>    <!-- 创建数据源DataSource -->  <bean id=*"dataSource"* class=*"com.mchange.v2.c3p0.ComboPooledDataSource"* destroy-method=*"close"*>  <property name=*"driverClass"* value=*"${jdbc.driverClassName}"* />  <property name=*"jdbcUrl"* value=*"${jdbc.url}"* />  <property name=*"user"* value=*"${jdbc.username}"* />  <property name=*"password"* value=*"${jdbc.password}"* />  <property name=*"minPoolSize"* value=*"10"* /> <!-- 连接池中保留的最小连接数 -->  <property name=*"maxPoolSize"* value=*"100"* /> <!-- 连接池中保留的最大连接数。Default: 15 -->  <property name=*"maxIdleTime"* value=*"600"* /> <!-- 1800 --><!-- 最大空闲时间,600秒内未使用则连接被丢弃。若为0则永不丢弃。Default: 0 -->  <property name=*"acquireIncrement"* value=*"2"* />  <property name=*"maxStatements"* value=*"0"* />  <property name=*"initialPoolSize"* value=*"10"* />  <property name=*"idleConnectionTestPeriod"* value=*"0"* /> <!-- 60 --><!-- 每30秒检查所有连接池中的空闲连接。Default: 0 -->  <property name=*"testConnectionOnCheckout"* value=*"false"* />  <property name=*"acquireRetryAttempts"* value=*"0"* /> <!-- 定义在从数据库获取新连接失败后重复尝试的次数。Default: 30 -->  <property name=*"acquireRetryDelay"* value=*"1000"* /> <!-- 两次连接中间隔时间，单位毫秒。Default: 1000 -->  <property name=*"breakAfterAcquireFailure"* value=*"false"* />  </bean>    <!-- 创建SqlSessionFactory对象 -->  <bean id=*"sqlSessionFactory"* class=*"org.mybatis.spring.SqlSessionFactoryBean"*>  <!-- 关联连接池 -->  <property name=*"dataSource"* ref=*"dataSource"*/>  <!-- 加载sql映射文件集 -->  <property name=*"configLocation"* value=*"classpath:ibatic/sql-map-config.xml"* />  <!-- 加载sql映射具体文件路径 -->  <property name=*"mapperLocations"* value=*"classpath:mapper/\*Mappper.xml"*/>  </bean>    <!-- 创建CommonMapperImpl对象，注入SqlSessionFactory -->  <bean id=*"commonMapper"* class=*"com.ways.app.common.dao.impl.CommonMapperImpl"*>  <!-- 注入sqlSessionFactory -->  <property name=*"sqlSessionFactory"* ref=*"sqlSessionFactory"*/>  </bean>    </beans> |

4.编写测试类：

|  |
| --- |
| **package** ssmTest;  **import** java.util.HashMap;  **import** java.util.List;  **import** java.util.Map;  **import** org.junit.Test;  **import** org.springframework.context.ApplicationContext;  **import** org.springframework.context.support.ClassPathXmlApplicationContext;  **import** com.ways.app.common.dao.CommonMapper;  **public** **class** MybatisSpringTest {    @Test  **public** **void** test() {  //1.加载spring配置  ApplicationContext ac = **new** ClassPathXmlApplicationContext("applicationContext.xml");  //2.获取对象  CommonMapper commonMapper = (CommonMapper) ac.getBean("commonMapper");  //3.调用方法  Map<String, Object> paramsMap = **new** HashMap<String, Object>();  List<Map<String, Object>> userList = commonMapper.getUserList(paramsMap);  **for**(**int** i = 0; i < userList.size(); i++) {  Map<String, Object> userMap = userList.get(i);  System.***out***.println("用户名: " + userMap.get("USER\_NAME"));  }  }  } |

三、Mybatis整合spring - 没有Mapper实现类

修改applicationContext.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:aop=*"http://www.springframework.org/schema/aop"*  xmlns:tx=*"http://www.springframework.org/schema/tx"*  xmlns:jee=*"http://www.springframework.org/schema/jee"*    xsi:schemaLocation=*"http://www.springframework.org/schema/beans*  *http://www.springframework.org/schema/beans/spring-beans-3.0.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*  *http://www.springframework.org/schema/jee*  *http://www.springframework.org/schema/jee/spring-jee-2.0.xsd"*>    <!-- 读取jdbc.properties -->  <context:property-placeholder location=*"classpath:jdbc.properties"*/>    <!-- 创建数据源DataSource -->  <bean id=*"dataSource"* class=*"com.mchange.v2.c3p0.ComboPooledDataSource"* destroy-method=*"close"*>  <property name=*"driverClass"* value=*"${jdbc.driverClassName}"* />  <property name=*"jdbcUrl"* value=*"${jdbc.url}"* />  <property name=*"user"* value=*"${jdbc.username}"* />  <property name=*"password"* value=*"${jdbc.password}"* />  <property name=*"minPoolSize"* value=*"10"* /> <!-- 连接池中保留的最小连接数 -->  <property name=*"maxPoolSize"* value=*"100"* /> <!-- 连接池中保留的最大连接数。Default: 15 -->  <property name=*"maxIdleTime"* value=*"600"* /> <!-- 1800 --><!-- 最大空闲时间,600秒内未使用则连接被丢弃。若为0则永不丢弃。Default: 0 -->  <property name=*"acquireIncrement"* value=*"2"* />  <property name=*"maxStatements"* value=*"0"* />  <property name=*"initialPoolSize"* value=*"10"* />  <property name=*"idleConnectionTestPeriod"* value=*"0"* /> <!-- 60 --><!-- 每30秒检查所有连接池中的空闲连接。Default: 0 -->  <property name=*"testConnectionOnCheckout"* value=*"false"* />  <property name=*"acquireRetryAttempts"* value=*"0"* /> <!-- 定义在从数据库获取新连接失败后重复尝试的次数。Default: 30 -->  <property name=*"acquireRetryDelay"* value=*"1000"* /> <!-- 两次连接中间隔时间，单位毫秒。Default: 1000 -->  <property name=*"breakAfterAcquireFailure"* value=*"false"* />  </bean>    <!-- 创建SqlSessionFactory对象 -->  <bean id=*"sqlSessionFactory"* class=*"org.mybatis.spring.SqlSessionFactoryBean"*>  <!-- 关联连接池 -->  <property name=*"dataSource"* ref=*"dataSource"*/>  <!-- 加载sql映射文件集 -->  <property name=*"configLocation"* value=*"classpath:mapper/sqlMapConfig.xml"* />  <!-- 加载sql映射具体文件路径 -->  <!-- <property name="mapperLocations" value="classpath:mapper/\*Mappper.xml"/> -->  </bean>    <!-- 创建CommonMapperImpl对象，注入SqlSessionFactory -->  <!-- <bean id="commonMapper" class="com.ways.app.common.dao.impl.CommonMapperImpl">  注入sqlSessionFactory  <property name="sqlSessionFactory" ref="sqlSessionFactory"/>  </bean> -->    <!-- 配置Mapper接口 -->  <bean id=*"commonMapper"* class=*"org.mybatis.spring.mapper.MapperFactoryBean"*>  <!-- 注入mapper接口 -->  <property name=*"mapperInterface"* value=*"com.ways.app.common.dao.CommonMapper"*/>  <!-- 注入SqlSessionFactory -->  <property name=*"sqlSessionFactory"* ref=*"sqlSessionFactory"*/>  </bean>    </beans> |

1. Mybatis整合spring – mapper接口扫描
2. 修改applicationContext.xml

|  |
| --- |
| 1. <?xml version=*"1.0"* encoding=*"UTF-8"*?> 2. <beans xmlns=*"http://www.springframework.org/schema/beans"* 3. xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* 4. xmlns:context=*"http://www.springframework.org/schema/context"* 5. xmlns:aop=*"http://www.springframework.org/schema/aop"* 6. xmlns:tx=*"http://www.springframework.org/schema/tx"* 7. xmlns:jee=*"http://www.springframework.org/schema/jee"* 9. xsi:schemaLocation=*"http://www.springframework.org/schema/beans* 10. *http://www.springframework.org/schema/beans/spring-beans-3.0.xsd* 11. *http://www.springframework.org/schema/context* 12. *http://www.springframework.org/schema/context/spring-context.xsd* 13. *http://www.springframework.org/schema/aop* 14. *http://www.springframework.org/schema/aop/spring-aop.xsd* 15. *http://www.springframework.org/schema/tx* 16. *http://www.springframework.org/schema/tx/spring-tx.xsd* 17. *http://www.springframework.org/schema/jee* 18. *http://www.springframework.org/schema/jee/spring-jee-2.0.xsd"*> 20. <!-- 读取jdbc.properties --> 21. <context:property-placeholder location=*"classpath:jdbc.properties"*/> 23. <!-- 创建数据源DataSource --> 24. <bean id=*"dataSource"* class=*"com.mchange.v2.c3p0.ComboPooledDataSource"* destroy-method=*"close"*> 25. <property name=*"driverClass"* value=*"${jdbc.driverClassName}"* /> 26. <property name=*"jdbcUrl"* value=*"${jdbc.url}"* /> 27. <property name=*"user"* value=*"${jdbc.username}"* /> 28. <property name=*"password"* value=*"${jdbc.password}"* /> 29. <property name=*"minPoolSize"* value=*"10"* /> <!-- 连接池中保留的最小连接数 --> 30. <property name=*"maxPoolSize"* value=*"100"* /> <!-- 连接池中保留的最大连接数。Default: 15 --> 31. <property name=*"maxIdleTime"* value=*"600"* /> <!-- 1800 --><!-- 最大空闲时间,600秒内未使用则连接被丢弃。若为0则永不丢弃。Default: 0 --> 32. <property name=*"acquireIncrement"* value=*"2"* /> 33. <property name=*"maxStatements"* value=*"0"* /> 34. <property name=*"initialPoolSize"* value=*"10"* /> 35. <property name=*"idleConnectionTestPeriod"* value=*"0"* /> <!-- 60 --><!-- 每30秒检查所有连接池中的空闲连接。Default: 0 --> 36. <property name=*"testConnectionOnCheckout"* value=*"false"* /> 37. <property name=*"acquireRetryAttempts"* value=*"0"* /> <!-- 定义在从数据库获取新连接失败后重复尝试的次数。Default: 30 --> 38. <property name=*"acquireRetryDelay"* value=*"1000"* /> <!-- 两次连接中间隔时间，单位毫秒。Default: 1000 --> 39. <property name=*"breakAfterAcquireFailure"* value=*"false"* /> 40. </bean> 42. <!-- 创建SqlSessionFactory对象 --> 43. <bean id=*"sqlSessionFactory"* class=*"org.mybatis.spring.SqlSessionFactoryBean"*> 44. <!-- 关联连接池 --> 45. <property name=*"dataSource"* ref=*"dataSource"*/> 46. <!-- 加载sql映射文件集 --> 47. <property name=*"configLocation"* value=*"classpath:mapper/sqlMapConfig.xml"* /> 48. <!-- 加载sql映射具体文件路径 --> 49. <!-- <property name="mapperLocations" value="classpath:mapper/\*Mappper.xml"/> --> 50. </bean> 52. <!-- 创建CommonMapperImpl对象，注入SqlSessionFactory --> 53. <!-- <bean id="commonMapper" class="com.ways.app.common.dao.impl.CommonMapperImpl"> 54. 注入sqlSessionFactory 55. <property name="sqlSessionFactory" ref="sqlSessionFactory"/> 56. </bean> --> 58. <!-- 配置Mapper接口 --> 59. <!-- <bean id="commonMapper" class="org.mybatis.spring.mapper.MapperFactoryBean"> 60. 注入mapper接口 61. <property name="mapperInterface" value="com.ways.app.common.dao.CommonMapper"/> 62. 注入SqlSessionFactory 63. <property name="sqlSessionFactory" ref="sqlSessionFactory"/> 64. </bean> --> 66. <!-- Mappper接口的扫描 67. 注意：如果使用Mapper接口包扫描，那么每个Mapper接口在Spring容器中的id名称为类名：例如：CommonMapper -> commonMapper(首字母小写) 68. --> 69. <bean class=*"org.mybatis.spring.mapper.MapperScannerConfigurer"*> 70. <!-- 配置mapper接口所在的包路径 会扫描此包以及其子包--> 71. <property name=*"basePackage"* value=*"com.ways.app.common.dao"*/> 72. </bean> 74. </beans> |

1. Mybatis 整合spring – 整合jdbc事务

1.修改applicationContext.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:aop=*"http://www.springframework.org/schema/aop"*  xmlns:tx=*"http://www.springframework.org/schema/tx"*  xmlns:jee=*"http://www.springframework.org/schema/jee"*    xsi:schemaLocation=*"http://www.springframework.org/schema/beans*  *http://www.springframework.org/schema/beans/spring-beans-3.0.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*  *http://www.springframework.org/schema/jee*  *http://www.springframework.org/schema/jee/spring-jee-2.0.xsd"*>    <!-- 读取jdbc.properties -->  <context:property-placeholder location=*"classpath:jdbc.properties"*/>    <!-- 创建数据源DataSource -->  <bean id=*"dataSource"* class=*"com.mchange.v2.c3p0.ComboPooledDataSource"* destroy-method=*"close"*>  <property name=*"driverClass"* value=*"${jdbc.driverClassName}"* />  <property name=*"jdbcUrl"* value=*"${jdbc.url}"* />  <property name=*"user"* value=*"${jdbc.username}"* />  <property name=*"password"* value=*"${jdbc.password}"* />  <property name=*"minPoolSize"* value=*"10"* /> <!-- 连接池中保留的最小连接数 -->  <property name=*"maxPoolSize"* value=*"100"* /> <!-- 连接池中保留的最大连接数。Default: 15 -->  <property name=*"maxIdleTime"* value=*"600"* /> <!-- 1800 --><!-- 最大空闲时间,600秒内未使用则连接被丢弃。若为0则永不丢弃。Default: 0 -->  <property name=*"acquireIncrement"* value=*"2"* />  <property name=*"maxStatements"* value=*"0"* />  <property name=*"initialPoolSize"* value=*"10"* />  <property name=*"idleConnectionTestPeriod"* value=*"0"* /> <!-- 60 --><!-- 每30秒检查所有连接池中的空闲连接。Default: 0 -->  <property name=*"testConnectionOnCheckout"* value=*"false"* />  <property name=*"acquireRetryAttempts"* value=*"0"* /> <!-- 定义在从数据库获取新连接失败后重复尝试的次数。Default: 30 -->  <property name=*"acquireRetryDelay"* value=*"1000"* /> <!-- 两次连接中间隔时间，单位毫秒。Default: 1000 -->  <property name=*"breakAfterAcquireFailure"* value=*"false"* />  </bean>    <!-- 创建SqlSessionFactory对象 -->  <bean id=*"sqlSessionFactory"* class=*"org.mybatis.spring.SqlSessionFactoryBean"*>  <!-- 关联连接池 -->  <property name=*"dataSource"* ref=*"dataSource"*/>  <!-- 加载sql映射文件集 -->  <property name=*"configLocation"* value=*"classpath:mapper/sqlMapConfig.xml"* />  <!-- 加载sql映射具体文件路径 -->  <!-- <property name="mapperLocations" value="classpath:mapper/\*Mappper.xml"/> -->  </bean>    <!-- 创建CommonMapperImpl对象，注入SqlSessionFactory -->  <!-- <bean id="commonMapper" class="com.ways.app.common.dao.impl.CommonMapperImpl">  注入sqlSessionFactory  <property name="sqlSessionFactory" ref="sqlSessionFactory"/>  </bean> -->    <!-- 配置Mapper接口 -->  <!-- <bean id="commonMapper" class="org.mybatis.spring.mapper.MapperFactoryBean">  注入mapper接口  <property name="mapperInterface" value="com.ways.app.common.dao.CommonMapper"/>  注入SqlSessionFactory  <property name="sqlSessionFactory" ref="sqlSessionFactory"/>  </bean> -->    <!-- Mappper接口的扫描  注意：如果使用Mapper接口包扫描，那么每个Mapper接口在Spring容器中的id名称为类名：例如：CommonMapper -> commonMapper(首字母小写)  -->  <bean class=*"org.mybatis.spring.mapper.MapperScannerConfigurer"*>  <!-- 配置mapper接口所在的包路径 会扫描此包以及其子包-->  <property name=*"basePackage"* value=*"com.ways.app.common.dao"*/>  </bean>    <!-- 开启Spring ioc注解扫描 -->  <context:component-scan base-package=*"com.ways.app"*/>    <!-- 开启spring的事务 -->  <!-- 事务管理器 -->  <bean id=*"transactionManager"* class=*"org.springframework.jdbc.datasource.DataSourceTransactionManager"*>  <!-- 注入dataSource -->  <property name=*"dataSource"* ref=*"dataSource"*/>  </bean>  <!-- 启用Spring的事务注解 -->  <tx:annotation-driven transaction-manager=*"transactionManager"*/>    </beans> |

2.在业务方法上添加注解

|  |
| --- |
| **package** com.ways.app.common.service.impl;  **import** java.util.Map;  **import** javax.annotation.Resource;  **import** org.springframework.stereotype.Service;  **import** org.springframework.transaction.annotation.Transactional;  **import** com.ways.app.common.dao.CommonMapper;  **import** com.ways.app.common.service.CommonService;  @Service("commonService")  @Transactional  **public** **class** CommonServiceImpl **implements** CommonService {  //注入Mapper对象  @Resource  **private** CommonMapper commonMapper;  @Override  **public** **void** saveUser(Map<String, Object> paramsMap) {    commonMapper.saveUser(paramsMap);  **int** i = 100/0;  commonMapper.saveUser(paramsMap);  }        } |

3.测试方法：

|  |
| --- |
| package ssmTest;  import java.util.HashMap;  import java.util.List;  import java.util.Map;  import org.junit.Test;  import org.springframework.context.ApplicationContext;  import org.springframework.context.support.ClassPathXmlApplicationContext;  import com.ways.app.common.dao.CommonMapper;  import com.ways.app.common.service.CommonService;  public class MybatisSpringTest {    // @Test  public void test() {  //1.加载spring配置  ApplicationContext ac = new ClassPathXmlApplicationContext("applicationContext.xml");  //2.获取对象  CommonMapper commonMapper = (CommonMapper) ac.getBean("commonMapper");  //3.调用方法  Map<String, Object> paramsMap = new HashMap<String, Object>();  List<Map<String, Object>> userList = commonMapper.getUserList(paramsMap);  for(int i = 0; i < userList.size(); i++) {  Map<String, Object> userMap = userList.get(i);  System.out.println("用户名: " + userMap.get("USER\_NAME"));  }  }    @Test  public void test1() {  //1.加载spring配置  ApplicationContext ac = new ClassPathXmlApplicationContext("applicationContext.xml");  //2.获取对象  CommonService commonService = (CommonService) ac.getBean("commonService");  Map<String, Object> paramsMap = new HashMap<String, Object>();  paramsMap.put("userName", "zhaohy");  paramsMap.put("sex", 1);  paramsMap.put("hobby", "跑步打球运动");  commonService.saveUser(paramsMap);  }  } |