Spring_day3

Spring 春天 项目管理框架 作者:小陈

1. SM整合思路

1.导入mybatis的jar包

```
<groupId>org.mybatis
   <artifactId>mybatis</artifactId>
   <version>3.2.8
</dependency>
```

2.导入spring的jar包

```
<groupId>org.springframework
   <artifactId>spring-core</artifactId>
   <version>4.3.2.RELEASE
</dependency>
   <groupId>org.springframework
   <artifactId>spring-context</artifactId>
   <version>4.3.2.RELEASE
</dependency>
   <groupId>org.springframework
   <artifactId>spring-context-support</artifactId>
   <version>4.3.2.RELEASE
</dependency>
   <groupId>org.springframework
   <artifactId>spring-jdbc</artifactId>
   <version>4.3.2.RELEASE
</dependency>
```

```
<groupId>org.springframework
   <artifactId>spring-aop</artifactId>
   <version>4.3.2.RELEASE
</dependency>
   <groupId>org.springframework
   <artifactId>spring-beans</artifactId>
   <version>4.3.2.RELEASE
</dependency>
   <groupId>org.springframework
   <artifactId>spring-expression</artifactId>
   <version>4.3.2.RELEASE
</dependency>
   <groupId>org.springframework
   <artifactId>spring-aspects</artifactId>
   <version>4.3.2.RELEASE
</dependency>
   <groupId>org.springframework
   <artifactId>spring-tx</artifactId>
   <version>4.3.2.RELEASE
</dependency>
   <groupId>org.springframework
   <artifactId>spring-web</artifactId>
   <version>4.3.2.RELEASE
</dependency>
```

3.导入mybatis-spring包

4.导入数据库驱动jar

```
<dependency>
<groupId>mysql</groupId>
```


5.整合思路



2. 创建数据源

1.导入jar包

2.创建数据源对象

3. 整合Mybatis 框架

```
1.配置spring整合mybatis配置文件
<!-- 创建数据源 -->
<bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource">
    cproperty name="driverClassName" value="com.mysql.jdbc.Driver"/>
    cproperty name="url" value="jdbc:mysql://localhost:3306/test"/>
    property name="username" value="root"/>
    property name="password" value="root"/>
<!-- 创建sqlSessionFactoryBean -->
<bean id="sessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
    property name="dataSource" ref="dataSource"/>
    </bean>
<!-- 创建DAO
    注意:
       1.这个bean不用起名称 它将自动扫描DAO包中所有的接口类型,并将扫描的所有接口的实例在工厂中创建好
       2.默认接口实例在工厂中的名字为接口首字母小写 如: UserDAO---userDAO EmpDAO---empDAO
<bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
    property name="basePackage" value="com.baizhi.dao"/>
    property name="sqlSessionFactoryBeanName" value="sessionFactory"/>
2.生成mybatis的mapper配置文件提示
```

```
PublicID: -//mybatis.org//DTD Mapper 3.0//EN
address: http://mybatis.org/dtd/mybatis-3-mapper.dtd
```

4. 管理事务

```
spring提供的事务管理器:DataSourceTransactionManager
引入事务的命名空间
    <beans xmlns="http://www.springframework.org/schema/beans"</pre>
           xmlns:aop="http://www.springframework.org/schema/aop"
xmlns:tx="http://www.springframework.org/schema/tx"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 \verb|http://www.springframework.org/schema/beans/spring-beans-3.2.xsd|
                                 http://www.springframework.org/schema/aop
                                 http://www.springframework.org/schema/aop/spring-aop-3.2.xsd
                                 http://www.springframework.org/schema/tx
                                 http://www.springframework.org/schema/tx/spring-tx-3.2.xsd ">
<!-- 对业务的方法做详细配置-
    <tx:attributes
       c:attributes>
<tx:method name="save" propagation="REQUIRED" />
<tx:method name="update" propagation="REQUIRED"/>
<tx:method name="delete" propagation="REQUIRED"/>
<tx:method name="queryAll" propagation="SUPPORTS"/>
    </tx:attributes>
</tx:advice>
<!-- 配置事务切面 -
<aop:config>
    <aop:pointcut_expression="execution(* zpark.service.*.*(..))" id="pc"/>
    <aop:advisor advice-ref="txAdvice" pointcut-ref="pc"/>
</aop:config>
```

5. 事务的传播属性

```
propagation (传播属性):
REQUIRED 表示需要事务(如果没有事务,开启新的事务,如果事务已存在,则加入当前事务)
SUPPORTS 表示支持事务(如果没有事务,不开启新事务,如果事务已存在,则加入当前事务)
REQUIRES_NEW 每次开启新的事务
MANDATORY 强制事务,必须开启事务,没有事务就报错
NEVER 不支持事务,如果事务已经开始,报错
NOT_SUPPORTED,不支持事务,如果事务已经开始,不会加入当前事务
NESTED 有一些数据库不支持
```

6. 事务隔离级别

```
isolation(隔离级别):

DEFAULT 采用数据库的壁以隔离级别

脏读(另一方读到一方未提交的数据) READ_COMMITTED(避免脏读现象的发生) Oracle的壁以隔离级别为提交读
不可重复读(一方更新,另一方查询两次结果不一致) REPEATABLE_READ(可重复读,避免不可重复读这种情况,mysql的壁以隔离级别)

SERIALIZABLE

注意:隔离级别越高查询效率越低,一般采用数据库默认隔离级别
```

7. 读写性与异常

```
readOnly:
    true 本次事务中只能读,不能写 false 可读可写

<tx:method name="save" propagation="REQUIRED" read-only="false" rollback-for="java.lang.RuntimeException"/>
    <tx:method name="update" propagation="REQUIRED"/>
    <tx:method name="delete" propagation="REQUIRED"/>
    <tx:method name="dueryAll" propagation="SUPPORTS" read-only="true" no-rollback-for="java.lang.Exception" />
    rollback-for:出现什么异常回被事务,默认只有Error与RuntimeExceptin教者子类 出现异常才会回接
    如果指定其它异常可在rollback-for中修改如:rollback-for="Exception"

<tx:method name="save" propagation="REQUIRED" read-only="false" rollback-for="java.lang.RuntimeException"/>
    <tx:method name="update" propagation="REQUIRED"/>
    <tx:method name="delete" propagation="REQUIRED"/>
    <tx:method name="delete" propagation="REQUIRED"/>
    <tx:method name="queryAll" propagation="REQUIRED"/>
    <tx:method name="queryAll" propagation="SUPPORTS" read-only="true" no-rollback-for="java.lang.Exception"/>
```

8. SM 整合思路

- 1. 导入spring、mybatis、druid、mysql、mybatis-spring整合jar
- 2. 书写sql语句 创建表
- 3. 开发**实体**类
- 4. 书写**DAO接口**
- 5. 书写mapper配置文件
- 6. 开发业务接口类
- 7. 开发**业务实现类并注入DAO**对象
- 8. 开发spring配置文件
 - 创建数据源对象 并注入连接相关参数
 - 创建sqlSessionFactory对象并注入数据源和mapper文件位置
 - 创建DAO对象 并注入sqlSessionFactory以及DAO接口所在包
 - 创建Service对象并注入DAO组件
 - 创建事务管理器并注入数据源对象
 - 将事务管理器转为环绕通知,并做业务方法的细粒度配置
 - 配置事务切面
- = 9.启动工厂、并测试