

```
1 | <center> <font size="6" color="blue">声明式事务的注解方式</font></center>
```

1、 UserServiceImpl.java 加了注解Autowired之后， set和get就没有必要了

```
1 | @Autowired
2 | private UserMapper userMapper;
```

```
1 | /**
2 |  * 获取
3 |  */
4 | /* public UserMapper getUserMapper() {
5 |     return userMapper;
6 | }*/
```

```
1 | /**
2 |  * set注入
3 |  */
4 | /* public void setUserMapper(UserMapper userMapper) {
5 |     this.userMapper = userMapper;
6 | }*/
```

2、 声明式事务xml配置方式 (applicationContext.xml) 原来代码：

```
1 |     <!-- 配置声明式事务 -->
2 |     <!-- 事务管理器 -->
3 |     <bean id="transactionManager"
4 | class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
5 |         <property name="dataSource" ref="dataSource"></property>
6 |     </bean>
7 |     <!--事务增强 -->
8 |     <!-- <tx:advice id="advice" >
9 |         <tx:attributes>
10 |             <tx:method name="save*" propagation="REQUIRED" rollback-
11 | for="Exception"/>
12 |             <tx:method name="get*" read-only="true"
13 | propagation="REQUIRED"/>
14 |         </tx:attributes>
15 |     </tx:advice> -->
16 |     <!-- 切面配置 -->
17 |     <!-- <aop:config>
18 |         <aop:pointcut expression="execution(* com.javaxyz.service...*(..))" id="pointcut"/>
19 |         <aop:advisor advice-ref="advice" pointcut-ref="pointcut"/>
20 |     </aop:config> -->
```

3、 将声明式事务xml配置方式 (applicationContext.xml) 的相关代码注释：

```

1  <!--事务增强 -->
2  <!--      <tx:advice id="advice" >
3          <tx:attributes>
4              <tx:method name="save*" propagation="REQUIRED" rollback-
for="Exception"/>
5              <tx:method name="get*" read-only="true"
propagation="REQUIRED"/>
6          </tx:attributes>
7      </tx:advice> -->
8  <!-- 切面配置 -->
9  <!--      <aop:config>
10         <aop:pointcut expression="execution(* com.javaxyz.service...*.
(..))" id="pointcut"/>
11         <aop:advisor advice-ref="advice" pointcut-ref="pointcut"/>
12     </aop:config> -->
13     <!-- 扫描service实现类里的Transactional -->

```

4、声明式事务使用注解之后applicationContext.xml改造后的代码：

```

1      <!-- 配置声明式事务 -->
2      <!-- 事务管理器 -->
3      <bean id="transactionManager"
class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
4          <property name="dataSource" ref="dataSource"></property>
5      </bean>
6      <!--事务增强 -->
7      <!--      <tx:advice id="advice" >
8          <tx:attributes>
9              <tx:method name="save*" propagation="REQUIRED" rollback-
for="Exception"/>
10             <tx:method name="get*" read-only="true"
propagation="REQUIRED"/>
11             </tx:attributes>
12         </tx:advice> -->
13     <!-- 切面配置 -->
14     <!--      <aop:config>
15         <aop:pointcut expression="execution(* com.javaxyz.service...*.
(..))" id="pointcut"/>
16         <aop:advisor advice-ref="advice" pointcut-ref="pointcut"/>
17     </aop:config> -->
18     <!-- 对注解配置Transactional 的支持-->
19     <tx:annotation-driven/>

```

5、声明式事务使用注解之后UserServiceImpl.java改造后的代码

```

1  /**
2   * @Author: DongGaoYun
3   * @Description:
4   * @Date 2019-9-24 下午12:04:57
5   * @Version 1.0
6   * @Company: www.springhome.org
7   */
8  package com.javaxyz.service.impl;
9
10 import java.util.List;
11

```

```
12 import javax.annotation.Resource;
13
14 import org.springframework.beans.factory.annotation.Autowired;
15 import org.springframework.stereotype.Service;
16 import org.springframework.transaction.annotation.Propagation;
17 import org.springframework.transaction.annotation.Transactional;
18
19 import com.javaxyz.entity.User;
20 import com.javaxyz.mapper.UserMapper;
21 import com.javaxyz.service.UserService;
22
23 /**
24  * @ClassName: UserService.java
25  * @Description: 用户业务实现类
26  * @Author: DongGaoYun
27  * @URL: www.javaxyz.com 或 www.gyun.org
28  * @Email: DongGaoYun@qq.com
29  * @QQ: 1050968899
30  * @Weixin: QingYunJiao
31  * @WeixinGongZhongHao: JavaForum
32  * @Date: 2019-10-9 下午12:04:57
33  * @Version: 1.0
34  */
35 @Transactional
36 @Service("userService")
37 public class UserServiceImpl implements UserService {
38     // 注入接口 UserDao
39     @Resource
40     private UserMapper userMapper;
41
42     /**
43      * 获取
44      */
45     /*public UserMapper getUserMapper() {
46         return userMapper;
47     } */
48
49     /**
50      * set注入
51      */
52     /*public void setUserMapper(UserMapper userMapper) {
53         this.userMapper = userMapper;
54     } */
55     @Transactional(propagation=Propagation.REQUIRED)
56     @Override
57     public int saveUser(User user) {
58         // 保存用户信息
59         return userMapper.addUser(user);
60     }
61
62     @Override
63     public List<User> getUserByUserRoleChinaResultMap(User user) {
64         return userMapper.getUserByUserRoleChinaResultMap(user);
65     }
66
67     /*
68      * (non-Javadoc)
69      */

```

```
70      * @see com.javaxyz.service.UserService#saveUser(java.util.List)
71      */
72      @Transactional(propagation=Propagation.REQUIRED)
73      @Override
74      public int saveUser(List<User> listUser) {
75          int num = 0;
76          //遍历集合
77          for (User user : listUser) {
78              //调用保存方法，并统计保存数量
79              num += saveUser(user);
80              int num2 = 2 / 0;
81          }
82          return num;
83      }
84  }
```