```
<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

```
@Autowired
private UserMapper userMapper;
```

```
/**
 * 获取
 */
/*  public UserMapper getUserMapper() {
        return userMapper;
    }*/
```

```
/**
 * set注入
 */
/*  public void setUserMapper(UserMapper userMapper) {
        this.userMapper = userMapper;
    }*/
```

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

```
    <!-- 配置声明式事务 -->
    <!-- 事务管理器 -->
    <bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
        <property name="dataSource" ref="dataSource"></property>
    </bean>
    <!--事务增强 -->
<!--    <tx:advice id="advice" >
        <tx:attributes>
            <tx:method name="save*" propagation="REQUIRED" rollback-for="Exception"/>
            <tx:method name="get*" read-only="true" propagation="REQUIRED"/>
        </tx:attributes>
    </tx:advice> -->
    <!-- 切面配置 -->
<!--    <aop:config>
        <aop:pointcut expression="execution(* com.javaxyz.service..**.*(..))" id="pointcut"/>
        <aop:advisor advice-ref="advice" pointcut-ref="pointcut"/>
    </aop:config> -->
```

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

```xml
<!--事务增强 -->
<!--    <tx:advice id="advice" >
        <tx:attributes>
            <tx:method name="save*" propagation="REQUIRED" rollback-for="Exception"/>
            <tx:method name="get*" read-only="true" propagation="REQUIRED"/>
        </tx:attributes>
    </tx:advice> -->
    <!-- 切面配置 -->
<!--    <aop:config>
        <aop:pointcut expression="execution(* com.javaxyz.service..**.*(..))" id="pointcut"/>
        <aop:advisor advice-ref="advice" pointcut-ref="pointcut"/>
    </aop:config> -->
    <!-- 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：

```xml
    <!-- 配置声明式事务 -->
    <!-- 事务管理器 -->
    <bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
        <property name="dataSource" ref="dataSource"></property>
    </bean>
    <!--事务增强 -->
<!--    <tx:advice id="advice" >
        <tx:attributes>
            <tx:method name="save*" propagation="REQUIRED" rollback-for="Exception"/>
            <tx:method name="get*" read-only="true" propagation="REQUIRED"/>
        </tx:attributes>
    </tx:advice> -->
    <!-- 切面配置 -->
<!--    <aop:config>
        <aop:pointcut expression="execution(* com.javaxyz.service..**.*(..))" id="pointcut"/>
        <aop:advisor advice-ref="advice" pointcut-ref="pointcut"/>
    </aop:config> -->
    <!-- 对注解配置Transactional 的支持-->
    <tx:annotation-driven/>
```

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

```java
/**
 * @Author: DongGaoYun
 * @Description:
 * @Date 2019-9-24  下午12:04:57
 * @Version 1.0
 * @Company: www.springhome.org
 */
package com.javaxyz.service.impl;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.javaxyz.entity.User;
import com.javaxyz.mapper.UserMapper;
import com.javaxyz.service.UserService;

/**
 * @ClassName：UserService.java
 * @Description：用户业务实现类
 * @Author：DongGaoYun
 * @URL：www.javaxyz.com 或 www.gyun.org
 * @Email：DongGaoYun@qq.com
 * @QQ：1050968899
 * @WeiXin：QingYunJiao
 * @WeiXinGongZhongHao: JavaForum
 * @Date：2019-10-9 下午12:04:57
 * @Version：1.0
 */
@Transactional
@Service("userService")
public class UserServiceImpl implements UserService {
    // 注入接口  UserDao
    @Resource
    private UserMapper userMapper;

    /**
     * 获取
     */
    /*public UserMapper getUserMapper() {
        return userMapper;
    } */

    /**
     * set注入
     */
    /*public void setUserMapper(UserMapper userMapper) {
        this.userMapper = userMapper;
    } */
    @Transactional(propagation=Propagation.REQUIRED)
    @Override
    public int saveUser(User user) {
        // 保存用户信息
        return userMapper.addUser(user);
    }

    @Override
    public List<User> getUserByUserRoleChinaResultMap(User user) {
        return userMapper.getUserByUserRoleChinaResultMap(user);
    }

    /*
     * (non-Javadoc)
     *
     * @see com.javaxyz.service.UserService#saveUser(java.util.List)
     */
    @Transactional(propagation=Propagation.REQUIRED)
    @Override
    public int saveUser(List<User> listUser) {
        int num = 0;
        //遍历集合
        for (User user : listUser) {
            //调用保存方法，并统计保存数量
            num += saveUser(user);
            int num2 = 2 / 0;
        }
        return num;
    }
}
```