**每日作业卷答案**

**Spring第3天**

极客营

# 关卡1

## 训练案例

### 训练描述

现在有UserServiceImpl的类，实现了UserService的接口，该类中有2个方法，都已经进行了实现。需求是在不改变UserServiceImpl类源代码的情况下，对save方法进行增强，要求使用的是JDK动态代理的方式进行增强。增强只要在控制台上输入一句内容即可。内容随意编写。

### 操作步骤答案

1. 创建代理类MyJdkProxy  
   public class MyJdkProxy
2. 在MyJdkProxy类中编写方法。对UserServiceImpl类进行增强，编写增强的代码。使用JDK的动态代理增强。  
   public class MyJdkProxy {

public static UserService getProxy(final UserService us){

// 获取到代理对象

Object proxy = Proxy.newProxyInstance(MyJdkProxy.class.getClassLoader(), us.getClass().getInterfaces(), new InvocationHandler() {

public Object invoke(Object proxy, Method method, Object[] args) throws Throwable {

if(method.getName().equals("save")){

System.out.println("增强save方法了...");

}

return method.invoke(us, args);

}

});

return (UserService) proxy;

}

}

1. 修改demo1类中的run1测试方法，获取到代理对象。调用代理对象的方法进行增强。  
   @Test

public void run1(){

UserService us = new UserServiceImpl();

us.save();

us.delete();

UserService proxy = MyJdkProxy.getProxy(us);

proxy.save();

proxy.delete();

}

# 关卡2

## 训练案例1

### 训练描述

使用Spring框的AOP配置文件的方式对UserServiceImpl的save方法进行增强，增强的功能是在控制台上打印输出（使用前置通知）。

### 操作步骤描述

1. 在applicationContext.xml配置文件中配置通知类。

<bean id="myAdvice" class="com.igeek.demo1.MyAdvice"></bean>

1. 在applicationContext.xml配置文件中配置AOP的增强  
   <aop:config>

<aop:aspect ref="myAdvice">

<aop:before method="writeLog" pointcut="execution(public \* com.igeek.demo1.UserServiceImpl.save(..))"/>

</aop:aspect>

</aop:config>

# 关卡3

## 训练案例1

### 训练描述

使用Spring框架的AOP注解的方式对UserServiceImpl的save方法进行增强，增强的功能是在控制台上打印输出（使用前置通知）。

### 操作步骤描述

1. 在MyAdvice类上添加注解，把该类交给Spring的IOC容器管理  
   @Component(value="myAdvice")
2. 在MyAdvice类上添加注解，声明该类为切面类  
   @Aspect
3. 在applicationContext.xml配置文件中开启Spring框架的自动代理  
   <aop:aspectj-autoproxy/>
4. 在MyAdvice类中的writeLog方法上添加前置通知的注解，完成增强  
   @Before(value="execution(public \* com.igeek.demo1.UserServiceImpl.save(..))")

# 关卡4

## 训练案例

### 训练描述

使用Spring框架的AOP技术模拟事务管理。现在已经提供了保存用户数据到数据库的代码，但是service层没有添加事务代码，程序运行是不能保存成功的，需要使用AOP技术对service的程序进行增强，进行事务管理。

### 操作步骤描述

1. 先在MySQL数据库中创建名称为spring\_day01的数据库  
   create database spring\_day01;

2. 制作事务的通知（在TxAdvice类中tx方法中进行事务的增强代码编写）  
public void tx(ProceedingJoinPoint joinPoint){

// 获取到session对象

Session session = HibernateUtils.getCurrentSession();

Transaction tx = session.beginTransaction();

try {

// 执行目标对象的方法

joinPoint.proceed();

tx.commit();

} catch (Throwable e) {

tx.rollback();

e.printStackTrace();

}

}

3. 使用IOC注解的方式管理事务通知类  
  
@Component(value="txAdvice")

public class TxAdvice {

4. 使用注解Aspect标识通知类为切面  
  
@Component(value="txAdvice")

@Aspect

public class TxAdvice {

5. 开启自动代理  
  
<aop:aspectj-autoproxy/>

6. 在tx方法上使用环绕通知进行AOP的增强  
@Around(value="execution(public \* com.igeek.service.impl.UserServiceImpl.save(..))")