13 Spring_AOP_相关术语

`Spring`

一、Joinpoint(连接点):

> 1、目标对象中,所有都<可以增强>的方法,叫做连接点

二、Pointcut(切入点):

> 1、目标对象,<已经增强>的方法,叫做切入点

三、Advice(通知/增强):

> 1、<增强的>代码
> 2、所谓通知是指拦截到Joinpoint之后所要做的事情就是通知.
> |-通知分为
> |-前置通知
> -目标方法运行前调用
> |-后置通知
> -目标方法运行后调用
> |-异常通知
> -如出现异常就会调用
> |-最终通知 (无论是否出现异常都会调用)
> -目标方法运行后调用
> |-环绕通知(切面要完成的功能)
> -目标方法运行前后调用

四、Target(目标对象):

> 1、代理的目标对象(被代理的对象)

五、Weaving(织入):

> 1、将通知应用到切入点的过程

六、Proxy（代理）:

> 1、将通知织入到目标对象之后,而形成代理对象

七、Aspect(切面):

> 1、切入点和通知（引介）的结合

```java
public class UserSerivceImp implements IUserSerivce {
    @Override
    public void save() throws Exception {
        System.out.println("save:查询用户");
    }
    @Override
    public void del() throws Exception {
        System.out.println("del:删除用户");
    }
    @Override
    public void upde() throws Exception {
        System.out.println("upde:修改用户");
    }
    @Override
    public void add() throws Exception {
        System.out.println("add:添加用户");
    }
}
```

Target(目标对象)

Joinpoint(连接点)

Joinpoint(连接点)

Joinpoint(连接点)

Joinpoint(连接点)

```java
@Override
public Object intercept(Object obj, Method method, Object[] arg,
    System.out.println("---------开启事务");
    // 调用原有方法
    Object invokeSuper = methodProxy.invokeSuper(obj, arg);
    System.out.println("提交事务---------");

    return invokeSuper;
}

@Test
public void getVoid01() throws Exception{
    //获取接口对象
    IUserSerivce  userSerivce=new UserSerivceImp();
    //获取动态代理工厂
    CglibProxyFactory proxyFactory = new  CglibProxyFactory();
    //获取代理对象
    IUserSerivce serivceProxyFactory = proxyFactory.getCglibProxyFactory(
    serivceProxyFactory.add();
    //判断代理对象是否属于被代理对象类型
```

Advice(通知/增强)

Advice(通知/增强)

Proxy（代理）