Spring AOP是构成Spring框架的重要基石，spring AOP 构建在IOC

之上，和ioc浑然天成，统一于Spring容器之中。

AOP有特定的应用场合：适合于具有特定逻辑的应用场合，如性能检测，访问控制，事物管理以及日志记录。

AOP(Aspect Oriented Programing)面向切面编程，根据软件重构的思想，如果多个类中出现相同的代码，将这些相同的代码提取到父类中，AOP通过横向抽取机制为这类无法通过纵向继承体系进行抽象的重复性代码提供了解决方按。

AOP术语：

1. 连接点（JoinPooint）：所谓连接点是指那些被拦截到的点。在spring中，这些点指的是方法，因为spring只支持方法类型的连接点。（即指的是业务层的所有方法）
2. 切入点（Pointcut）：所谓切入点是指我们要对哪些Joinpoint进行拦截的定义。

（即指的是业务层的所有被增强的方法）

1. 通知/增强（Advice）：所谓通知是指拦截到Joinpoint之后所要做的事情就是通知。通知的类型：前置通知（开启事务），后置通知（事务提交），异常通知（事务回滚），最终通知（释放资源），环绕通知（整个invoke方法）。

注意：在环绕通知中有名且的切入点方法调用

4，目标对象（Target）：代理的目标对象（userDao）

5，引介（Introduction）：引介是一种特殊的通知在不修改类代码的前提下，Introduction可以在运行期为类动态地添加一些方法或Field。 （了解）

6，织入（Weaving）：是指把增强应用到目标对象来创建新的代理对象的过程。Spring采用

代理织入，而AspectJ采用编译期织入和类装载期织入。 即指的是加入事务支持的过程。

7，代理（Proxy）： 一个类被AOP增强后，就产生一个结果代理类。（即指的是最后返回的对象）

8，切面（Aspect）：是切入点和通知（引介）的结合。

总的来说：AOP的工作重心在于如何将增强应用于目标对象的连接点上，这包括第一，如何通过切点和增强定位到连接点，第二，如何在增强中编写切面代码。

Execution（用于匹配是连接点的执行方法）

语法结构：execution( 方法修饰符  方法返回值 所属类 匹配方法名 ( 方法中的形参表)  方法申明抛出的异常 )

其中红色字体的部分时不能省略的，各部分都支持通配符 “\*” 来匹配全部。

　　比较特殊的为形参表部分，其支持两种通配符

* "\*"：代表一个任意类型的参数；
* “..”：代表零个或多个任意类型的参数。

