# Spring中AOP

1,什么是AOP?

面向切面的编程,是程序用动态代理技术,对程序功能进行统一维护的一种技术.

AOP能实现,在不改变原代码的情况下,添加新功能.

2,AOP的通知方式

<aop:before method=*""*/>

<aop:around method=*"run"* pointcut-ref=*"point1"*/>

<aop:after method=*""*/>

<aop:after-returning method=*""*/>

<aop:after-throwing method=*""*/>

before 前置通知

around 环绕通知

after-returning 后置通知

after 最终通知

after-throwing 异常通知

3,切面

就是一段统一的代码,能对横切的方法做统一的维护.

4,连接点

spring中就是我们横切的哪个方法.

5,环绕通知和其他通知的区别

5.1 环绕通知里面包含了其他四种通知方式

5.2 入参不同, 环绕通知可以用ProceedingJoinPoint,其他通知只能用JoinPoint

5.3 环绕通知里包含了目标的方法,在目标方法执行前和执行后都在环绕通知里.

6,横切的写法

expression=*“execution(\* com.turing.spring.aop.\*.\*(..))"*

横切com.turing.spring.aop包内的任何类中的任何方法,任何返回类型,任何类型和个数的参数.

7,典型的应用场景

事务管理和日志管理

(安全控制,性能监控,统一异常处理)

8,用aop给项目添加日志

8.1. 确定业务

我们对所有的控制层做横切,添加日志信息

日志格式为ip [时间] [登录人] 类名.方法名

8.2 在springMVC的配置文件中添加代码

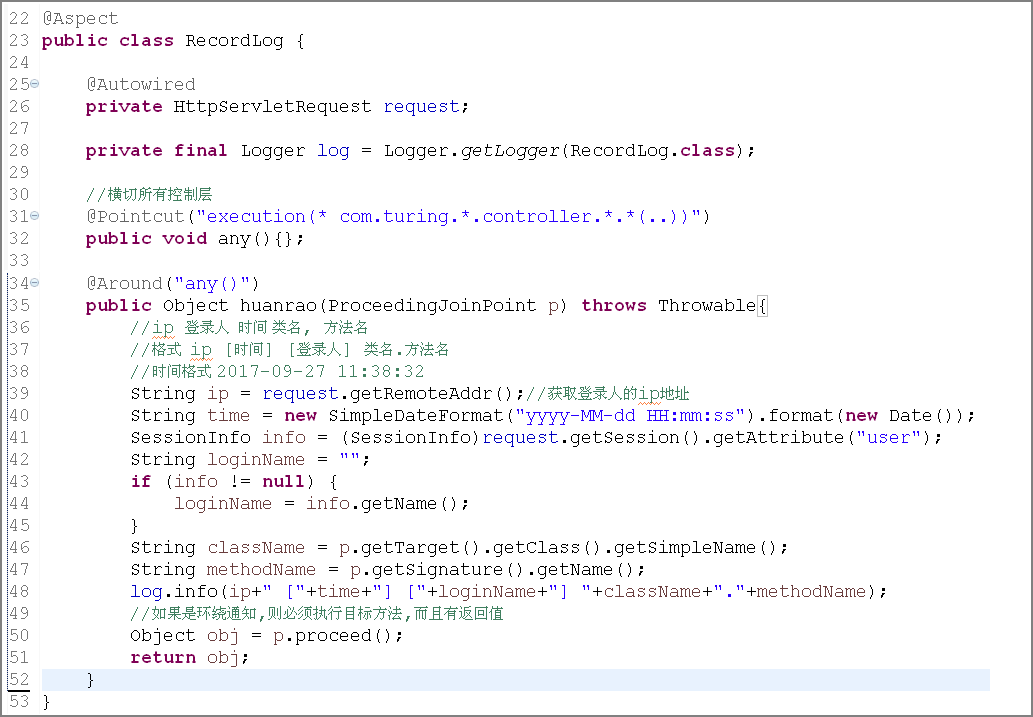
<!-- 启动aop的注解 -->

<aop:aspectj-autoproxy></aop:aspectj-autoproxy>

<!-- 设置一个日志的代理 -->

<bean class="com.turing.framework.log.RecordLog"></bean>

8.3 在日志代理类中,添加代码,(此处采用环绕通知方式)



10,面试题

44.什么是AOP

45.项目中哪些地方可能会用到AOP

46.AOP中，什么是切面

一段共同的代码称之为切面，例如日志，事务等。

47.AOP中，什么是连接点

应用程序执行过程中插入切面的地点, 在spring中,这些点指的是方法.

48.AOP的通知方式有哪些

49.后置通知与最终通知的区别

50.环绕通知与其他通知方式的区别

5.1 环绕通知里面包含了其他四种通知方式

5.2 入参不同, 环绕通知可以用ProceedingJoinPoint,其他通知只能用JoinPoint

5.3 环绕通知里包含了目标的方法,在目标方法执行前和执行后都在环绕通知里.

51.如何利用注解实现AOP

首先在spring的配置文件中启动AOP的注解功能，然后写一个bean节点用于做AOP的实现。

在AOP的实现类上写一个注解@Aspect，让这个类作为一个切面。在类中方法上写@Before等注解当做AOP的通知。

然后在一个方法上写@PointCut注解来确定切入点，即在哪些方法上执行AOP。

最后将PointCut注解所在方法的方法名写在通知方式的注解中来确定哪些切入点使用哪些通知方式。