# Spring AOP

AOP(面向切面编程)是编程范式，是一种程序设计思想，下面是AOP的概念：

* 切面（Aspect），关注点的模块化，这个关注点会横切多个对象
* 连接点（JoinPoint），在程序执行过程中某个特点的点，比如某方法调用或者处理一次，在Spring AOP中，一个连接点总是表示方法的执行
* 通知（Advice），在切面的某个特点连接点上执行的动作，其中包括around/before和after等不同类型的通知。许多AOP框架都使用拦截器做通知模型并维护一个以连接器为中心的拦截器链
* 切入点（Pointcut），匹配连接点的断言，通知和一个切入点表达式关联，并在满足这个切入点的连接点上运行。切入点表达式如何和连接点匹配是AOP的核心。Spring缺省使用Aspect切入点语法
* 引入（Introduction），用来给一个类型声明额外的方法或者属性，Spring引入新的接口到任何被代理的对象，例如可以使一个bean实现IsModified接口，以便简化缓存机制
* 目标代理（Target Object），被一个或者多个切面所通知的对象
* AOP代理（AOP Proxy），AOP框架创建的对象，用来实现切面契约（如通知方法执行等），在Spring中AOP代理可以是JDK动态代理或者CGLib代理
* 织入（Weaving），把切面连接到其他应用程序类型或者对象上，并创建一个被通知的对象。其通知的类型包括：

1. 前置通知（Before advice），在某个连接点之前执行的通知，但是该通知不能阻止连接点之前的执行流程
2. 后置通知（After returning advice），在某连接点正常完成后执行的通知，例如一个方法没有抛出任何异常，正常返回
3. 异常通知（After throwing advice），在方法抛出异常退出时执行的通知
4. 最终通知（After Advice)，当某连接点退出时执行的通知
5. 环绕通知（Around Advice），包围一个连接点的通知，如方法调用。环绕通知是最常用的通知类型。

AOP框架是Spring的重要组成部分，是Spring IOC容器的补充，切面使用普通的bean定义语法来配置，其作用：

* 提供声明式企业服务，特别是替代EJB声明式服务
* 允许用户实现自定义切面，用AOP来完善OOP的使用

Spring使用J2SE动态代理（dynamic proxies）来作为AOP的代理，这样任何接口都可以被代理。

https://github.com/xkcoding/spring-boot-demo/tree/master/spring-boot-demo-aoplog

http://shouce.jb51.net/spring/aop.html

https://docs.spring.io/spring/docs/2.5.x/reference/aop.html