# SpringAop的执行顺序

## 01、分析

在项目中，很多不同业务会使用aop技术取对应解决。这个aop执行顺序问题就浮现出来。如何保证呢？

- 日志AOP
- 限流AOP
- 权限校验AOP
- ……

## 02、日志AOP

作用：可以完成对一些核心业务接口的方法进行性能的排查和优化。

## 03、限流AOP

作用：如果一些核心业务并发量非常的大，可以使用限流进行优化和处理。

后端技术：redis + lua

前端技术：节流 + 防抖

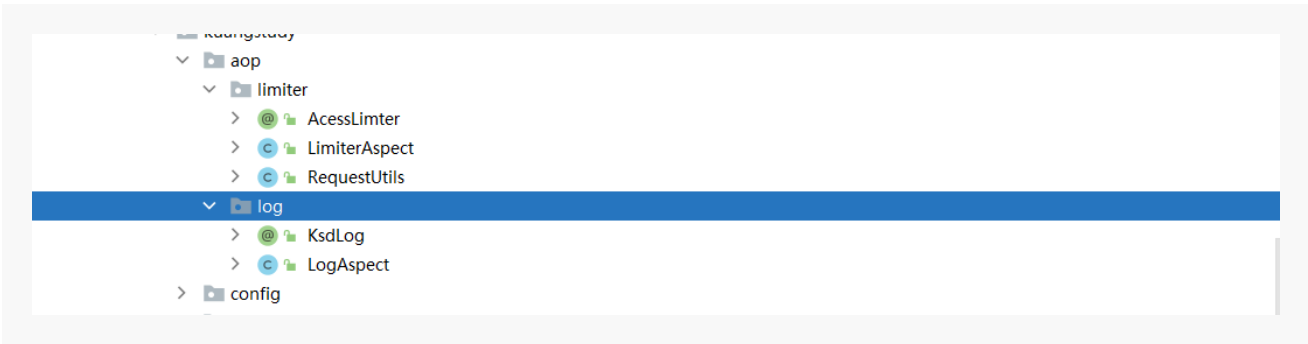
## 04、AOP顺序保证具体实现

目的：

1、明白aop的执行顺序，解决业务问题

2、明白看源码的时候，@Order 或者 实现Ordered接口的时候。

## 代码截图




## 第一种方式@Order注解

```
@Component // 让aop切面让springioc来管理变成一家人
@Aspect // 代表是一个aop切面
@Slf4j
@Order(0)
public class LogAspect {
}
```

```
@Component
@Aspect
@Slf4j
@Order(1)
public class LimiterAspect {
}
```

注意：@Order注解的数字越小越先进入。所以上面的结论是：先进日志 --- 限流

## 第二种方式：实现Ordered接口

实现ordered接口覆盖getOrder方法，方法返回数字越小越先执行

```
package com.kuangstudy.aop.log;

import lombok.extern.slf4j.Slf4j;
import org.aspectj.lang.ProceedingJoinPoint;
import org.aspectj.lang.annotation.Around;
import org.aspectj.lang.annotation.Aspect;
import org.aspectj.lang.annotation.Pointcut;
import org.aspectj.lang.reflect.MethodSignature;
import org.springframework.core.Ordered;
import org.springframework.core.annotation.Order;
import org.springframework.stereotype.Component;

/**
 * @author 飞哥
 * @Title: 学相伴出品
 * @Description: 飞哥B站地址:
https://space.bilibili.com/490711252
 * 记得关注和三连哦!
 * @Description: 我们有一个学习网站：https://www.kuangstudy.com
 * @date 2021/12/21 23:31
 */
@Component // 让aop切面让springioc来管理变成一家人
@Aspect // 代表是一个aop切面
@Slf4j
public class LogAspect implements Ordered {

    @Override
    public int getOrder(){
        return 0;
    }
}

```

## 05、总结

- springaop的切面的执行是可以保证执行顺序：
  - @Order注解
  - 实现Ordered接口覆盖 getOrder方法
- 在开发中，两者也可以混用。
- 如果和内部的出现冲突，你在进行修改，不要先考虑那么多。只要范围在：

```
package org.springframework.core;

public interface Ordered {
    int HIGHEST_PRECEDENCE = -2147483648;
    int LOWEST_PRECEDENCE = 2147483647;

    int getOrder();
}

```

- 两者都一样：数字越小越先执行。