#### Spring Boot

##### 之面向切面编程（AOP） -Tracing

问题描述

Spring Boot 2.2.5

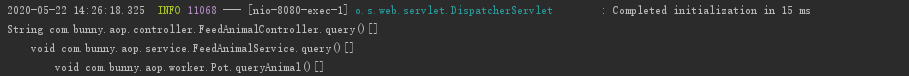
JDK 8

面向切面编程（AOP）可用于代码追踪（Tracing）

解决

@Aspect  
@Component  
public class TraceAspect {  
  
 private int level = 0;  
  
 @Pointcut("within(com.bunny.aop..\*)")  
 public void trace() {  
  
 }  
  
 @Before("trace()")  
 public void doTrace(JoinPoint point) {  
 for (int i = 0; i < level; i++) {  
 System.*out*.print("\t");  
 }  
 System.*out*.println(point.getSignature() + Arrays.*toString*(point.getArgs()));  
 level++;  
 }  
  
 @After("trace()")  
 public void doneTrace(JoinPoint point) {  
 level--;  
 }  
}

效果预览



笔记

1. 极大地方便了legacy code（陈旧代码）的维护工作：使用IDE进行静态分析费时费力。Tracing动态打印调用栈清晰明了地观察程序的执行逻辑。通过调整函数入参，观察调用栈的变化，方便了解函数功能
2. within: Limits matching to join points within certain types (the execution of a method declared within a matching type when using Spring AOP).

□