# 让你面试脱颖而出的技能-Java动态代理

主讲：鲁班

时间：2018/8/14 8:00

地址：腾讯课堂-图灵学院

**课程概要：**

1. 动态代理本质探究
2. JAVA Proxy 实现过程
3. JAVA 应用协议代理 （FTP、Http、Https、File）

**讲师介绍：**



一、动态代理和Class 字节码的关系

**提问：**

动态代理有什么作用 及应用场景？

1. 日志集中打印
2. 事务
3. 权限管理
4. AOP

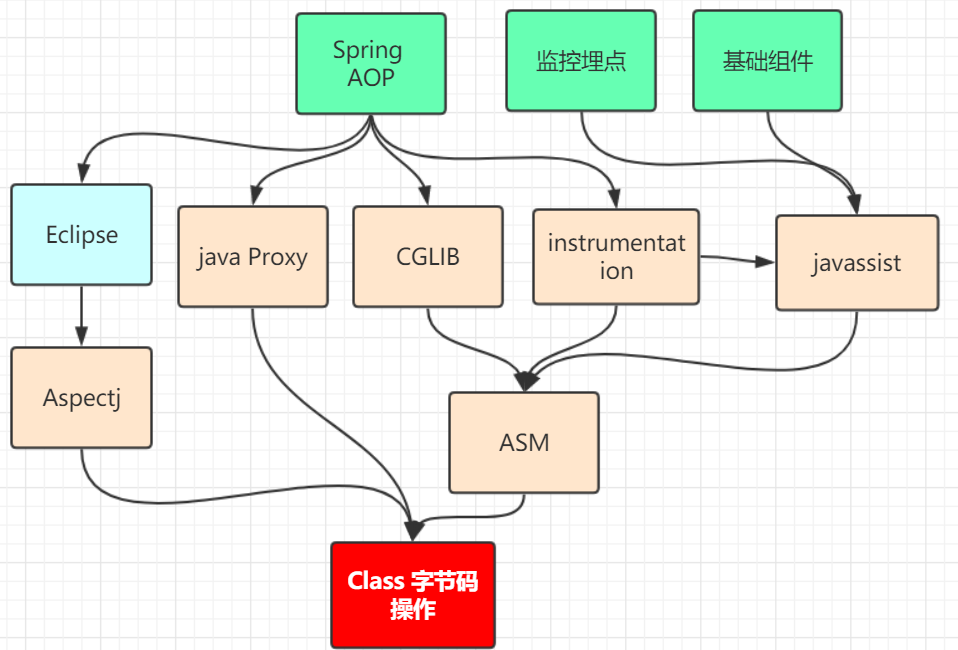
**在Spring Aop 当中可以哪些方式实现,及区别？**

1. java Proxy (动态构建字节码) (动态构建全新字节码 bean 初始化的时候)
2. cglib (动态构建字节码) (动态构建全新字节码 bean 初始化的时候)
3. Aspectj（修改目标类的字节，织入代理的字节，在程序编译（编译的时候 插入动态代理的字节码，不会生成全新的Class ）
   1. instrumentation（修改目标类的字节码、类装载的时候动态拦截去修改，基于javaagent） -:spring-instrument-4.3.8.RELEASE.jar （类装载的时候 插入动态代理的字节码，不会生成全新的Class ）

从中是否发现了动态代理的底层秘密？

无论哪种方式实现动态代理，其本质实现都是对字节码的修改。其区别是从哪里进行切入修改字节码。

**动态代理技术栈图：**



**总结：**

1. 动态代理可以由 java proxy、cglib、instrumentation(javaagent)、Aspectj、等多种方式实现
2. 动态代理本质是对class 字节码进行动态构建或修改。
3. 修改的工具有 ASM 、javavssist
4. 多种实现方式的区别在于 对字节码切入方式不一样。可选方式的有：
5. java proxy 、Cglib 是基于动态构建 接口实现类字节
6. AspectJ 是借助 Eclipse工具 在编译时织入代理字节
7. instrumentation 是基于 javaagent 在类装载时修改Class 织入代理字节
8. 使用字定义ClassLoader 在装载时织入代理字节

## JAVA Proxy 实现过程

**演示java Proxy 的基本使用**

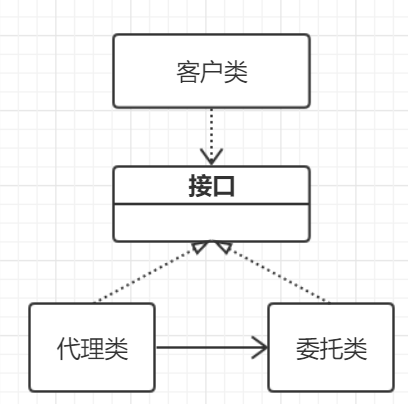
编写一个简单接口

编写invokert

使用proxys 生成代理对象

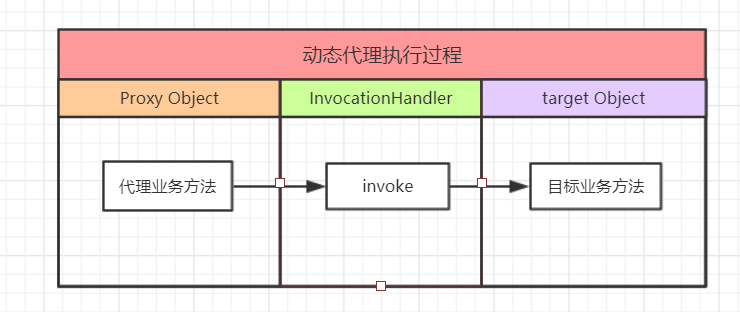
运行输出

**在整个示例当中有 客户类、接口、代理对象、目标对象他们的关系是什么?**



**代理对象是如何做到代理目标对象的呢？**

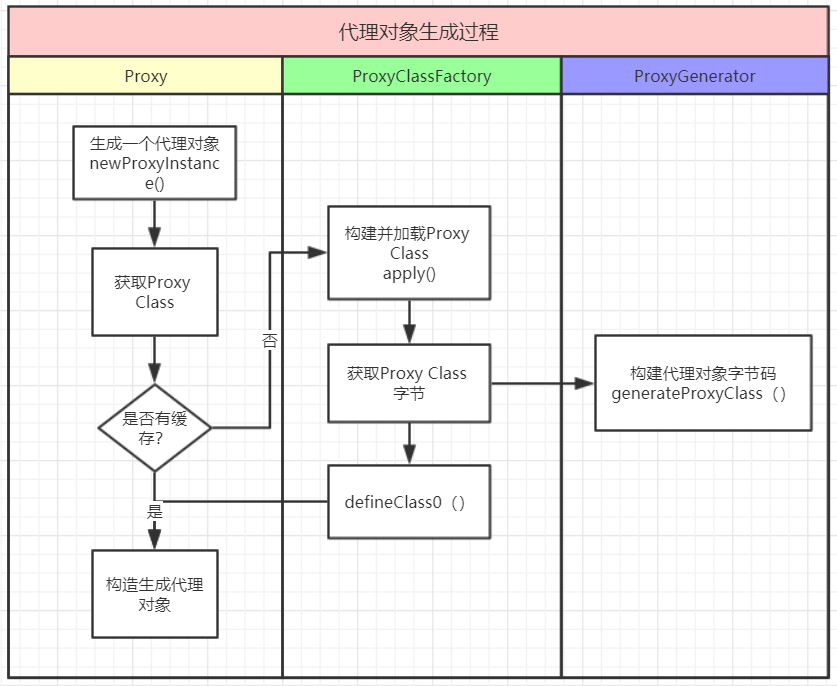
其中关键就在于 InvocationHandler接口，由该接口将 代理对象、目标对象、和代理逻辑连接在一起，下图是接口调用的实际执行过程。



**最后一个问题:代理对象是如何生成的？**

从上一小节我们知道 java proxy 在构建代理对象的过程当中是构建了一个新的Class。其构建过程如下：

1. Proxy 基于代理接口获取其代理Class
2. 从缓存中获取，如果没有则继续一下步
3. 使用ProxyGenerator 构建代理Class字节
4. 调用本地方法装载Class字节 至当前ClassLoader
5. 返回新的Class 对象
6. 使用反谢生成代理对象
7. 调用代理对象



补充内容

**使用javassist 实现 类似Cglib动态代理**

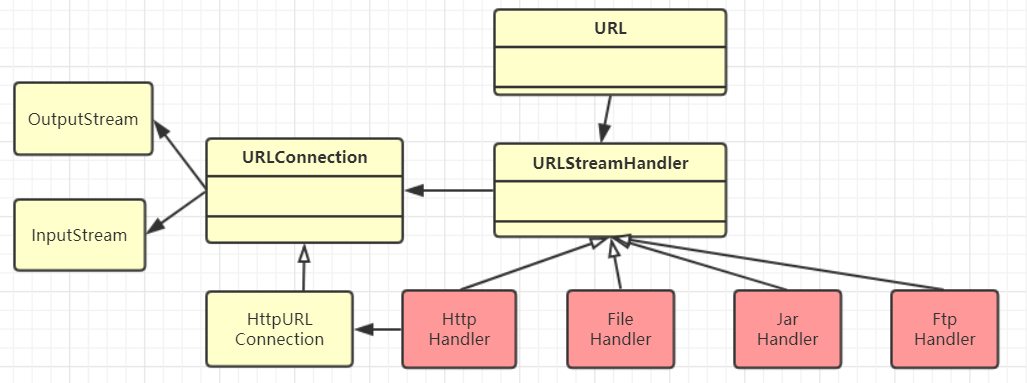
**// 静态代理**

## JAVA 应用协议代理 （FTP、Http、Https、File）

**提问：**

如果要知道一个系统所有URL协议（ http 、ftp、File） 协议的连接用时、写入性能、读取性能如何实现？

**分析一下URL的类结构：**



**URL执行过程：**

1. URL 基于protocol 构建对应 UrlStreamHandler
2. UrlStreamHandler.openConnection() 打开连接，返回URlConnection
3. URlConnection 设置连接属性
4. URlConnection 打开 outPutStream
5. URlConnection 打开 inPutStream
6. 关闭连接