1 什么是代理模式

- 为目标对象提供一个代理对象，以控制目标对象的访问（关键是控制访问！）
- 用户只能通过代理对象来访问目标对象

2 什么情况用代理模式

- **远程代理：** 为一个对象在远程地址提供一个代理，客户端通过访问这个代理对象来访问远程对象，代理对象隐藏了网络访问的细节
- **虚代理：** 控制目标对象创建的时机，如网页加载，图片加载比较耗时间，所以可以先加载一个虚图片(假的占位的图片)，再慢慢刷新成真实的图片
- **保护代理：** 控制对原始对象的访问。及不同用户对原始对象有不同的访问权限
- **智能引用：** 访问对象时，执行一些附加的操作。（这个和装饰者模式差不多）

3 为什么使用代理模式

- 主要是控制用户对原始对象的访问
- 使用代理模式可以使用户和真实对象解耦

4 远程代理示例

- Java RMI 提供了方便的远程代理框架
- Java RMI 远程方法调用

5 虚代理示例

- github 源码地址
- 虚拟代理模式-Virtual Proxy(Java实现)

6 保护代理示例

- github 源码地址

7 动态代理示例（智能引用）

- 动态代理的核心是：InvocationHandler 接口和 java.lang.reflect.Proxy类
- 通过实现 InvocationHandler 接口，来指定动态代理类的增强方法
- 通过 Proxy 类来运行时创建代理类
- github 源码地址
- 动态代理与静态代理区别
- JAVA学习篇--静态代理VS动态代理

8 静态代理示例

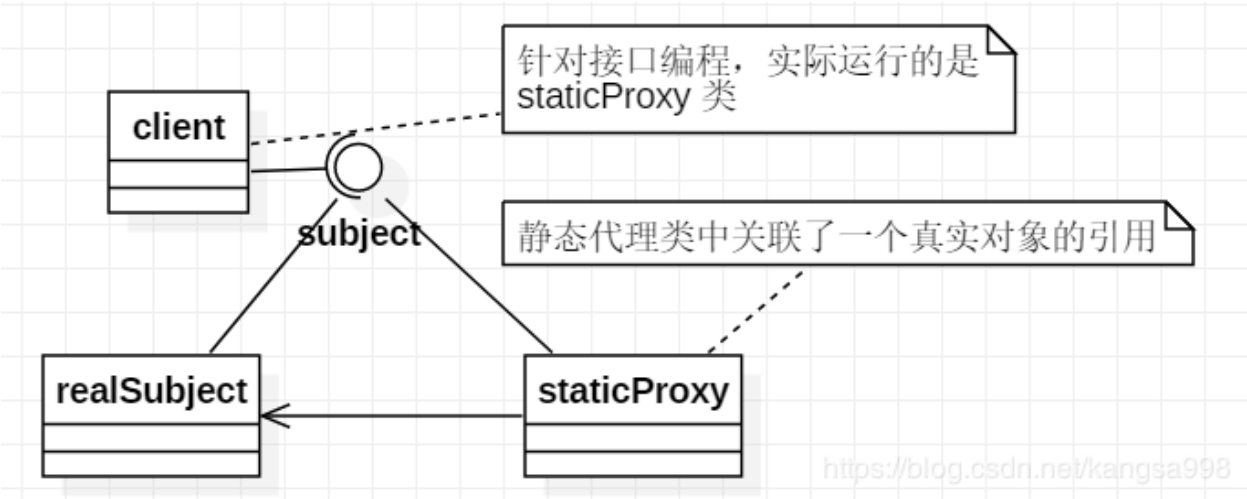
- 在程序运行前就已经存在代理类的字节码文件，代理类和委托类的关系在运行前就确定了
- 除了动态代理别的代理类型一般都是静态代理
- 当然 虚代理、保护代理等都可以由静态代理来实现！

9 缓存代理示例

- 缓存代理可以当作一种智能引用代理，它只是增加了缓存结果方法
- 代理模式（三）：远程代理，虚拟代理，缓冲代理

10 UML 类图

- 这是静态代理类UML图，动态代理在运行期间生成类，最后也就是静态代理的形式



参考

虚拟代理模式-Virtual Proxy(Java实现) 代理模式（三）：远程代理，虚拟代理，缓冲代理 动态代理与静态代理区别 JAVA学习篇--静态代理VS动态代理