实验6 结构型模式3

1 实验目的

学会用UML设计类图，熟练掌握代理模式。

2 实验环境

开发环境：JDK 8.0（或更高版本，高版本要下载独立JavaFX）
开发工具：Eclipse
设计工具：StarUML（或PlantUML）

3 基础知识

3.1 代理模式

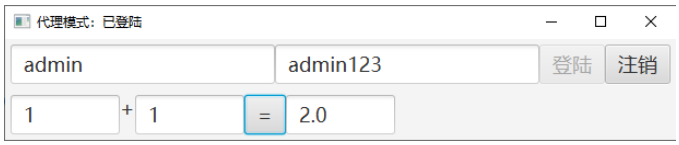
代理模式为其他对象提供一种代理，以控制对这个对象的访问。

4 实验内容

实验1 身份认证与操作日志

问题描述

> 开发一个界面小程序，通过动态代理模式实现身份认证和操作日志记录功能。程序基本功能包括（1）登陆功能。用户输入正确的用户名和密码后，程序将登陆状态修改为登陆状态，登陆状态下可以使用计算模块。（2）加法计算模块。加法计算模块实现两个数求和，将结果显示在界面。只有在登陆状态下才能使用。（3）操作日志功能。操作计算模块后，会将操作记录在文件中。（采用动态代理实现）



提示与解析：

1、根据题意，在操作加法功能之前需要登陆状态判断，在操作之后需要进行记录。采用代理方式实现，首先需要明确代理的接口以及对应的被代理对象：

```
public interface IAdd {
    public Double add(double x, double y);
}
public class Calculator implements IAdd{
    @Override
    public Double add(double x, double y) {
        return x + y;
    }
}
```

2、设计代理类的调用逻辑：

```
public class MyInvocationHandler implements InvocationHandler {
    public MyInvocationHandler(Object obj) {
        super();
        this.obj = obj;
    }
    Object obj;
    @Override
    public Object invoke(Object proxy, Method method, Object[] args) throws Throwable {
        Manager m = Manager.getInstance();
        Object o = null;
        if(m.isLogin()) {
            o = method.invoke(obj, args);
            // 写日志：成功调用
        }else {
            // 写日志：未调用成功
        }
        return o;
    }
}
```

3、动态生产代理对象：

```
IAdd add = new Calculator();
MyInvocationHandler mih = new MyInvocationHandler(add);
IAdd addProxy = (IAdd) Proxy.newProxyInstance(add.getClass().getClassLoader(), new Class[] {IAdd.c
```

4、用户登陆及登陆状态实现：

```
public class Manager {
    private static Manager instance = new Manager();
    private Map<String, String> user2password;
    private boolean login = false;
    private PrintWriter pw;

    private Manager() {
        user2password = new HashMap<String, String>();
        user2password.put("admin", "0192023A7BBD73250516F069DF18B500"); //admin123
        try {
            pw = new PrintWriter("log.txt");
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }
    public static Manager getInstance() {
        return instance;
    }
    public boolean login(String user, String pd) {
        if(user2password.containsKey(user)) {
            if(user2password.get(user).equals(md5(pd))) {
                login = true;
            }
        }
        return login;
    }

    public void logout() {
        login = false;
    }
}
```

采用一个单例`Manager`管理用户登陆，设计登陆和注销函数，采用`Map`保存用户列表，密码采用MD5编码，当用户信息能够匹配，则将登陆状态设置为`true`。

5、日志功能也通过单例`Manager`管理，设计`log()`函数记录操作。

具体要求：

1、调试样例程序，使其正常运行；

2、绘制代理模式相关的类图，说明实现过程和使用效果。

3、如果这里增加减法或其他运算功能，也需要登陆状态认证和操作记录，是否需要重新编写一

个`InvocationHandler`类？若不是，应该如何实现？

5 实验要求

5.2 实验评价

1、完成实验内容（60%）
2、对实验思路进行阐述（20%）
3、对实验过程进行总结（20%）

5.1 实验报告

1、根据要求完成实验内容、思路阐述和总结。
2、截取清晰的核心代码、设计图和效果到报告中；