**实验三 结构型设计模式**

学生姓名： 黄丹 学 号： 6130116093 专业班级：航天班

实验类型：□ 验证 □ 综合 □ 设计 □ 创新 实验日期： 2019/4/2 实验成绩：

一、实验项目名称

结构型设计模式

二、实验目的

1、了解回顾熟悉结构型设计模式，熟练使用相关设计模式；

2、了解类适配器设计模式及对象适配器模式的异同；

3、掌握模式结构，学习运用模式解决实际设计中遇到的问题；

4、熟练使用动态多态。

三、实验基本原理

四、主要仪器设备及耗材

PC微机

Windows 操作系统

IDEA/JDK1.8编程环境

GitHub

五、实验内容

1、使用Java语言实现一个双向适配器实例，使得猫可以学狗叫，狗可以学猫抓老鼠。

2、某日志记录器（Logger）既可以支持不同的操作系统，还可以支持多种编程语言，并且可以使用不同的输出方式。使用桥接模式设计该系统。

3、使用组合设计模式设计一个杀毒软件（AntiVirus）的框架，该软件既可以对某个文件夹（Folder）杀毒，也可以对某个指定的文件（File）进行杀毒，文件种类包括文本文件TextFile、图片文件ImageFile、视频文件VideoFile。

4、某图书管理系统中，书籍类（Book）具有借书方法borrowBook（）和还书方法returnBook（）。现需要动态给书籍对象添加冻结方法freeze（）和遗失方法lose（）。使用装饰模式设计该系统。

5、在计算机主机（Mainframe）中，只需要按下主机的开机按钮（on()），即可调用其他硬件设备和软件的启动方法，如内存（Memory）的自检（check()）、CPU的运行（run()）、硬盘（HardDisk）的读取（read()）、操作系统（OS）的载入（load()）等，如果某一过程发生错误则计算机启动失败。使用外观模式模拟该过程。

6、使用享元设计模式设计一个围棋软件，在系统中只存在一个白棋对象和一个黑棋对象，但是它们可以在棋盘的不同位置显示多次。要求使用简单工厂模式和单例模式实现享元工厂类的设计。

7、应用软件所提供的桌面快捷方式是快速启动应用程序的代理，桌面快捷方式一般使用一张小图片（Picture）来表示，通过调用快捷方式的run（）方法将调用应用软件（Application）的run（）方法。使用代理模式模拟该过程。

8、已知在一个绘图系统中有两个类，描述如下：

public class DrawCircle {

public void DrawCircle() { …… }

public void insert(String msg){

System.out.println("DrawCircle insert():"+msg);

}

……

}

public class DrawAngle {

public void DrawAngle() { …… }

public void insert(String msg){

System.out.println("DrawAngle insert():"+msg);

}

……

}

现在要在DrawCircle中调用DrawAngle的方法，可是类已经分发出去了，不能修改，如何用适配器模式解决这个问题，画出类关系图并说明采用的是哪种适配器模式？

9、假设存在一个Java定义的类Car，它的定义如下：

public final class Car

{

public void move()

{System.out.println("在陆地上移动！");}

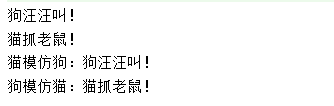
}

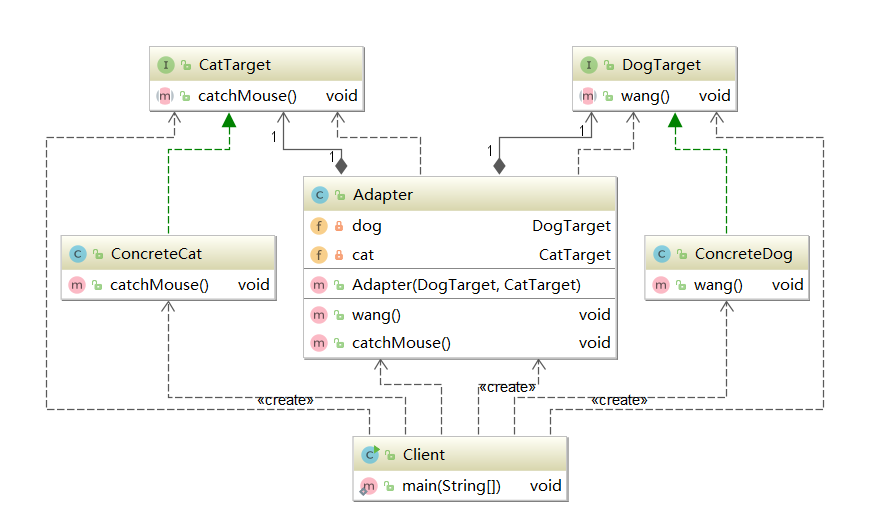
现在，有会飞的汽车，请给该汽车的对象动态添加会飞的功能。请采用合理的设计模式给出相关设计。

10、一杯咖啡，有中杯和大杯之分，同时还有加奶和不加奶之分。如果用单纯的继承，这四个具体实现(中杯大杯加奶不加奶)之间有概念重叠，因为有中杯加奶，也有中杯不加奶，如果再在中杯这一层再实现两个继承,很显然混乱，扩展性极差。请使用桥接模式进行设计。

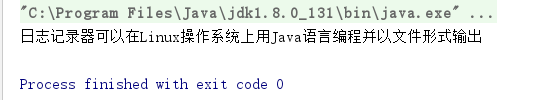
六、实验数据及处理结果

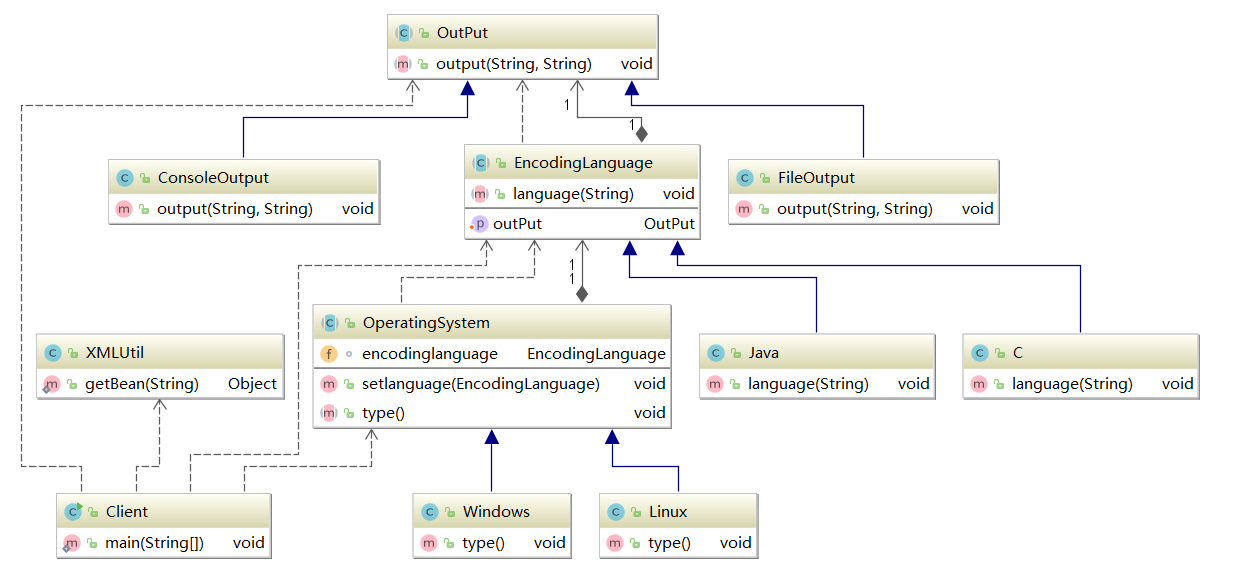
1.双向适配器实例





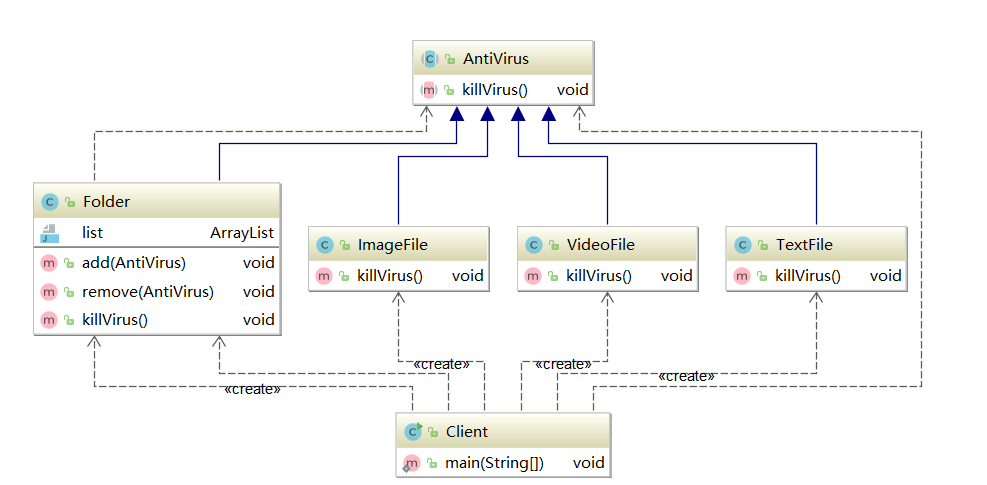
2.日志记录器





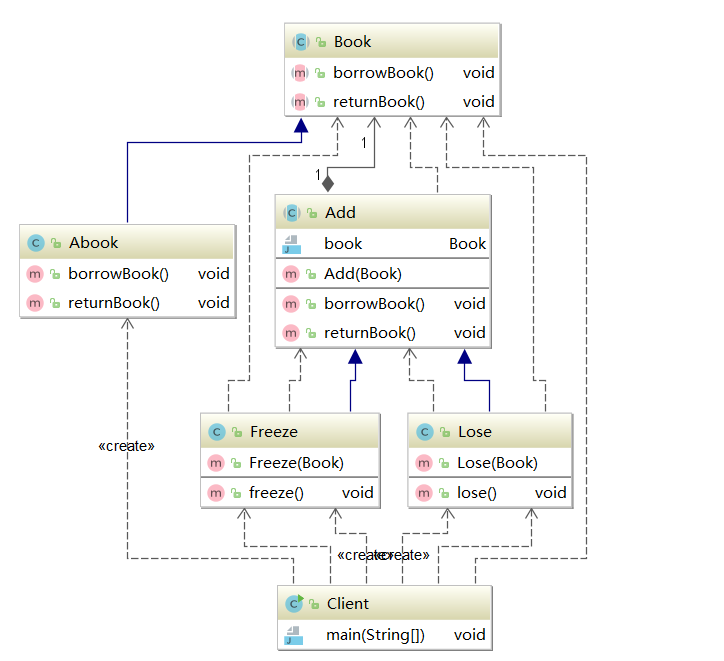
3.杀毒软件



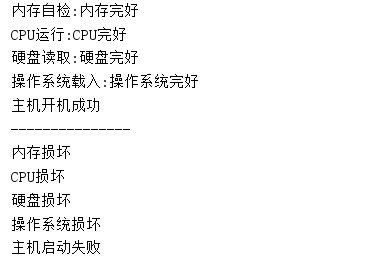


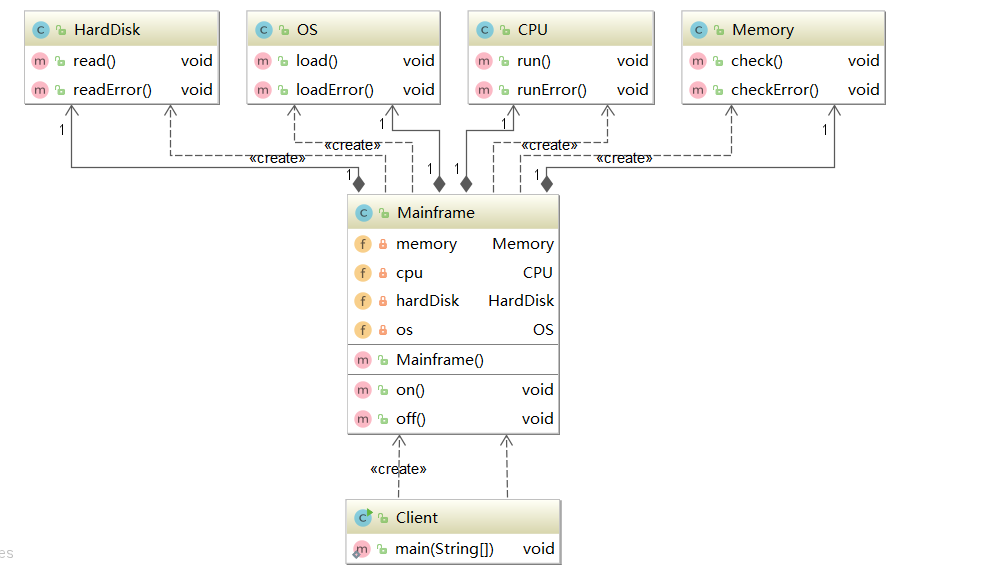
4.图书管理系统



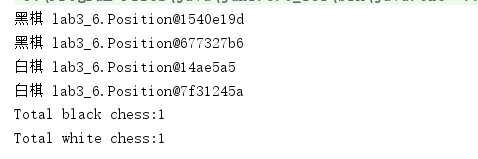


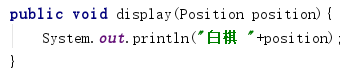
5.计算机主机



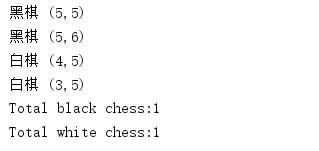


6.围棋软件



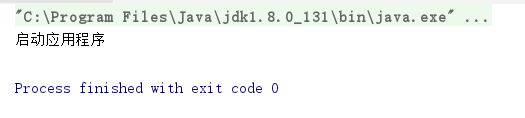


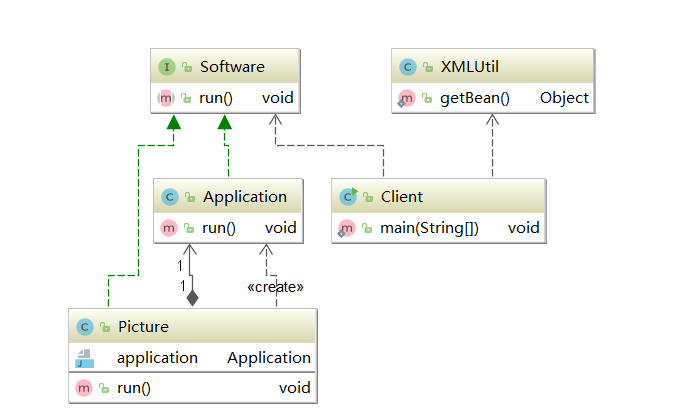
调用输出了对象，应调用对象的方法





7.桌面快捷方式

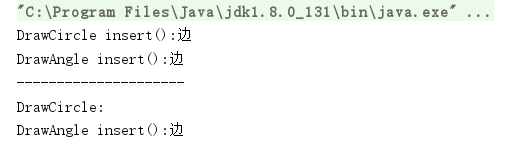


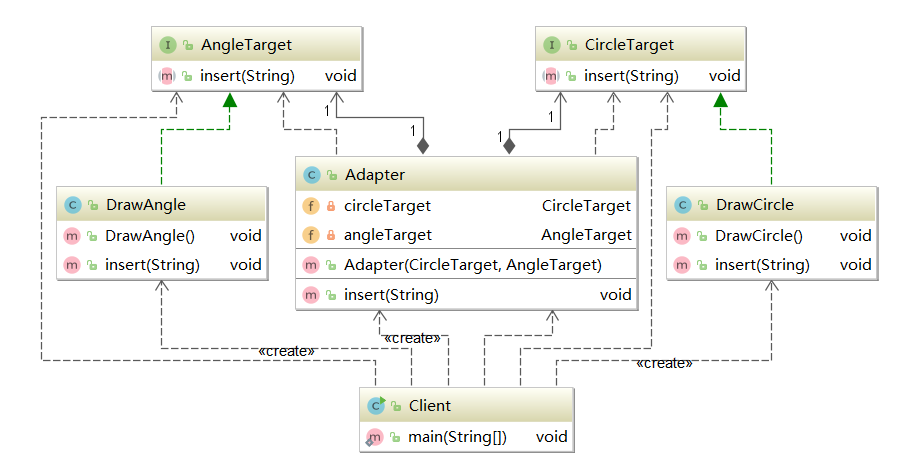


8.绘图系统

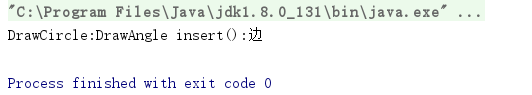
使用两种适配器模式尝试：第二种方式实现比较容易

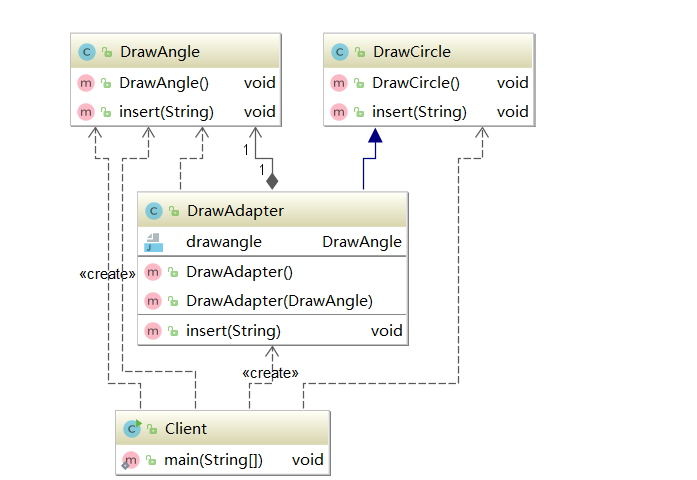
第一种：双向适配器模式



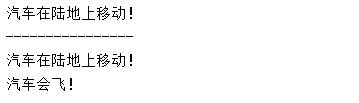


第二种：对象适配器模式

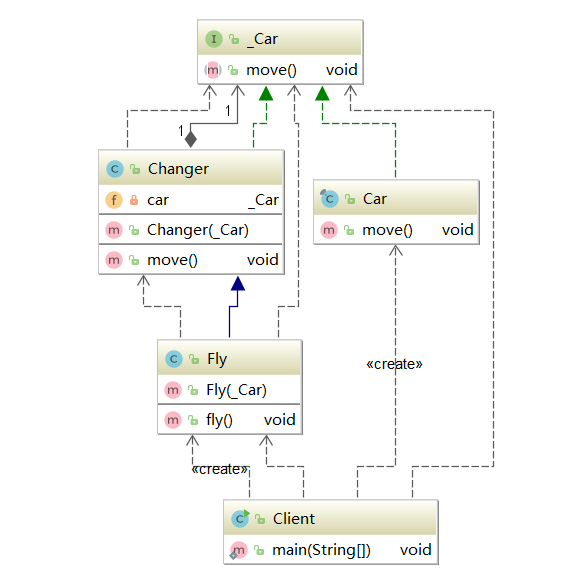




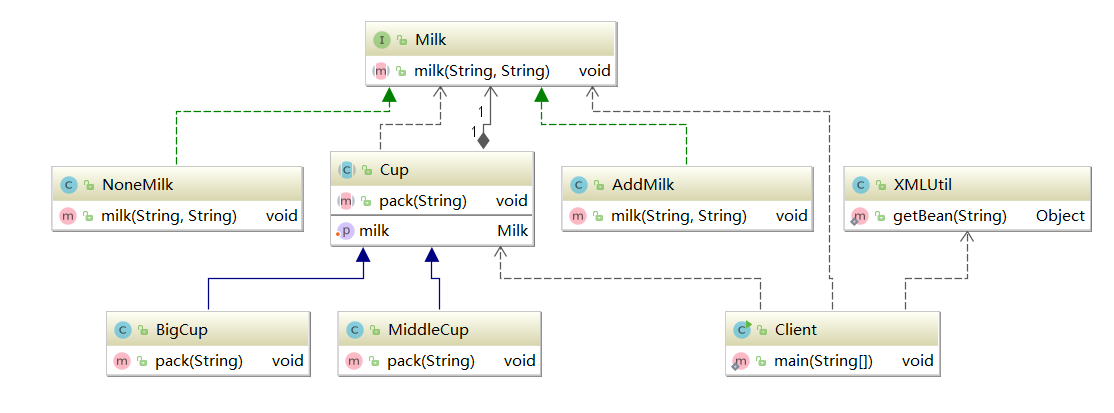
9.类Car



给汽车添加会飞的功能，不可以用适配器模式，因为汽车类为final类型，无法被继承，因此选择装饰模式。新建汽车类的接口，给它增加新的会飞的功能。



10.咖啡



七、思考讨论题

1、比较结构型设计模式与创建型设计模式；

答：创建型设计模式主要用于创建对象，结构型模式主要用于处理类或对象的组合。

2、比较类结构设计模式与对象结构模式的异同；

答：类模式处理类和子类之间的关系，这些关系通过继承建立，在编译时刻就被确定下来，是属于静态的，在类结构模式中一般只存在继承关系和实现关系；对象模式处理对象之间的关系，这些关系在运行时刻变化，更具动态性，通过关联关系使得在一个类中定义另一个类的实例对象，然后通过该对象调用其方法。

3、了解符合面向对象的设计原则。

答：单一职责原则，开闭原则，里氏代换原则，依赖倒转原则，接口隔离原则，合成复用原则，迪米特原则。

八、参考资料

九、实验代码

<https://github.com/inocentnine/JAVA-StructuralPatterns>