实验报告

|  |  |
| --- | --- |
| 一、实验内容  一个软件团队开发绘图系统，设计了圆对象(Circle)、矩形对象（Rectangle),线对象(Line)都支持（Draw)函数，即可以通过（Draw)函数绘制图形。为了加快项目进度，将角度对象(Angle)绘制功能交给了合作团队实现。但合作团队将角度对象绘制函数定为了（DrawAngle)。绘图系统提供给用户后，用户不满意，希望能统一调用，不用记太多命令。现在使用适配器模式来实现接口统一，请分别设计对应的类适配器和对象适配器模式，画出各自类图并给出实验程序。 | |
| 二、实验过程记录 | |
| 类图 | 1. 类适配器模式      1. 对象适配器模式 |
| 实验过程记 录 | 因为本例的适配器类和适配者类是一对一的适配关系，因此对象适配器模式和类适配器模式都可以使用。在这两种方式的设计中，我都定义了目标抽象类接口ShapePainter，它为用户类Client提供了draw()方法，而Rectangle、Line和Circle类，包括适配器类都继承自它，而它们各自有不同的实现，在Client类调用方法时，可以创建对象的实例，并都用draw()实现画图，不用记太多命令。这样就达成了用户对于实现接口统一的需求。而对象适配器和类适配器在设计中有所不同的是，前者的适配器类ShapeAdapter和适配者类Angle是关联关系，适配器类中包装了适配者的一个实例；后者适配器类AngleAdapter实现了目标接口ShapePainter，并继承了适配者类Angle。设计思路主要就是以上这些，详细代码见后文截图。 |
| 总结 | 适配器模式的主要目标是让那些由于接口不兼容而不能交互的类可以一起工作，主要任务是对代码的组织，而不是功能实现。有时在没有源码，不能修改原来的抽象目标类和适配者接口时，通过使用该模式可以有效统一对象访问接口。通过这个例子，我熟悉掌握了类适配器模式和对象适配器模式的类图设计和代码实现，并且更加深刻理解了对象适配器作为实际开发中使用频率更高的模式，拥有更多的灵活性。我在对象适配器模式中还用到了XML配置文件和反射机制，创建适配器类的实例，在需要用到新的适配器类时可以通过更改XML文件实现，更提高了程序的可扩展性。 |