文章编号：2096-1472(2017)-07-15-03

软件设计模式在Java程序设计课程教学中的应用研究

张 璞，夏 英
(重庆邮电大学计算机科学与技术学院，重庆 400065)

摘 要：Java程序设计课程学习过程中，学生面临面向对象编程思维方式转换困难，难以灵活应用面向对象特性等问题。针对这些问题，提出将软件设计模式有机融入于课程教学的教学模式。阐述了教学模式的教学过程设计、在面向对象概念教学过程及Java SE类库教学过程中软件设计模式的应用，最后，对上述内容进行了总结。

关键词：Java程序设计；软件设计模式；面向对象；教学模式
中图分类号：TP311 **文献标识码**：A

Application Research of Software Design Pattern in Java Programming Course Teaching

ZHANG Pu,XIA Ying
(*Department of Computer Science and Technology,Chongqing University of Posts and Telecommunications,Chongqing* 400065,*China*)

Abstract: In the learning process of Java programming course,students face the difficulty of converting to object oriented programming thinking mode,and it is also difficult for students to flexibly apply object-oriented characteristics. In view of these problems,the paper puts forward that the software design pattern should be organically integrated into the course teaching mode.It expounds the teaching process design of the teaching mode,and the application of the software design pattern in the teaching process of the object-oriented concept and Java SE class library.Finally,the above contents are summarized.

Keywords: Java programming;software design pattern;object-oriented;teaching mode

1 引言(Introduction)

作为一种面向对象程序设计语言，Java由于具有面向对象、平台无关、安全性、内置多线程等众多优良特性[1]，已被广泛应用于Web开发、智能手机、桌面应用等不同领域，是当前最流行的编程语言之一。作为程序设计教学的一个重要分支，国内外众多高校均开设了Java程序设计课程，该课程已成为一门重要的专业基础课。

Java程序设计课程教学过程中，需要将面向对象编程思维的培养作为重点，并贯穿于教学过程始终，使学生能够灵活运用面向对象思想来解决实际问题。从实际情况来看，许多高校将C语言开设为第一门程序设计语言课程，因此，学习Java语言时，学生已经具备C语言基础。由于受面向过程编程思想的影响，很多学生仍然存在由面向过程转换为面向对象编程思维的困难。例如，在进行类的设计时，一些学生习惯于将许多静态方法定义于一个类中，甚至觉得类只是在C语言函数的基础上加个框而已。一些学生则陷于Java编程语法的记忆与机械理解之中，未能掌握面向对象编程语言的基本特征及概念，不熟悉使用类进行面向对象设计的基本原则，导致遇到具体问题时，不能灵活应用继承、多态等面向对象特性，所实现代码的可复用性、可扩展性差。

软件设计模式是从许多优秀软件系统中总结出的可复用设计方案[2,3]。文献[2]最先将设计模式的概念引入软件开发领域，经过分类编目后，归纳总结出了23种设计模式。已有研究表明[4,5]，设计模式思想在面向对象程序设计教学中有重要的作用，把设计模式引入教学过程中，能使学生更加深刻地理解面向对象思想，了解面向对象设计的基本原则，提高编程能力，有助于开发更易维护、可扩展性强、复用性好的系统。

针对学生转换面向对象编程思维困难，难以灵活应用面向对象特性等问题，通过教学实践，笔者在不同教学模块中有针对性、有目的性地引入一些经典设计模式，在教学过程中有机融入设计模式，取得了良好效果。下面对软件设计模式在Java课程教学中的应用进行介绍。

2 教学过程设计(Design of the teaching process)

面向对象程序设计课程的教学核心不是学习设计模式[4]，因此，在教学过程中，不需要学习全部设计模式，而需要紧密结合教学内容，挑选引入一部分设计模式来深化学生对面向对象编程思想的理解和灵活运用。表1给出了笔者在课程教学活动中所引入的设计模式。

基金项目：重庆邮电大学校级教育教学改革项目(XJG1309)：以“案例驱动、项目导向”为核心的Java语言程序设计课程教学模式探索与实践；重庆市教育教学改革项目(153069).

表1 Java 教学过程中的设计模式引入

Tab.1 Introduction of design patterns in Java teaching process

教学模块	所引入设计模式	引入目的
类和对象	单例模式	帮助掌握public、private、static、构造方法的灵活运用
类的继承	模板方法模式	帮助掌握继承、多态等面向对象语言基本特征
接口	策略模式	帮助理解面向对象设计中的开放-闭合等基本原则
集合框架	迭代器模式	帮助理解Java集合框架类库的设计思想
输入输出流	装饰模式	帮助理解java.io类库的设计思想
图形用户界面	组合模式、策略模式、观察者模式、装饰模式	帮助理解java.awt及javax.swing类库中的组件、布局管理器、事件处理机制等的设计思想
多线程	单例模式	通过线程安全的单例模式实现，帮助理解线程同步控制

表1中的“类和对象”“类的继承”“接口”等教学模块属于Java中面向对象部分的教学内容，而“集合框架”“输入输出流”“图形用户界面”“多线程”等教学模块则属于Java SE类库部分的教学内容。本文后续内容将分别介绍设计模式在以上两部分教学内容中的应用。

在各个教学模块的教学实施过程中，为了遵循学生的认知规律，激发学生的学习兴趣，通过精心设计教学案例，笔者采用了“案例驱动”的设计模式教学过程，如图1所示。

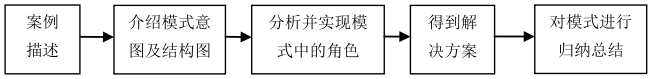

图1 “案例驱动”的设计模式教学过程

Fig.1 "Case-Driven" teaching process for design pattern

由教师先描述案例，介绍所需要解决的问题，让学生带着问题进入内容的学习，从而引起学生的好奇心。之后由教师对解决案例所用设计模式的意图及模式结构图进行介绍，根据模式结构图的UML图形描述来介绍模式结构中的各种角色，再从案例中分析得出各种角色所对应的类或接口，并用Java语言进行实现，得到案例的解决方案。最后，由教师对设计模式的效果、其中体现的面向对象思想及设计原则等进行归纳总结。

3 面向对象教学内容中引入设计模式(Introducing design pattern into object-oriented teaching content)

3.1 通过设计模式来深化基本概念的理解

面向对象的核心思是将数据和对数据的操作封装在一起形成对象，并通过抽象找出同一类对象的共同状态和行为，从而得到类。Java程序的基本单位是类，编写程序时要先定义好类，再由类实例化生成对象。“类和对象”是Java中的重要教学内容。在教学过程中，学生会接触到许多新概念，例如类、对象、构造方法、类及成员的访问权限、静态变量、实例变量、静态方法、实例方法等。在学习了这些基本概念的基础上，笔者在教学过程中引入单例设计模式来加深学生对以上基本概念的理解。

教学过程中，首先让学生了解在实际编程中，有些对象仅需要一个，例如操作系统中的打印池对象。继而提出问题，如何保证设计的类在仅有一个实例对象。

案例设计：操作系统中，打印池(Print Spooler)是一个用于管理打印任务的应用，在系统中只允许运行一个打印池对象，要求使用单例模式来模拟打印池。

描述了案例后，再由教师介绍单例模式的意图和模式结构图，从案例入手，分析出单例角色类(PrintSpooler)，给出案例的模式实现代码：

```
class PrintSpooler {
    private static PrintSpooler instance=null; //私有静态成员变量
    private PrintSpooler(){    }    //私有构造方法
    public static PrintSpooler getInstance(){ //公有的静态方法
            if(instance==null) {
                instance=new PrintSpooler();
            }
            return instance;
    }
    public void print(){//实例方法
            System.out.println("执行打印任务");
    }
}
```

讲解了模式实现代码后，再由教师启发学生思考并回答该模式的几个要点：单例类为何要定义私有构造方法？为何要定义私有的静态成员变量和公有静态方法来保存和获得对象实例？单例模式是如何保证对象实例的唯一性的？最后再总结单例模式的适用情况。通过将单例模式与Java面向对象语法相结合，逐步深入讲解来加深学生对面向对象概念的理解。

3.2 通过设计模式来深化面向对象语言基本特征的理解

在学习面向对象语言的基本特征时，学生普遍感觉多态的概念比较抽象，难以理解。因此，笔者将模板方法模式引入到“类的继承”教学模块中，设计了模板方法模式来模拟银行业务办理流程的案例。

案例设计：客户在银行办理业务时，一般都包含取号排队、办理具体业务、对工作人员进行评分等步骤。无论具体业务是取款、存款还是转账，其基本流程都是固定的。要求使用模板方法模式来模拟银行业务的办理流程。

叙述案例后，由教师介绍模板方法模式的意图：定义一个操作中算法的骨架，而将一些步骤延迟到子类中，从而使子类可以不改变一个算法的结构即可重定义该算法的某些特定步骤。在此基础上，给出模式的UML结构图来介绍模式中所包含的两个角色：抽象类(AbstractClass)和具体子类(ConcreteClass)，然后从案例中引导学生根据模式结构图来定义出案例中的抽象类和几个具体子类，进而给出案例的模板方法模式实现代码：

```
abstract class BankProcess{  //抽象类
    public void takeNumber() {
            System.out.println("取号排队");
}
     public abstract void transact();
     public void evaluate()     {
System.out.println("反馈评分");
     }
    public void process()     {  //模板方法
         this.takeNumber();
```

```
            this.transact();
            this.evaluate();
        }
    }
    class Deposit extends BankProcess {   //具体子类：存款类
        public void transact()    {
                System.out.println("存款");

        }
    }
    … …
    class Transfer extends BankProcess {   //具体子类：转账类
        public void transact()    {
                System.out.println("转账");

        }
    }
```

进行代码讲解后，由教师有意识地引导学生画出上述实现代码的UML类图，如图2所示，加深学生对程序结构和模式结构的理解。

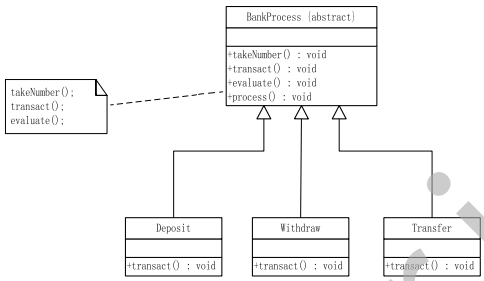

图2 银行业务办理流程案例的模板方法模式结构图

Fig.2 Template method pattern diagram of bank business process case

再由教师引导学生思考继承及多态性等特征在该模式中的体现，使学生认识到在模板方法模式中，子类可以体现多态性。即子类可以根据各自的需要覆盖父类中定义的方法，由于面向对象的多态性，子类对象执行从父类继承来的模板方法时，其中的方法调用将通过动态绑定方式来实现子类对父类行为的反向控制。最后，由教师总结归纳模板方法模式的适用情况。

3.3　通过设计模式来深化面向对象设计原则的理解

Java语言的学习过程中，掌握好面向对象基本语法和概念后，了解一下用类进行面向对象设计的基本原则，如“开放—闭合”原则、聚合复用原则、依赖倒转原则等，不仅有助于加深学生对面向对象编程思想的理解，还有助于编写出易维护、易扩展和易复用的程序代码。

在学习Java中“接口”部分教学模块时，笔者引入了策略模式来帮助学生理解“开放—闭合”等设计原则。

案例设计：在多个裁判负责打分的比赛中，每位裁判给选手一个评分，选手的最后得分的评分方案(策略)可以有多种。如可以是所有评分的平均值，也可以是去掉一个最高分和一个最低分后的平均值。要求使用策略模式来从多种评分方案中选择比赛的评分方案。

和前述教学过程类似，通过对案例进行描述及分析后，使学生了解清楚策略模式的意图是定义及封装一系列算法，并让它们可以相互替换。然后再根据策略模式结构图中的上下文(Context)、策略(Strategy)和具体策略(ConcreteStrategy)等不同角色找出案例中的对应类，进行代码实现，并引导学生画出如图3所示的模式结构图。

```
    interface Strategy {   //策略接口
        public double computeAverage(double [] a);
    }
    class  AverageScore{   //上下文类
        Strategy strategy;
        public void setStrategy(Strategy strategy){
            this.strategy=strategy;
        }
        public double getAverage (double [] a){
            if(strategy!=null)
              return strategy.computeAverage(a);
        }
    }
    class StrategyA implements Strategy{   //具体策略类
        public double computeAverage(double [] a){
            double score=0,sum=0;
            for(int i=0;i<a.length;i++){
                sum=sum+a[i];
            }
            score=sum/a.length;
            return score;
        }
    }
    class StrategyB implements Strategy{   //具体策略类
        public double computeAverage(double [] a){
            if(a.length<=2)  return 0;
            double score=0,sum=0;
            Arrays.sort(a);    //排序数组
            for(int i=1;i<a.length-1;i++){
                sum=sum+a[i];
            }
            score=sum/(a.length-2);
            return score;
        }
    }
```

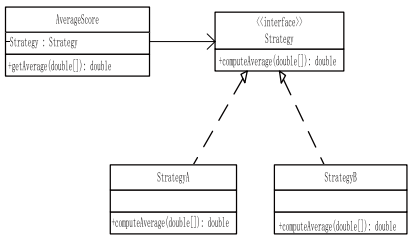

图3 评分方案案例的策略模式结构图

Fig.3 Strategy pattern diagram of the scoring scenario case

接下来，由教师对模式中所涉及到的“开放—闭合”原则进行分析。上下文类AverageScore只知道它要使用某一个

实现Strategy接口类的实例,但不需要知道具体是哪一个类,因此,当增加新的具体策略时,不需要修改上下文类的代码,上下文就可以引用新的具体策略的实例。最后,教师对策略模式的主要优点及适用情况进行总结。使学生掌握策略模式通过对“开放—闭合”原则的完美支持,在不修改原有系统的基础上可以更换算法或者增加新的算法,从而能很好地管理多种评分方案,提高代码可扩展性。

4 J2SE类库教学内容中引入设计模式(Introducing design pattern into J2SE class library teaching content)

Java SE平台的类库设计中也应用了大量的设计模式,因此,在API类库教学过程中,介绍所涉及的设计模式,一方面可以使学生更好地掌握类库中相关类和接口的作用及功能;另一方面,也能使学生对面向对象思想及基本原则、设计模式的实际运用有更深的体会。因此,笔者在进行集合框架、输入输出、图形用户界面编程、多线程等内容的教学过程中,对J2SE类库中所涉及到的典型模式也作了相应介绍。

在学习Java集合框架时,笔者引入了迭代器模式,通过介绍该模式来帮助学生更好地掌握集合框架的设计思想。JDK类库中,Collection接口的iterator()方法返回一个Iterator类型的对象,java.util.Iterator接口就是迭代器模式的应用。通过迭代器模式访问一个列表(List)或者一个集合(Set)对象的内容时,无需了解聚合对象的内部表示,使遍历操作变得简单。

在学习输入输出流时,Java中的I/O类库由于其庞杂的特点,一直是学生学习的难点。为此,笔者在介绍I/O类库的过程中,引入了装饰模式来帮助学生掌握I/O类库结构。在I/O处理中,Java将数据抽象为流(Stream),输入流(Input Stream)、输出流(OutputStream)类只提供最基本的数据读写功能。而FilterInputStream作为抽象装饰类,BufferedInputStream、DataInputStream、BufferedOutputStream、DataOutputStream等类则作为具体装饰类,这些类的设计很好地应用了装饰模式,以透明的方式动态地给一个对象附加上更多的功能,这样就可以在不需要创造更多子类的情况下,将对象的功能加以扩展。

在学习图形用户界面时,随着教学内容的展开,笔者分别引入了组合模式、装饰模式、策略模式、观察者模式来介绍java.awt和javax.swing包中的类和接口。例如,介绍AWT/Swing库中的组件类(Component)、容器类(Container)及系列子类的关系时,通过组合模式的介绍,学生很快地掌握了类库中的大量组件类、容器类之间的关系。介绍容器的布局管理方式时,笔者有意识地启发学生思考Java类库中Container类、LayoutManager接口间的关系实际上就是策略模式的一个经典应用。介绍事件处理时,为了帮助学生深刻理解事件处理机制,笔者相应介绍了观察者模式。在javax.swing包中,通过装饰模式能动态地给一些组件增加新的行为或改善其外观显示。例如,JList组件本身并不支持滚动条功能,要创建可以滚动的列表,则需要使用JScrollPane类来作为装饰器。

在学习多线程时,针对“类和对象”教学模块中的单例模式实现代码,由教师分析在多线程环境下所可能产生的线程安全问题。可以假设这样一种情况:当第一个线程判断引用变量为空之后,对象实例化过程尚未完成之前,第二个线程开始判断引用变量是否为空。由于第一个线程尚未完成对象创建,因而引用变量instance还为空,这将导致第二个线程实例化第二个单例对象,从而导致不正常情况的发生。解决方法是通过线程同步控制机制,在getInstance()方法前加上synchronized关键字,通过对象锁来实现线程安全的单例模式实现。

5 结论(Conclusion)

教学实践证明,在Java程序设计课程教学过程中有机地引入设计模式的这种教学模式,将设计模式与面向对象的语法、概念、设计原则相结合,对培养学生的面向对象编程思维方式,提高编程能力等方面均有积极作用。这种教学模式也得到了学生良好的评价,学生的学习主动性、灵活运用面向对象特性解决问题的能力均得到了提高,为后续课程的学习及实践应用奠定了良好基础。

参考文献(References)

[1] 耿祥义,张跃平.Java面向对象程序设计(第2版)[M].北京:清华大学出版社,2013.

[2] Erich G,et al.李英军,等,译.设计模式:可复用面向对象软件的基础[M].北京:机械工业出版社,2004.

[3] 刘伟.设计模式实训教程[M].北京:清华大学出版社,2012.

[4] 杨瑞龙,朱征宇,朱庆生.引入软件设计模式的面向对象程序设计教学方法[J].计算机教育,2012(10):97-100.

[5] 章品正,於文雪.设计模式在C++课程教学中的运用[J].计算机教育,2014,218(14):41-45.

作者简介:

张 璞(1977-),男,博士,副教授.研究领域:数据挖掘,自然语言处理.

夏 英(1972-),女,博士,教授.研究领域:数据库与数据挖掘,时空大数据分析.

(上接第30页)

54(2):14-33.

[2] 陈澎,熊耀华,周慧.基于CDIO模式的软件工程实践教学课程建设的研究[J].软件工程,2016(1):1-3.

[3] O.M.Zamyatina,et al.Analysis of Engineering Invention Competencies in Standards and Programmes of Engineering Universities[J].Procedia-Social and Behavioral Sciences,2015,171:1088-1096.

[4] Edstr m,et al.PBL and CDIO: Complementary Models for Engineering Education Development[J].European Journal of Engineering Education,2014,39(5):539-555.

[5] 康雁,李彤.基于SE-CDIO培养学生项目管理能力的新途径[J].计算机教育,2013(13):69-72.

作者简介:

康 雁(1972-),女,博士,副教授.研究领域:软件开发,数据挖掘.

李 彤(1963-),男,博士,教授.研究领域:软件工程,软件过程.

张 璇(1978-),女,硕士,副教授.研究领域:软件工程,信息安全.