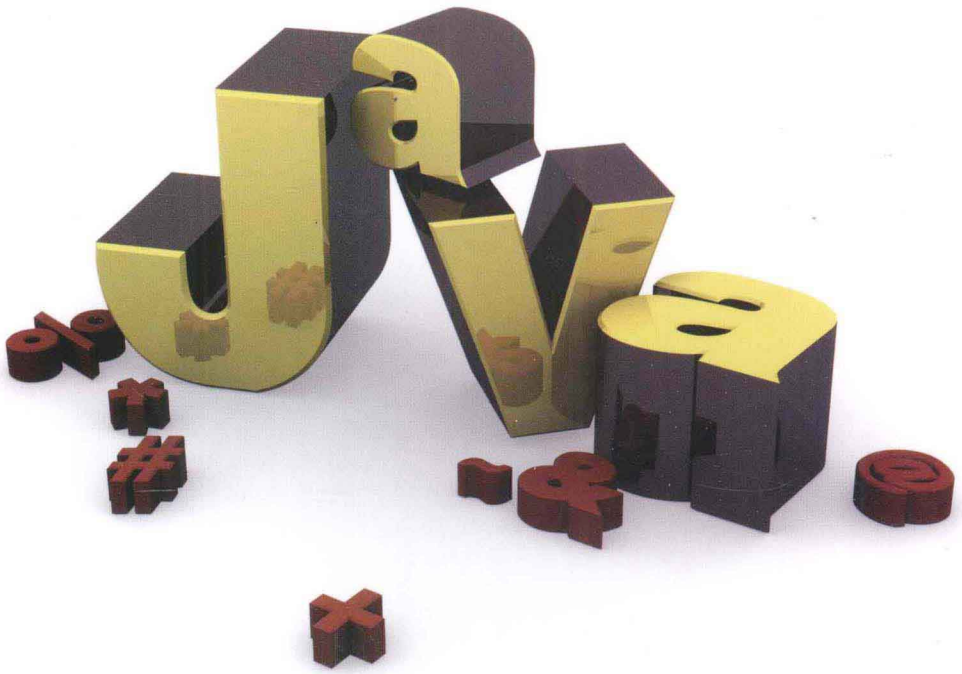
高等学校Java课程系列教材

面向对象与设计模式

耿祥义 张跃平 著



清华大学出版社

高等学校 Java 课程系列教材

面向对象与设计模式

耿祥义　张跃平　著

清华大学出版社
北京

内 容 简 介

本书是面向有一定 Java 语言基础和一定编程经验的读者，重点介绍了面向对象的核心内容以及作者在面向对象研究中的一些新思想，全面探讨在 Java 程序设计中怎样使用一些重要的设计模式。作者编写本书的目的是让读者不仅学习怎样在软件设计中使用好设计模式，更重要的是让读者通过学习使用设计模式深刻地理解面向对象的设计思想，以便更好地使用面向对象语言解决设计中的诸多问题。

本书可以作为计算机相关专业研究生或高年级学生的教材，也可以作为软件项目管理人员、软件开发工程师等专业人员的参考用书。可登录 www.tup.com.cn 下载书中的示例代码。

图书在版编目（CIP）数据

面向对象与设计模式 / 耿祥义，张跃平著. —北京：清华大学出版社，2013.6
高等学校 Java 课程系列教材
ISBN 978-7-302-30823-2

Ⅰ. ①面…　Ⅱ. ①耿… ②张…　Ⅲ. ①JAVA 语言-程序设计　Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2012）第 287864 号

责任编辑：魏江江　王冰飞
封面设计：杨　兮
责任校对：李建庄
责任印制：宋　林

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：北京鑫海金澳胶印有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：18.25　　**字　　数**：457 千字
版　　次：2013 年 6 月第 1 版　　**印　　次**：2013 年 6 月第1 次印刷
印　　数：1～3000
定　　价：29.50 元

产品编号：050593-01

前 言

目前，面向对象程序设计已经成为软件设计开发领域的主流，而学习使用设计模式无疑非常有助于软件开发人员使用面向对象语言开发出易维护、易扩展、易复用的代码，其原因是设计模式是从许多优秀的软件系统中总结出的成功的可复用的设计方案，已被成功应用于许多系统的设计中。本书是面向有一定 Java 语言基础和一定编程经验的读者，重点介绍了面向对象的核心内容，全面探讨在程序设计中怎样使用一些重要的设计模式。作者编写本书的目的是让读者不仅学习怎样在软件设计过程中使用好设计模式，更重要的是让读者通过学习使用设计模式深刻地理解面向对象的设计思想，以便更好地使用面向对象语言解决设计中的诸多问题。

本书共 26 章，前 6 章重点介绍了面向对象的核心内容，以及作者在面向对象研究中的一些新思想；第 7 章～第 26 章探讨在程序设计中怎样使用一些重要的设计模式。为了说明一个模式的核心实质，本书精心研究了针对每个模式的示例，以便让读者结合这样的示例更好地理解和使用模式。本书的全部示例由作者编写完成，本书示例代码及相关内容仅供学习设计模式使用。希望本书能对读者学习和使用设计模式有所帮助，并请读者批评指正。

作 者

2013 年 4 月

目 录

第1章 面向对象入门

本书是面向有一定 Java 语言基础和一定编程经验的读者，因此读者在学习本书之前应当有一定的 Java 语言基础，熟悉一种 Java 开发工具（如 Java SE 提供的 JDK）。

1.1 编程语言的几个发展阶段

1.1.1 面向机器语言

每种计算机都有自己独特的机器指令，比如，某种型号的计算机用 8 位二进制信息 10001010 表示加法指令，用 00010011 表示减法指令，等等。这些指令的执行由计算机的线路来保证，计算机在设计之初，事先就要确定好每一条指令对应的线路逻辑操作。计算机处理信息的早期语言是所谓的机器语言，使用机器语言进行程序设计需要面向机器来编写代码，即需要针对不同的机器编写诸如 01011100 这样的指令序列。用机器语言进行程序设计是一项累人的工作，代码难以阅读和理解，一个简单的任务往往蕴含着编写大量的代码，而且同样的任务需要针对不同型号的计算机分别进行编写指令，因为一种型号的计算机用 10001010 表示加法指令，而另一种型号的计算机可能用 11110000 表示加法指令。因此，使用机器语言编程也称为面向机器编程。20 世纪 50 年代出现了汇编语言，在编写指令时，用一些简单的容易记忆的符号代替二进制指令，但汇编语言仍是面向机器语言，需针对不同的机器编写不同的代码。习惯上称机器语言、汇编语言是低级语言。

1.1.2 面向过程语言

随着计算机硬件功能的提高，在 20 世纪 60 年代出现了过程设计语言，如 C 语言、FORTRAN 语言等。用这些语言编程也称为面向过程编程，语言把代码组成叫做过程或函数的块。每个块的目标是完成某个任务，例如，一个 C 的源程序就是由若干个书写形式互相独立的函数组成。使用这些语言编写代码指令时，不必再去考虑机器指令的细节，只需按照具体语言的语法要求去编写“源文件”。所谓源文件，就是按照编程语言的语法编写具有一定扩展名的文本文件，比如，C 语言编写的源文件的扩展名是 c，FORTRAN 语言编写的源文件的扩展名是 for 等。过程语言的源文件的一个特点是更接近人的“自然语言”，比如，C 语言源程序中的一个函数：

```
int max(int a,int b) {
   if(a>b)
     return a;
   else
```

```
    return b;
}
```


该函数负责计算两个整数的最大值。过程语言的语法更接近人们的自然语言，人们只需按照自己的意图编写各个函数，习惯上称过程语言为高级语言。

随着软件规模的扩大，过程语言在解决实际问题时逐渐显露出力不从心。对于许多应用型问题，人们希望编写出易维护、易扩展和易复用的程序代码，而使用过程语言很难做到这一点。面向过程语言的核心是编写解决某个问题的代码块，比如C语言中的函数，代码块是程序执行时产生的一种行为，但是面向过程语言却没有为这种行为指定“主体”，即在程序运行期间，无法说明到底是“谁”具有这个行为，并负责执行了这个行为。比如，C语言编写了一个“刹车”函数，却无法指定是哪个“实体”具有这样的行为。也就是说，面向过程语言缺少了一个概念，那就是“对象”。现实生活中，“行为”往往归结为某个具体的“主体”所拥有，即某个对象所拥有，并且该对象负责产生这样的行为。和面向过程语言不同的是，在面向对象语言中，核心的内容就是“对象”，一切围绕着对象，比如，编写一个“刹车”方法（面向过程称之为函数），那么一定会指定该方法的“主体”，比如，某个汽车拥有这样的“刹车”方法，该汽车负责执行“刹车”方法产生相应的行为。

1.1.3 面向对象语言

随着计算机硬件设备功能的进一步提高，使得基于对象的编程成为可能（面向对象语言编写的程序需要消耗更多的内存，需要更快的CPU保证其运行速度）。基于对象的编程更加符合人的思维模式，使用面向对象语言可以编写易维护、易扩展和易复用的程序代码，更重要的是，面向对象编程鼓励创造性的程序设计。

面向对象编程主要体现下列3个特性。

1．封装性

面向对象编程的核心思想之一就是将数据和对数据的操作封装在一起。通过抽象，即从具体的实例中抽取共同的性质形成一般的概念，比如，类的概念。在实际生活中，人们每时每刻都与具体的实物在打交道，例如，我们用的钢笔、骑的自行车、乘的公共汽车等。人们经常见到的卡车、公共汽车、轿车等都会涉及以下几个重要的属性：可乘载的人数、运行速度、发动机的功率、耗油量、自重、轮子数目等。另外，还有几个重要的行为（功能）：加速、减速、刹车、转弯等。可以把这些行为称做是它们具有的方法，而属性是它们的状态描述，仅仅用属性或行为不能很好地描述它们。在现实生活中，用这些共有的属性和行为给出一个概念——机动车类。也就是说，人们经常谈到的机动车类就是从具体的实例中抽取共同的属性和行为形成的一个概念，那么一个具体的轿车就是机动车类的一个实例，即对象。一个对象将自己的数据和对这些数据的操作合理有效地封装在一起，例如，每辆轿车调用“减速”行为改变的都是自己的运行速度。

2．继承

继承体现了一种先进的编程模式（见第 3 章）。子类可以继承父类的属性和行为，即继承父类所具有的数据和数据上的操作，同时又可以增添子类独有的数据和数据上的操作。比如，“人类”自然继承了“哺乳类”的属性和行为，同时又增添了人类独有的属性和行为。

3. 多态

多态是面向对象编程的又一重要特征。有两种意义的多态。一种是操作名称的多态，即有多个操作具有相同的名字，但这些操作所接收的消息类型必须不同。例如，让一个人执行“求面积”操作时，他可能会问你求什么面积。所谓操作名称的多态性，是指可以向操作传递不同消息，以便让对象根据相应的消息来产生相应的行为。另一种是和继承有关的多态，是指同一个操作被不同类型对象调用时可能产生不同的行为。例如，狗和猫都具有哺乳类的功能：“喊叫”。但是，狗操作“喊叫”产生的声音是“汪汪……”；而猫操作“喊叫”产生的声音是“喵喵……”。

1.1.4 使用 Java 的必要性

本书是面向有一定 Java 语言基础和一定编程经验的读者，因此读者在学习本书之前应当有一定的 Java 语言基础，熟悉一种 Java 开发工具（如 Java SE 提供的 JDK）。和 C++不同，Java 语言是纯面向对象编程的语言，不支持面向过程，并且更适合用来实现面向对象的程序设计。GOF 所著作的《设计模式》一书无疑是经典之作，但是该书中的示例代码相当简练，而且是采用 C++描述的，另外 C++中提到的接口就是指类的方法，但是在 Java 中，类和接口是两个不同的概念。本书采用 Java 语言讲解面向对象和设计模式的另一个原因是希望通过本书的学习能让读者更加熟悉怎样用 Java 语言来体现面向对象的设计思想。

1.2 从抽象到类

面向对象语言有 3 个重要特性：封装、继承和多态，其中，封装性是最基本的特性之一。首先观察下列简单的能输出矩形面积的 Java 应用程序的源文件：

ComputerRectArea.java

```
public class ComputerRectArea  {
   public static void main(String args[])  {
      double height;          //高
      double width;           //宽
      double area;            //面积
      height=23.89;
      width=108.87;
      area=height*width;      //计算面积
      System.out.println(area);
   }
}
```

上述 Java 应用程序输出宽为 108.87、高为 23.89 的矩形面积。

通过上述 Java 应用程序注意到这样一个事实：

如果其他 Java 应用程序也想计算矩形面积，同样需要知道使用矩形的宽和高来计算矩形面积的算法，即也需要编写和这里同样多的代码。现在提出如下问题：

能否将和矩形有关的数据以及计算矩形面积的算法进行封装，使得需要计算矩形面积

的 Java 应用程序无须编写计算面积的代码就可以计算出矩形面积呢？

面向对象的一个重要思想就是通过抽象得到类，即将某些数据以及针对这些数据上的操作封装在一个类中。也就是说，抽象的关键点有两点：一是数据；二是数据上的操作。

这里对所观察的矩形做如下抽象：

- 矩形具有宽和高的属性。
- 可以使用矩形的宽和高计算矩形面积。

现在根据如上的抽象，定义出如下的 Rect 类。

Rect.java

```
public class Rect
{
   double width;     //矩形的宽
   double height;    //矩形的高
   double getArea() //计算面积的方法
   {
      double area=width*height;
      return area;
   }
}
```

1．类声明

在上述代码中（第一行），class Rect 称做类声明，Rect 是类名。

2．类体

类声明之后的一对大括号{、}以及它们之间的内容称做类体，大括号之间的内容称做类体的内容。

上述 Rect 类的类体的内容由两部分构成：一部分是变量的声明，称做域变量或成员变量，用来刻画矩形的属性，如 Rect 类中的 width 和 height；另一部分是方法的定义（在 C 语言中称做函数），用来刻画行为功能，如 Rect 类中的 double getArea()方法。

Rect 类好比是生活中电器设备需要的一个电阻，如果没有电器设备使用它，电阻将无法体现其作用。

以下的 Java 应用程序的主类（含有 main 方法的类，Java 应用程序从主类开始运行）中使用 Rect 类创建对象，该对象可以完成计算矩形面积的任务，而使用该对象的 Java 应用程序的主类无须知道计算面积的算法就可以计算出矩形面积。

Application.java

```
public class Application {
   public static void main(String args[])    {
      Rect rectangle1,rectangle2;   //声明两个对象
      rectangle1 = new Rect();      //创建对象
      rectangle2 = new Rect();
      rectangle1.width=128;
      rectangle1.height=69;
      rectangle2.width=18.9;
```

```
        rectangle2.height=59.8;
        double area=rectangle1.getArea();
        System.out.println("rectangle1的面积:"+area);
        area=rectangle2.getArea();
        System.out.println("rectangle2的面积:"+area);
    }
}
```

1.3 类与程序的基本结构

一个应用程序（也称为一个工程）是由若干个类所构成，这些类可以在一个源文件中，也可以分布在若干个源文件中，如图 1.1 所示。

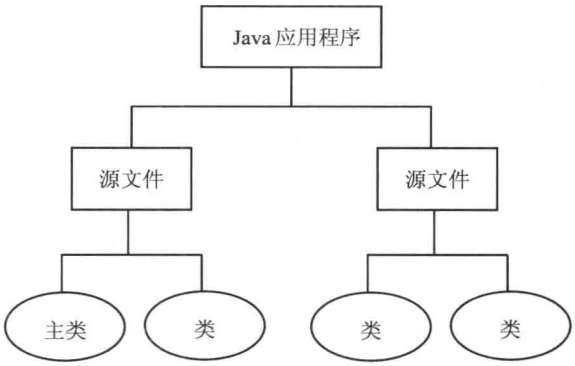


图 1.1 程序的结构

Java 程序以类为“基本单位”，即一个 Java 程序就是由若干个类所构成。一个 Java 程序可以将它使用的各个类分别存放在不同的源文件中，也可以将它使用的类存放在一个源文件中。一个源文件中的类可以被多个 Java 程序使用，从编译角度看，每个源文件都是一个独立的编译单位，当程序需要修改某个类时，只需要重新编译该类所在的源文件即可，不必重新编译其他类所在的源文件，这非常有利于系统的维护。从软件设计角度看，Java 语言中的类是可复用代码，编写具有一定功能的可复用代码是软件设计中非常重要的工作。

在下面的应用程序共有 3 个 Java 源文件，其中，Application.java 是主类。

Rect.java

```
public class Rect {
   double width;              //矩形的宽
   double height;             //矩形的高
   double getArea()  {
      double area = width*height;
      return area;
   }
}
```