高职高专计算机教学改革新体系规划教材

Java高级编程项目化教程

林萍　主　编
朱亚兴　朱婵　巫宇　副主编

清华大学出版社

高职高专计算机教学改革新体系规划教材

Java高级编程项目化教程

林萍 主编
朱亚兴 朱婵 巫宇 副主编

清华大学出版社
北京

内 容 简 介

本书以培养学生的实际动手能力为中心目标，以职业素质为突破点，以实用技能为核心，以案例为驱动，以讲练结合为训练思路，首先讲解面向对象的三大特性及其应用；然后讲解异常、集合类、I/O 读写；最后讲解多线程技术。通过学习本书，程序员在编程过程中能够逐渐精通面向对象和业务知识，最终成为架构师。

本书适合有一定编程基础的读者，可作为高职高专计算机相关专业高年级学生的 Java 课程强化教材。

图书在版编目(CIP)数据

Java 高级编程项目化教程/林萍主编. --北京：清华大学出版社，2015
高职高专计算机教学改革新体系规划教材
ISBN 978-7-302-38287-4

Ⅰ. ①J…　Ⅱ. ①林…　Ⅲ. ①JAVA 语言－程序设计－高等职业教育－教材　Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字(2014)第 231640 号

责任编辑：孟毅新
封面设计：傅瑞学
责任校对：袁　芳
责任印制：何　芊

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
课件下载：http://www.tup.com.cn，010-62795764
印 刷 者：北京富博印刷有限公司
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：14　　**字　　数**：321 千字
版　　次：2015 年 2 月第 1 版　　**印　　次**：2015 年 2 月第 1 次印刷
印　　数：1～2500
定　　价：32.00 元

产品编号：061994-01

前言

FOREWORD

面向对象一直是计算机界关心的重点，从20世纪90年代开始，它已成为主流的软件开发方法。现代企业级的应用系统业务复杂而繁多，代码量庞大，需要分析师、架构师、程序员、测试人员等合作完成。其中，架构师使用面向对象的方式设计系统所需要的类和接口。这些类和接口被分配到各个程序员，因此，理解和实现这些类和接口就成为程序员的主要工作。程序员在编程过程中逐渐精通面向对象和业务知识后，最终才可以成为架构师。

阅读本书之前最好有一门计算机语言基础，这有助于快速深入Java高级编程的世界。在本书中，读者将学到以下几个方面的内容。

第一部分(第1～5章)：讲解Java面向对象的核心内容，包括抽象和封装、继承、多态、抽象类和抽象方法、接口等，并用一个学生/教师信息系统贯穿整个部分，以使读者充分理解面向对象思想以及类与类之间的各种关系。

第二部分(第6～10章)：讲解Java中非常重要的知识，包括异常、输入/输出、集合、图形用户界面和多线程。异常处理机制使程序中的业务代码与异常处理代码分离，从而使代码更加简洁。输入/输出实现对文件的读写操作。集合弥补了数组的缺陷，更灵活更实用，可以大大提高软件的开发效率。图形用户界面使人机交互更加容易、方便，使用它可以直观地查看软件的功能。多线程使程序不再是顺序流程执行，而是让CPU分配时间来执行。这几个单元采用了一个点名器案例来贯穿讲解，整个案例简单有趣，能很好地融会贯通这些知识，从而增强了本书的趣味性。

本书以培养学生的实际动手能力为中心目标，以职业素质为突破点，以实用技能为核心，以案例为驱动，以讲练结合为训练思路。

本书每章围绕要完成的任务所需解决的问题导出对应的学习内容和知识点，然后讲解必要内容及解决问题的过程和步骤，再通过适当题材的练习巩固、强化所学知识，即实现“教、学、做”一体化。因此，使用本书作为教材时，最好采用适于“教、学、做”一体化的多媒体实训室或机房进行教学，效果会更好，最终达到“学用结合，以用为本，学以致用”的教学目的。

本书每章的“上机练习”可以巩固所学基础知识，也能够逐步培养学生的综合能力。每章的练习循序渐进，10章的综合实战结束后，也恰好完成了一个小型项目“点名器”，让同学们体会到Java编程的乐趣和成就感。这个

点名器也可以在一开始上课时就使用，它将大大地提高学生对 Java 的学习兴趣，这也是本书的一个特色。

本书不仅面向在校学生，也紧密联系企业实践。编者邀请有经验的企业一线 Java 程序员和相关项目经理参与教材的编撰，他们对教材案例的选取和知识点的遴选给了很好的建议与意见，充分体现了以实用技能为核心的思路。

本书的内容按 80/20 原则来取舍：书中选取的企业中使用频率很高的 20%的内容要求花读者 80%的精力去学好；而使用频率较低的 80%的内容只要求读者花 20%的精力去了解，真正践行“好钢用在刀刃上”和“抓主要矛盾”的理念。

本书通过简单有趣的案例使学生轻松掌握相关的知识点，使枯燥的知识学习过程变得简单化、趣味化。书中各个知识点环环相扣，连接紧密；各章知识循序渐进，由浅入深，体系合理，10 章的内容完整地为后续的课程打好了基础。

本书由林萍主编，负责全书的统稿工作。林萍编写第 1、2、8、9、10 章；朱亚兴编写第 3 章；朱婵编写第 4、6、7 章；企业副董事长巫宇编写第5 章，并对全书的实例和知识点的选择给出建议。另外，企业工程师范运标、唐月、谭月爱和刘平也对本书的编写提出了很好的建议，在此对所有给予本书支持、帮助的同人致以深深的谢意！

本书有配套的课程资源网站和项目资源库网站，网址分别如下。

http://61.145.231.44:8080/skills/solver/classView.do?classKey=6283824

http://61.145.231.44:8080/skills/solver/classView.do?classKey=7123323

由于编者水平有限，书中难免有不足之处，欢迎各位读者与专家批评、指正。

编　者

2015 年 1 月

目录

CONTENTS

第 1 章

Java 面向对象语言基础

Chapter 1

Java 的简单首先体现在精简的系统上，力图用最小的系统实现足够多的功能；对硬件的要求不高，在小型的计算机上便可以良好地运行。和所有新一代的程序设计语言一样，Java 也采用了面向对象技术并更加彻底，所有的 Java 程序均是对象，封装性实现了模块化和信息隐藏，继承性实现了代码的复用，用户可以建立自己的类库。而且 Java 采用的是相对简单的面向对象技术，去掉了运算符重载、多继承的复杂概念，而采用了单一继承、类强制转换、多线程、引用(非指针)等方式。无用内存自动回收机制也使得程序员不必费心管理内存，使程序设计更加简单，同时大大减少了出错的可能。Java 语言采用了 C 语言中的大部分语法，熟悉 C 语言的程序员会发现 Java 语言在语法上与 C 语言极其相似。

本章首先简要介绍 Java 的类和对象，然后介绍 Java 的方法开发，最后介绍 Java 中的数组和 Java 编程规范。

技能目标

理解类和对象的概念。

理解方法。

理解数组。

1.1 一切事物皆对象

任务描述

如何把“学生”用 Java 语言描述出来并输出学生信息？

任务分析

Java 把一切事物都当做对象，类是所有对象中属性和方法的抽象，对象是类的实例化。

相关知识与实施步骤

1. 类

一说到面向对象程序设计，人们首先想到的概念就是类和对象，那么什么是类，什么是对象呢？面向对象的第一个原则是把数据和对该数据的操作都封装在一个类中，在程序设计时要考虑多个对象及其相互间的关系。

Java 类的定义必须用到关键字 class。Java 类的组成比较简单，一般包括属性和方法，属性就是一些保持不变的东西，方法一般是指动作。例如，学生可以看作一个类，那么学生类可以有一些基本标识，比如学号、姓名、性别、年龄等，这些内容在 Java 面向对象编程中可以用属性来表示。属性也叫成员变量或者全局变量，就是直接隶属于类的变量。

```
//代码 1-1
public class Student {
    String id;
    String name;
    char gender;
    int age;
}
```

代码 1-1 定义了 4 个成员变量，也叫做类 Student 的属性。在生活中，对象一般除了有基本保持不变的静态属性外，还有自己的动作或者作用，这个动作或者作用在面向对象程序设计中用方法来表示。例如，学生可以学习、吃饭、睡觉等。如代码 1-2，在 Student 类中增加了一些学生的动作。

```
//代码 1-2
public class Student {
    String id;
    String name;
    char gender;
    int age;
    void study(){
        System.out.println(name+"正在学习 Java 进阶与提高!");
    }
    void eat(){
        System.out.println(name+"正在吃饭");
    }
}
```

这样，学生类才能比较准确地体现出活生生的“学生”模样。从代码 1-2 可以看出，Java 的类是由属性和方法组成的，属性和方法有机地组合在一起，才能真正完整地体现一个事物对象。代码 1-2 中定义了 4 个属性和 2 个方法，它们组合在一起使学生成为一个整体。也就是说，提到学生，就应该具有学号、姓名、性别、年龄这 4 个属性和学习、吃饭 2 个方法。类也是 Java 的一种数据类型，和基本数据类型的使用差不多，可以用来声明一个变量，然后使用这个变量。

注意：Java的类是由属性和方法组成的，初学者很容易把语句也搞成类的组成部分。例如，经常有同学如下定义类。

```
//代码 1-3
public class Student {
    String id;
    String name;
    char gender;
    int age;
    System.out.println(name+"正在学习 Java 的类!");
                        //错误,这条语句不是属性定义,也不是方法,不符合 Java 语法规则
    void study(){
        System.out.println(name+"正在学习 Java 进阶与提高!");
    }
    void eat(){
        System.out.println(name+"正在吃饭");
    }
}
```

初学者很容易将一条输出语句直接定义在Java类下面，编译器直接就说这条语句错误，但是初学者经常搞不明白错误原因。这里强调一下，Java类的组成非常简单，就是属性和方法，没有其他多余的东西，所以在发现有类似错误的时候，请检查是否有输出语句等内容直接放在类中了。

2. 对象

上面讲到，类定义完成后就是一种数据类型，和基本数据类型使用差不多，那么类到底该怎么使用呢？类的使用分两个步骤。①声明和初始化；②采用“.”运算符调用属性和方法。

下面对这两个步骤逐一进行详细的介绍。

(1) 声明和初始化

类的定义完成后，就可以用它来声明对象，这点和基本数据类型的声明差不多，比如，使用上面的 Student 类来声明一个具体的对象。

```
Student s1;          //类似于 int i;
s1=new Student();    //类似于 i=10;
```

上面第一行“Student s1;”表示 s1 是一个 Student 的对象；第二行接着给 s1 赋初值，给对象赋初值必须用到关键字 new。当然，定义和赋初值也可以在一行中完成。

```
Student s1=new Student();
```

(2) 对象的使用

定义好对象之后，那么对象 s1 就具备了类 Student 中的 4 个属性和两个方法，通过“.”运算符可以给属性赋值，也可以调用方法。

```
s1.name="张三";
```

```
s1.study();
```

当然，如果再定义一个对象 s2，s2 也具有 4 个属性和两个方法。

```
Student s2=new Student();
s2.name="李四";
s2.study();
```

s1 和 s2 在内存中是两个不同的存储单元。同基本数据类型相比，s1 和 s2 属于引用数据类型，它们是指向某个内存单元的引用，不像基本数据类型是直接给这个内存取了个名字，两者的区别如图 1-1 所示。

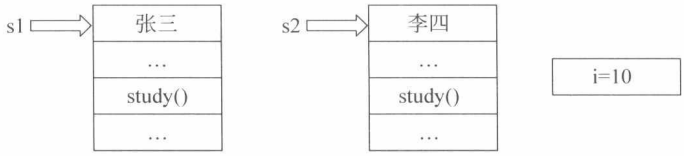

图 1-1　引用数据类型和基本数据类型在内存中的区别

对象 s1 和 s2 其实是指向内存空间单元的一个引用，这就好比内存是电视机，s1 和 s2 就是电视机遥控器，通过遥控器来给内存单元赋值和调用内存单元中的方法。如果使用以下语句给对象赋值，就会造成无用内存。

```
s2 =s1;
```

那么刚刚 s2 的内存单元就会成为无用的空间，需要 Java 垃圾回收机制来回收。采用上面赋值后，它们的关系如图 1-2 所示。

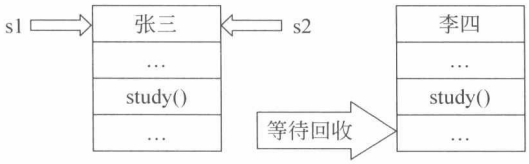

图 1-2　s1 和 s2 对象指向同一个空间单元，另一个空间单元等待回收

读者一定要明白对象在内存中的分配和存储，在用对象给对象赋值后，一定要搞清楚内存单元的变化。弄懂了内存单元的变化，很多对象引用的问题就迎刃而解了。

1.2　方法的声明与使用

任务描述

（1）类设计好之后，应该如何使用它？程序从哪里开始执行？

（2）学生对象属性赋值后，请输出学生信息。

（3）从键盘上录入学生 3 门课的成绩，计算平均成绩并输出。

任务分析

从某种意义上讲，写程序就是在写方法，方法设计是否合理恰当，直接关系到整个项目的运行等。Java 程序的入口是 main()方法，这是程序的起点，所以要单独列出讲解。另外，Java 中的方法从参数角度分，可以分为无参数方法和有参数方法。下面分别介绍它们。

相关知识与实施步骤

1. main()方法

在 Java 中，程序都是以类的方式组织的，Java 源文件都保存在以.java 为后缀的文件当中。每个可运行的程序都是一个类文件，或者称之为字节码文件，保存在.class 文件中。而作为一个 Java Application 程序，类中必须包含主方法，程序的执行从 main()方法开始，方法头的格式是确定不变的。

```
public static void main(String args[])
```

其中关键字 public 意味着方法可以由外部世界调用。main()方法的参数是一个字符串数组 args，虽然在以下程序中没有用到，但是必须列出来。如代码 1-4 所示。

```
//代码 1-4
public class Test {
    public static void main(String[] args) {
        Student s1=new Student();
        s1.id="20123101";
        s1.name="张三";
        s1.gender='男';
        s1.age=20;
        s1.study();
        s1.eat();
        Student s2=new Student();
        s2.id="20123102";
        s2.name="李四";
        s2.gender='男';
        s2.age=21;
        s2.study();
        s2.eat();
    }
}
```

有了 main()方法就可以运行了，运行并输出结果以下。

```
张三正在学习 Java 高级!
张三正在吃饭
李四正在学习 Java 高级!
李四正在吃饭
```

在类 Test 中，定义了 main()方法，main()方法由 4 部分组成，缺一不可。然后是方法主体，第一条是产生一个 Student 类的对象 s1，接着用对象名访问对象的属性，给属性赋值，比如“s1. name = "张三";”，再调用对象的方法，比如“s1. study();”；然后又产生了对象 s2，通过同样方式赋值并调用方法。

从代码 1-4 中，可以了解到 main()方法是整个程序的入口，main()方法中可以方便地使用 Student 类，Student 类作为一个整体具有自己的属性和方法，声明类对象后，对象就具有类中的所有属性和方法，并且可以给对象中的属性赋自己的值。

2. 无参方法

设计一个方法，这个方法可以让学生自我介绍，然后直接输出学生的属性信息，如代码 1-5 所示。

```
//代码 1-5
public class Student {
    String id;
    String name;
    char gender;
    int age;

    void study(){
        System.out.println(name+"正在学习 Java 高级!");
    }
    void eat(){
        System.out.println(name+"正在吃饭");
    }
    public String toString(){
        return "我是"+name +",我的学号是: "+id+",我是"+gender+"生,我今年"+age+
        "岁了!";
    }
}
```

上面代码在 Student 类中多增加了一个 toString()方法，方法返回学生的自我介绍信息。修改代码 1-4，在 main()方法中增加一条输出信息，并在输出语句中调用 toString()方法，如代码 1-6 所示。

```
//代码 1-6
public class Test {
    public static void main(String[] args) {
        Student s1=new Student();
        s1.id="20123101";
        s1.name="张三";
        s1.gender='男';
        s1.age=20;
        s1.study();
        s1.eat();
        System.out.println(s1.toString());
```

```
    }
}
```

运行代码，输出如下结果。

```
张三正在学习 Java 高级!
张三正在吃饭
我是张三,我的学号是: 20123101,我是男生,我今年 20 岁了!
```

从上面代码看出，Java 无参方法比较简单，其定义语法如下。

```
public 返回值类型 方法名() {
    //这里编写方法的主体
}
```

其中方法名的命名一定要符合 Java 命名规则，Java 的属性和方法命名规则如下。

(1) 命名都采用驼峰式的命名，类名以大写字母开头，方法名、属性名都以小写字母开头，后面是字母数字或者下划线字符串。

(2) 命名要有一定的意义，最好看到名称就知其用处。

方法的返回值分以下两种情况。

(1) 如果方法具有返回值，方法中必须使用关键字 return 返回该值，返回类型为该返回值的类型，如代码 1-5 中的 toString()方法，返回值类型是 String，所以必须用 return 语句返回一个 String 类型的值。

(2) 如果方法没有返回值，返回类型为 void，代码 1-5 中 study()方法和 eat()方法就是这种情况。

方法的调用也分为两种情况。

(1) 类内部互相调用，可以直接写方法名。假设 eat()方法中调用 study()方法，代码如下。

```
void eat(){
    study();    //直接调用即可
    System.out.println(name+"正在吃饭");
}
```

(2) 类外面调用，如上面示例中，在类 Test 中调用 Student 类的方法，必须通过 Student 类的对象才能调用。

```
public static void main(String[] args) {
    Student s1=new Student();
    s1.study(); //必须通过对象才能调用
    s1.eat();
    System.out.println(s1.toString());
}
```

无参方法相对来说比较简单，在设计无参方法时只须考虑是否有返回值即可，调用的时候根据是否有返回值来决定是单独一行还是作为一个表达式。上面的“s1. eat();”调用就是单独成行，像 eat()方法这样无参无返回值的只能这样调用。“System. out. println

(s1.toString());"语句中,就是把 s1.toString()方法的调用结果给另一个方法(println()方法)做参数。

提示:凡是在类中通过 public String toString()的方式定义的 toString()方法不需要显示调用,也就是说下面两条语句效果相同。

```
System.out.println(s1.toString());
System.out.println(s1);
```

3. 带参数的方法

在 StudentScore 类中,设计一个方法,这个方法的功能是计算学生 3 门课程的平均成绩。

```
//代码 1-7
public class StudentScore {
    public float calcAvg(int scoreJava,int scoreSQL,int scoreEnglish){
        return (scoreJava+scoreSQL+scoreEnglish)/3.0f;
    }
    public static void main(String[] args) {
        StudentScore ss=new StudentScore();
        System.out.println(ss.calcAvg(94,64,78));
    }
}
```

运行代码,输出结果如下。

```
78.666664
```

方法 calcAvg()是一个带有 3 个整型参数的方法,目的是计算这 3 门课程的平均成绩并返回,在 main()方法中,ss.calcAvg(94,64,78)调用方法的时候,就会把 94 赋值给参数 scoreJava,64 赋值给参数 scoreSQL,78 赋值给参数 scoreEnglish,计算完成后把结果作为 println()方法的参数,并最后输出。

带参数的方法也并不复杂,关键是要在设计的时候确定带几个参数,什么类型的参数。带参数的方法比较难以理解和掌握的是参数的类型问题,即参数是基本数据类型还是引用数据类型,这两种类型会导致处理结果不同。

```
//代码 1-8
class A{
    int a;
    int b;
}

public class TestParam {
    public void exchange(int x, int y){
        System.out.println("交换前: x="+x+",y="+y);
        int temp=x;
        x=y;
```

```
        y=temp;
        System.out.println("交换后：x="+x+",y="+y);
    }
    public void exchange(A a){
        System.out.println("交换前：a.a="+a.a+",a.b="+a.b );
        int temp=a.a;
        a.a=a.b;
        a.b=temp;
        System.out.println("交换后：a.a="+a.a+",a.b="+a.b );
    }

    public static void main(String[] args) {
        A a=new A();
        a.a=1;
        a.b=2;
        TestParam tp=new TestParam();
        int x =5;
        int y=10;
        System.out.println("在 main()方法中,交换前：x="+x+",y="+y);
        tp.exchange(x, y);
        System.out.println("在 main()方法中,交换后：x="+x+",y="+y);

        System.out.println("\n\n在 main()方法中,交换前：a.a="+a.a+",a.b="+a.b);
        tp.exchange(a);
        System.out.println("在 main()方法中,交换后：a.a="+a.a+",a.b="+a.b);
    }
}
```

运行代码，结果输出如下。

```
1: 在 main()方法中,交换前：x=5,y=10
2: 交换前：x=5,y=10
4: 交换后：x=10,y=5
5: 在 main()方法中,交换后：x=5,y=10


6: 在 main()方法中,交换前：a.a=1,a.b=2
7: 交换前：a.a=1,a.b=2
8: 交换后：a.a=2,a.b=1
9: 在 main()方法中,交换后：a.a=2,a.b=1
```

上面的输出结果中，从第 5 行和第 9 行就能看出参数是基本数据类型和引用数据类型的区别。

(1) exchange(int x, int y)带有两个整型参数，整型参数在传递数据的时候是值传递，也就是 main 方法中 x,y 的值传给了 exchange 中的 x 和 y，在方法里面交换后，不影响 main()方法中 x,y 的值。因此在 main 方法中，x 和 y 的值始终保持不变。

(2) exchange(A a)带有一个引用数据类型，类 A 的对象参数，当 main()方法中的 a 传给 exchange 中的 a 时，实际上传递的是对象 a 的地址值，也就是前面提到的“遥控器”，