高等学校计算机专业教材精选·算法与程序设计

Java高级程序设计

徐传运 张杨 王森 编著

清华大学出版社

高等学校计算机专业教材精选·算法与程序设计

Java高级程序设计

徐传运 张杨 王森 编著

清华大学出版社
北京

内 容 简 介

Java是目前最流行的开发语言之一，本书以Java语言中的8个高级主题作为核心内容，使读者在掌握了基本的Java语言基础之上，学习Java的高级技术，提升编程能力，并为掌握其他Java扩展技术打下基础。本书内容包括类型信息与反射、泛型、注解、网络编程、多线程、序列化、数据库编程、Web编程基础等。除此之外，本书还在第1章论述了程序设计的原则和规范，在最后1章简要介绍了Tomcat的原理。本书附带的电子材料包括：电子课件(PPT)、示例源代码、部分课后习题的参考答案。

本书适合作为高等院校软件工程、计算机相关专业的编程能力提升课程的教材，即在Java编程基础课程之后的高级Java编程课程的教材，也可以作为JEE课程的前导基础课程的教材。本书亦可以作为有编程经验的软件开发人员作为编程能力提升的参考书。

图书在版编目(CIP)数据

Java高级程序设计/徐传运，张杨，王森编著. —北京：清华大学出版社，2014
高等学校计算机专业教材精选·算法与程序设计
ISBN 978-7-302-34672-2

Ⅰ. ①J… Ⅱ. ①徐… ②张… ③王… Ⅲ. ①JAVA语言－程序设计－高等学校－教材 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2013)第290835号

责任编辑：张　玥　薛　阳
封面设计：傅瑞学
责任校对：梁　毅
责任印制：李红英

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，010-62795954
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**：20.5　　**字　　数**：512千字
版　　次：2014年6月第1版　　**印　　次**：2014年6月第1次印刷
印　　数：1～2000
定　　价：34.50元

产品编号：056100-01

出版说明

我国高等学校计算机教育近年来迅猛发展，应用所学计算机知识解决实际问题，已经成为当代大学生的必备能力。

时代的进步与社会的发展对高等学校计算机教育的质量提出了更高、更新的要求。现在，很多高等学校都在积极探索符合自身特点的教学模式，涌现出一大批非常优秀的精品课程。

为了适应社会的需求，满足计算机教育的发展需要，清华大学出版社在进行了大量调查研究的基础上，组织编写了《高等学校计算机专业教材精选》。本套教材从全国各高校的优秀计算机教材中精挑细选了一批很有代表性，且特色鲜明的计算机精品教材，把作者们对各自所授计算机课程的独特理解和先进经验推荐给全国师生。

本系列教材特点如下。

(1) 编写目的明确。本套教材主要面向广大高校的计算机专业学生，使学生通过本套教材，学习计算机科学与技术方面的基本理论和基本知识，接受应用计算机解决实际问题的基本训练。

(2) 注重编写理念。本套教材作者群为各校相应课程的主讲，有一定经验积累，且编写思路清晰，有独特的教学思路和指导思想，其教学经验具有推广价值。本套教材中不乏各类精品课配套教材，并力图把不同学校的教学特点反映到每本教材中。

(3) 理论知识与实践相结合。本套教材贯彻从实践中来到实践中去的原则，书中的许多必须掌握的理论都将结合实例来讲，同时注重培养学生分析、解决问题的能力，满足社会用人要求。

(4) 易教易用，合理适当。本套教材编写时注意结合教学实际的课时数，把握教材的篇幅。同时，对一些知识点按教育部教学指导委员会的最新精神，进行合理取舍与难易控制。

(5) 注重教材的立体化配套。大多数教材都将配套教师用课件、习题及其解答，学生上机实验指导、教学网站等辅助教学资源，方便教学。

随着本套教材陆续出版，相信能够得到广大读者的认可和支持，为我国计算机教材建设及计算机教学水平的提高，为计算机教育事业的发展做出应有的贡献。

清华大学出版社

序

Java 诞生于 1995 年，是由 Sun Microsystems 公司于 1995 年 5 月推出的 Java 程序设计语言和 Java 平台的总称。Java 原是印度尼西亚爪哇岛的英文名称，盛产咖啡，因此 Java 语言中许多库、类的名称与咖啡有关，如 JavaBeans(咖啡豆)、NetBeans(网络豆)、ObjectBeans(对象豆)等。

十多年来，Java 就像爪哇咖啡一样誉满全球，成为企业级应用平台的霸主，而 Java 语言也如咖啡一样醇香动人。由于通常被用在网络环境中，Java 语言不但全面支持 Internet 应用的开发，提供丰富的、用于网络编程的类库，而且除具有许多安全特性外，还具有一个专门针对通过网络下载的类的安全防范机制，如分配不同的名字空间以防替代本地的同名类、字节代码检查，并提供安全管理机制让 Java 应用设置安全哨兵。Java 语言的设计目标之一是适应于动态变化的环境，因此 Java 程序在 Java 平台上被编译为体系结构中立的字节码格式，然后可以在实现这个 Java 平台的任何系统中运行，这种方式适合异构的网络环境和软件的分发；同时，Java 程序需要的类能够通过网络动态地载入到运行环境，这也有利于软件的升级。Java 对对象技术的全面支持和 Java 平台内嵌的 API 能缩短应用系统的开发时间，并降低成本，特别是 Java 企业应用编程接口为企业计算和电子商务应用系统提供了有关技术和丰富的类库。

以 Sun 公司公布的 Java 最新框架结构为标准，将 Java 语言以 Java2 为中心，其组成分为以下三个部分。

(1) 企业版 J2EE。该版本是面对各大企业环境为中心而开发的一种以应用程序为主体的计算机网络平台，其中还包括 EJB、JSP 和 Servlet 三个层次。

(2) 标准版 J2SE。其中，Java 核心编程为图形用户界面的编程、工具包程序的编写以及数据库的程序编写等。

(3) 微型版 J2ME。该版本一直以消费品和各种嵌入式设备的网络应用平台为研究中心，主要涉及的领域为手机、手机中的各种无线游戏等，其核心技术为移动信息设备小程序。

当下，Java 正被广泛应用于计算机软件的开发，尤其是 Web 领域。由于 Sun、IBM、Oracle、BEA 等国际厂商相继推出各种基于 Java 技术的应用服务器和应用软件，带动了 Java 在金融、电信、制造等领域日益广泛的应用。而无线手持设备、通信终端、医疗设备、信息家电、汽车电子设备等是近来比较热门的 Java 应用领域，最新的统计数据表明，Android 的使用率已经超过苹果的电子设备，也就是说 Java 在移动平台上的使用率将随之上升。未来，Java 在 Web、移动设备、云计算等方面前景广阔，越来越多的企业将其应用部署在 Java 平台上，例如 Salesforce.com 和 VMware 的 VMforce 服务将在云计算中布置 Java 应用、Google 应用引擎仍旧一如既往地支持 Java。在 Oracle 的技术投资担保下，Java 也是企业在云应用方面回避微软平台、在移动应用方面回避苹果公司的一个最佳选择。

目前，对 Java 前景的争议主要集中在 Oracle 可能关于 Java 的政策，以及 Java 与其他程序语言的竞争方面。至于前者，我们认为 Oracle 不可能不尽全力去发展 Java 这个唯一的编程语言；至于后者，每个程序语言都有其特点和适合领域，不是简单地一个语言替代另一个语言的问题。因此，我们相信在将来的很长一段时间内，Java 依然是主要的企业级应用开发语言。

前　言

当前，中国在软件开发领域拥有大量的现有程序员资源和潜在的程序员资源（即各大院校软件工程专业的本科生和硕士生），但丰富的人口红利并没有带来与之相当的技术创新优势。从业者大多停滞在单纯使用技术的低层次阶段，而难以对技术进行与应用相关的主动创新。笔者认为这与当下高校在软件工程（尤其是软件项目开发）教学中各门课程没能环环相扣有关，也与有针对性的相关原理性讲解的专业书籍较少有一定关系。

现有“Java 程序设计基础”课程相对应的相关教材主要讲述 Java 语言的基本语法（包括 Java 语言基础、数据类型、Java 类和对象等），而软件工程专业普遍开设的 J2EE 课程相对应的内容又主要是 Servlet/JSP、SSH（Struts，Spring、Hibernate）等企业级应用。为了填补 Java 程序设计基础和 J2EE 等 Java 高级应用之间的空白带，本书讲解了 Java 的高级技术以及高级技术的应用实例，让读者了解 Java 技术背后的原理。笔者认为学习技术不仅要会使用，还要知道技术后面的原理，这样才能深入地掌握技术，才能快速、彻底解决技术使用过程中出现的问题，才能科学客观地评估技术存在的风险，才能有效地优化技术的使用效率。因此本书通过对 Java 高级技术的讲解，除了让读者学会 Java 技术的使用，还要让读者明白 Java 技术后面的原理。

本书特色

1. 内容体系完整，从基础开始，由浅及深

教材是实现教学要求的重要保证，本书体系完整，注重应用，强调实践。

每一个章节的内容都是由浅入深、循序渐进地展开，使读者可以渐进地学习本书的全部知识。

2. 编著人员项目经验丰富，实例源于真实项目

本书的作者都是参加实际开发项目的负责人或主要成员，有丰富的 Java 程序开发实践经验，因此本书内容都是实际应用中确实需要的知识和技能。

本书所用实例全部来源于项目小组开发，且正在使用的真实项目，相关细节切合真实的软件开发实践环境。

3. 各章实例丰富，有助于读者理解所述知识

本书的每一个章节都提供了充分的实例，这些例子经过了精心设计与调试，能够恰当地展示相关知识点的实现细节。读者可以在学完相关理论知识后，通过上机实践这些实例来更加深入地了解，进而掌握这些知识点。

4. 使用最新版本的开发平台

本书所用的开发工具和相关框架在写作本书时都是最新版本，力图反映 Java 相关技术的新发展。读者可以在学习开发技术的同时，接触最新版本的开发平台，为以后的深入实践奠定基础。

5. 配有源代码等相关电子文档，方便读者使用

为了方便读者使用本书提供的大量示例程序，特将所有源代码都收录到本书附带的电子材料中，教师和读者可以运行这些代码，以利于读者更深入地理解相关的理论知识。

同时，本书配套资料中还提供了课后习题的参考答案，以供广大读者练习时借鉴。

另外，作为一本教材，本书还专门为广大教师配备了与教材内容一致的电子课件，以方便授课使用。

读者对象

- 初步掌握 Java 技术，想进一步学习 Java 高级编程的读者；
- 计算机专业的本科生；
- 非计算机专业的硕士研究生。

本书内容

Java 高级技术本身是由基本技术通过综合、交叉后发展而来，本书试图让读者了解这种从简单技术到复杂技术的演变过程，掌握演变规律，让读者有创新发明技术的能力。

第 1 章的内容主要是一些关于写出好代码的规则、惯例、模式。

第 2 章主要是类型信息与反射的相关知识，包括类型信息的存储、加载、核心类及其具体应用(即反射、动态代理)等内容。

第 3 章主要是泛型的相关知识，包括泛型的类、方法、接口、边界以及通配符等。

第 4 章主要是注解的相关知识，包括注解的使用、自定义及其处理器，以及实体映射与翻译等内容。

第 5 章主要是基于 Java 的网络编程，包括网络协议、流、TCP 编程、UDP 编程、HTTP 编程等内容。

第 6 章的内容主要是多线程的相关知识，包括线程基础知识、线程共享资源、线程协作、同步器等内容。

第 7 章的内容主要是序列化的相关知识，包括对象序列化、自定义序列化、XML 文件、JSON 等内容。

第 8 章的内容主要是基于 Java 的数据库编程，包括数据库基础知识、JDBC 及其进阶等。

第 9 章的内容主要是 Web 编程与 Tomcat，包括 Web 服务器概述、Servlet、JSP、Tomcat、简单的 Web 应用服务器等相关知识。

电子资源

本书附带的电子材料中主要有以下内容：

- 与教材内容一致的电子课件(PPT)；
- 本书中示例源代码；
- 本书各个章节部分课后习题的参考答案(仅向教师提供)。

目　录

第1章 关于代码

这是一本讲解 Java 高级编程技术的书，希望读者能够借此深入理解 Java 编程中一些比较高级的技术，这些技术在开发应用系统中可能并不常用，但在一定的应用场景却可能帮上大忙。之所以这些高端的编程技术对基于 Java 的软件开发至关重要，是因为当前一些功能强大的类库、框架就是在这些高端技术的支撑上创建出来的，如 Struts、Hibernate、Spring 等。

掌握了高级技术并不一定能写出好的代码，就像学会了大量的华丽词汇和富于表现的语法并不能写出好文章一样，还需要掌握很多关于编码的规则、惯例、模式，本章所讲解的正是一些关如何写出好代码的规则、惯例、模式。

1.1 编码的艺术

笔者时常感觉写代码就像写诗，好的代码能给人美感和艺术的享受。笔者常常被一段代码规范的命名、简洁优美的结构、精巧的细节而感动，感受到代码作者融入代码中的执着、用心、智慧，所以好的代码就是一件精美的艺术作品。

什么是好的代码呢？不同的程序员可能有不同的看法。《代码整洁之道》的作者 Robert C. Martin 认为“简洁的代码就是好的代码”；《C++ 程序设计语言》的作者 Bjarne Stroustrup 认为“优雅和高效的代码是好的代码”；《面向对象分析与设计》的作者 Grady Booch 认为“好的代码从不隐藏设计者的意图，充满了干净利落的抽象和直截了当的控制语句”；《重构》的作者 Martin Fowler 认为“没有坏味道的代码是好的代码”。本书认为“容易理解、容易修改的代码就是好的代码”。

代码的价值有两个：一个是告诉计算机应该怎么执行以完成软件的功能，另一个是告诉未来的代码修改者（包括代码原作者本人）代码的功能是什么。前一个价值是理所当然的，而后一个价值却经常被忽略。据统计，在代码的整个生命周期中，每一次写代码所花的工作量只占所有工作量的 30%，这就意味着大量的工作量花在代码的修改过程中。而在代码修改过程中，大部分的时间又花在对原代码的理解上。因此，代码需要直接、清晰地展现代码的功能。

代码的上述两种价值产生出两种“代码观”，即对代码的认识。一种认为：代码是指令的序列，在指令的驱动下，计算机完成期望的功能。另外一种认为：代码是语义的组合，每行代码体现一定的语义，所有代码语义的综合形成了系统的功能语义。两种代码认识观本质并不矛盾，两种认识观结合起来形成一种综合的认识观：代码是驱动计算机运行的指令序列，每个指令体现着一定的语义，指令的组合也是语义的组合，指令组合后计算机完成的功能应与语义组合后的综合语义所体现的功能一致。基于综合认识观，编码的过程就是把程序员所理解的语义翻译成计算机指令代码的过程，如果翻译后的代码在概念、结构上与语义相近，代码就更能直接体现程序员的意图，代码也具有更高的可读性。

下面将分别讲解代码块、函数、类、模块的编写方法，以达到使代码充满艺术美感的理想目标。

1.2 概念与命名

程序代码中，命名无处不在，如给变量、参数、函数、类、包、模块、子系统，甚至系统命名，好的名称能够清晰、直接地体现名称所代表对象的语义，如果代码没有好的命令，阅读这种代码无疑像阅读天书，例如下面代码的功能是什么？能看出来吗？

```
private float c(float a[],int b[])
{
    float d=0,e=0;
    int f=0;
    float s=0;
    for(int i=0; i<a.length; i++)
    {
        if(b[i]==2)
        {
            a[i]=a[i]*1.1f;
        }
        else
        {
            a[i]=a[i]*1.1f;
        }
        if(d<a[i])
        {
            d=a[i];
        }
        if(e>a[i])
        {
            e=a[i];
        }
    }
    for(int i=0; i<a.length; i++)
    {
        if(a[i] !=d && a[i] !=e)
        {
            f++;
            s=s+a[i];
        }
    }
    return s/f;
}
```

明白上面代码的功能了吗？上面的代码大约 30 行，如果这样的代码有 300 行、3000

行，你还有耐心读下去吗？上面的代码为什么难读，是因为代码中的方法、参数、变量、常量没有一个能体现语义的名称，没有办法把代码与帮助理解的语义直接关联起来，需要从代码的前后逻辑关系去猜测语义，然后在后面的代码中来验证这些语义，如果对语义的猜测是错误的，那就必须重新从头再读一遍代码，提出新的猜测。

下面将合适的命名给予上述代码：

```
public final static int SENIOR_JUDAGE =2;
public final static float SENIOR_JUDAGE_WEIGHT =1.1;
private float calculateAvgScore(float rawScores[],int judgeLevels[])
{
    float maxScore=0;
    float minScore=0;
    float actualScores[]=new float[rawScores.length];
    for(int i=0; i<rawScores.length; i++)
    {
        if(judgeLevels[i]==ScoreRecorder.SENIOR_JUDAGE)
        {
            actualScores[i]=rawScores[i] * ScoreRecorder.SENIOR_JUDAGE_WEIGHT;
        }
        else
        {
            actualScores[i]=rawScores[i];
        }
        if(maxScore<actualScores[i])
        {
            maxScore=actualScores[i];
        }
        if(minScore>actualScores[i])
        {
            minScore=rawScores[i];
        }
    }
    int validScoreCount=0;
    float sumScore=0;
    for(int i=0; i<rawScores.length; i++)
    {
        if(rawScores[i] !=maxScore && rawScores[i] !=minScore)
        {
            validScoreCount++;
             sumScore=sumScore+rawScores[i];
        }
    }
    float avgScore=sumScore/validScoreCount;
    return avgScore;
}
```