国外计算机科学教材系列

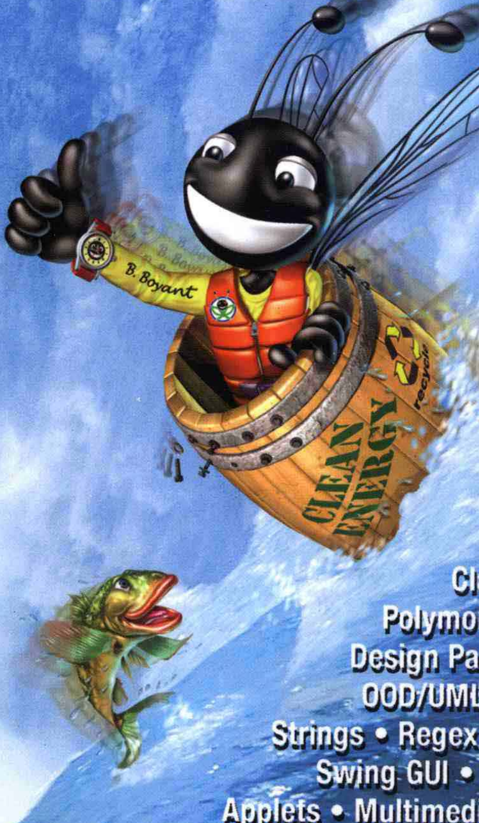
PEARSON

DEITEL®

Java大学教程

——（第八版）（英文版）——

Java How to Program, Eighth Edition



I/O
Types
Operators
Early Objects
Control Statements
Methods
Arrays • ArrayLists
Classes • Inheritance
Polymorphism • Interfaces
Design Patterns • Exceptions
OOD/UML 2 ATM Case Study
Strings • Regex • Files • Debugger
Swing GUI • Nimbus • Graphics
Applets • Multimedia • Java Web Start
Networking • Threads • SwingWorker
Recursion • Searching • Sorting
Data Structures • Generics • Collections
JDBC • Java DB • MySQL®
Web-Apps • Ajax • Web Services

[美] Paul Deitel
Harvey Deitel 著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

国外计算机科学教材系列

Java 大学教程

（第八版）
（英文版）

Java How to Program
Eighth Edition

［美］ Paul Deitel
Harvey Deitel 著

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内容简介

本书是关于Java语言的经典教材，秉承Deitel系列丛书的一贯特点：内容丰富、覆盖面广，提供详细代码与实例研究，总结出大量的面向对象编程技巧和经验。本书详细说明了在Java中面向对象编程的基本理论及实用知识，以初学者为起点，由点到面、由浅入深、循序渐进地介绍了事件处理、对象、接口、内置类、继承、多态性、数据结构和集合、流文件、applet、图形界面及多线程等多种Java特性。第八版在前一版的基础上增加了更多的实际案例，更新了很多内容，有助于读者学习和借鉴。本书包括更广泛的教学特性，其中列举了数百个可实际使用的程序实例，并给出其实际的运行结果，可以使学生在学习时更为直观。

本书结构清晰、逻辑性强，适合作为相关专业Java程序设计课程的双语教材，是所有对Java程序设计感兴趣的读者的有益参考书，也可供各类软件开发人员参考。

图书在版编目(CIP)数据

Java大学教程：第8版：英文/(美)戴特尔(Deitel，P.)，(美)戴特尔(Deitel，H.)著.
北京：电子工业出版社，2012.10
书名原文：Java How to Program，Eighth Edition
国外计算机科学教材系列
ISBN 978-7-121-18188-7

I. ①J…　II. ①戴…②戴…　III. ①JAVA语言－程序设计－高等学校－教材－英文　IV. ①TP312

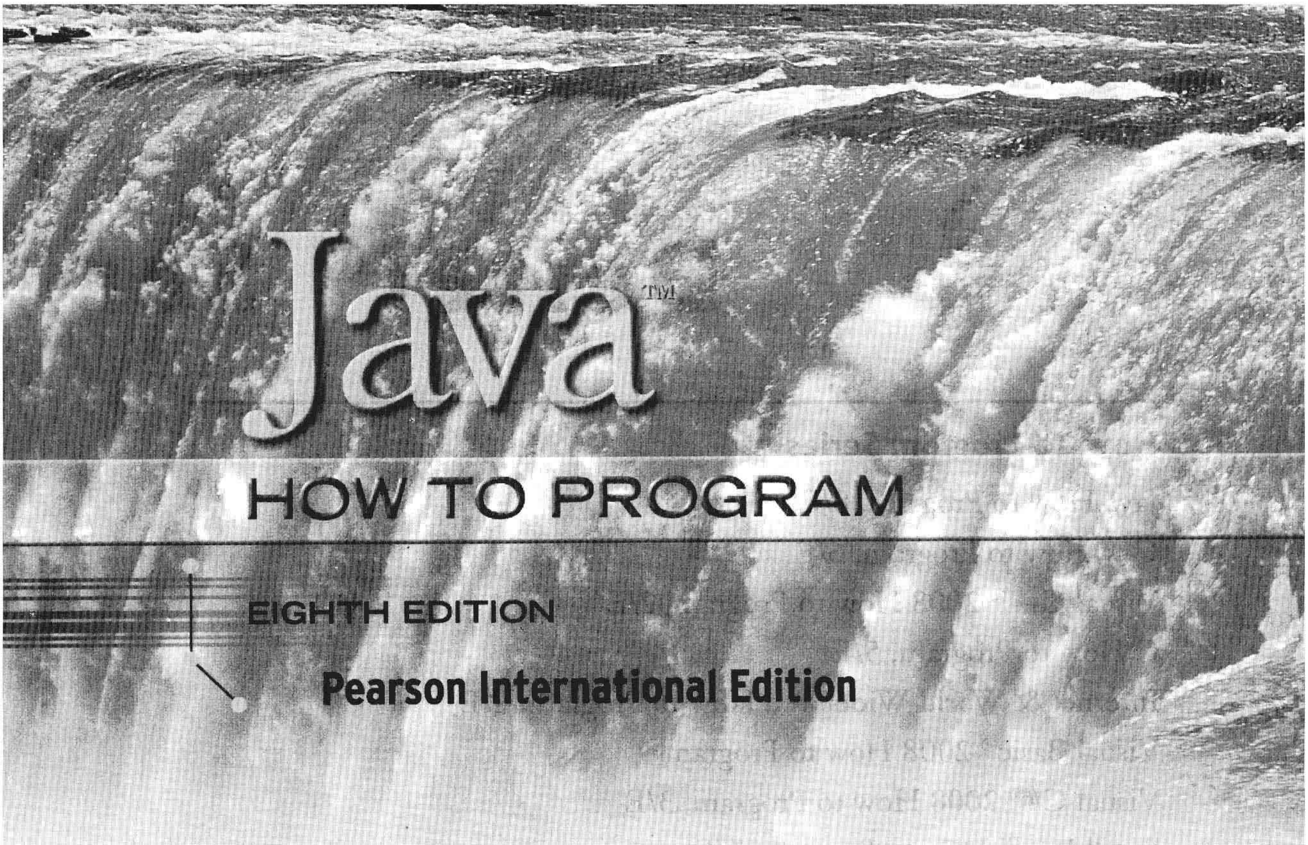
中国版本图书馆CIP数据核字(2012)第209704号

策划编辑：冯小贝
责任编辑：冯小贝
印　　刷：三河市鑫金马印装有限公司
装　　订：三河市鑫金马印装有限公司
出版发行：电子工业出版社
　　　　　北京市海淀区万寿路173信箱　邮编　100036
开　　本：787×1092　1/16　印张：71　字数：2717千字
印　　次：2012年10月第1次印刷
定　　价：128.00元(含光盘1张)

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(010)88254888。

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线：(010)88258888。



Java™

HOW TO PROGRAM

EIGHTH EDITION

Pearson International Edition

Deitel® Ser

How To Program Series

Java How to Program, 8/E

C++ How to Program, 6/E

Visual C++® 2008 How to Program, 2/E

C How to Program, 5/E

Internet & World Wide Web How to Program, 4/E

Visual Basic® 2008 How to Program

Visual C#® 2008 How to Program, 3/E

Small Java™ How to Program, 6/E

Small C++ How to Program, 5/E

Simply Series

Simply C++: An Application-Driven Tutorial Approach

Simply Java™ Programming: An Application-Driven Tutorial Approach

Simply C#: An Application-Driven Tutorial Approach

Simply Visual Basic® 2008, 3/E: An Application-Driven Tutorial Approach

CourseSmart Web Books

`www.deitel.com/books/CourseSmart.html`

C++ How to Program, 5/E & 6/E

Java How to Program, 6/E, 7/E & 8/E

Simply C++: An Application-Driven Tutorial Approach

Simply Visual Basic 2008: An Application-Driven Tutorial Approach, 3/E

Small C++ How to Program, 5/E

Small Java How to Program, 6/E

Visual Basic 2008 How to Program

Visual C# 2008 How to Program, 3/E

ies Page

Deitel® Developer Series

AJAX, Rich Internet Applications and Web Development for Programmers

C++ for Programmers

C# 2008 for Programmers, 3/E

Java for Programmers

Javascript for Programmers

LiveLessons Video Learning Products

`www.deitel.com/books/LiveLessons/`

Java Fundamentals Parts 1 and 2

C# Fundamentals Parts 1 and 2

C++ Fundamentals Parts 1 and 2

JavaScript Fundamentals Parts 1 and 2

To follow the Deitel publishing program, please register for the free *Deitel® Buzz Online* e-mail newsletter at:

`www.deitel.com/newsletter/subscribe.html`

To communicate with the authors, send e-mail to:

`deitel@deitel.com`

For information on government and corporate *Dive-Into® Series* on-site seminars offered by Deitel & Associates, Inc. worldwide, visit:

`www.deitel.com/training/`

or write to

`deitel@deitel.com`

For continuing updates on Prentice Hall/Deitel publications visit:

`www.deitel.com`
`www.prenhall.com/deitel`

Check out our Resource Centers for valuable web resources that will help you master Java, other important programming languages, software and Internet- and web-related topics:

`www.deitel.com/ResourceCenters.html`

If you purchased this book within the United States or Canada you should be aware that it has been wrongfully imported without the approval of the Publisher or the Author.

Vice President and Editorial Director, ECS: ***Marcia J. Horton***
Associate Editor: ***Carole Snyder***
Supervisor/Editorial Assistant: ***Dolores Mars***
Director of Team-Based Project Management: ***Vince O'Brien***
Senior Managing Editor: ***Scott Disanno***
Managing Editor: ***Robert Engelhardt***
A/V Production Editor: ***Greg Dulles***
Art Director: ***Kristine Carney***
Cover Design: ***Abbey S. Deitel, Harvey M. Deitel, Francesco Santalucia, Kristine Carney***
Interior Design: ***Harvey M. Deitel, Kristine Carney***
Manufacturing Manager: ***Alexis Heydt-Long***
Manufacturing Buyer: ***Lisa McDowell***
Director of Marketing: ***Margaret Waples***
Marketing Manager: ***Erin Davis***

The authors and publisher of this book have used their best efforts in preparing this book. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The authors and publisher make no warranty of any kind, expressed or implied, with regard to these programs or to the documentation contained in this book. The authors and publisher shall not be liable in any event for incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks and registered trademarks. Where those designations appear in this book, and Prentice Hall and the authors were aware of a trademark claim, the designations have been printed in initial caps or all caps. All product names mentioned remain trademarks or registered trademarks of their respective owners.

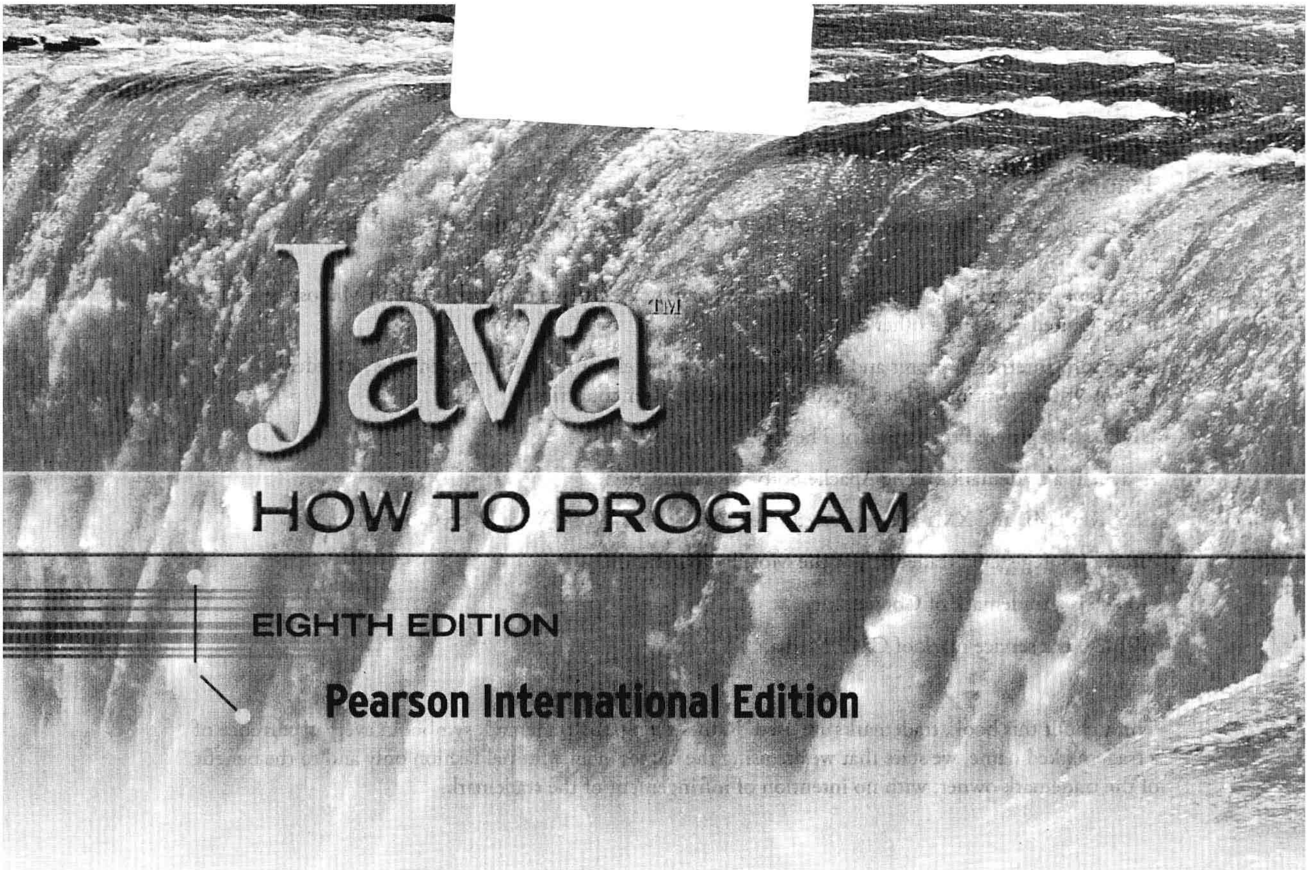
Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

ISBN-10: 0-13-136483-9

ISBN-13: 978-0-13-136483-7

Pearson Education Ltd., *London*
Pearson Education Australia Pty. Ltd., *Sydney*
Pearson Education Singapore, Pte. Ltd.
Pearson Education North Asia Ltd., *Hong Kong*
Pearson Education Canada, Inc., *Toronto*
Pearson Educación de Mexico, S.A. de C.V.
Pearson Education–Japan, *Tokyo*
Pearson Education Malaysia, Pte. Ltd.
Pearson Education, Inc., *Upper Saddle River, New Jersey*



Java™

HOW TO PROGRAM

EIGHTH EDITION

Pearson International Edition

Paul Deitel
Deitel & Associates, Inc.

Harvey Deitel
Deitel &-Associates, Inc.

DEITEL®

PEARSON
Prentice Hall

Upper Saddle River, New Jersey 07458

Trademarks

In memory of

Kristen Nygaard, co-inventor of Simula, the world's first object-oriented programming language.

Paul and Harvey Deitel

Deitel Resource Centers

Our Resource Centers focus on the vast amounts of free content available online. Find resources, downloads, tutorials, documentation, books, e-books, journals, articles, blogs, RSS feeds and more on many of today's hottest programming and technology topics. For the most up-to-date list of our Resource Centers, visit:

```
www.deitel.com/ResourceCenters.html
```

Let us know what other Resource Centers you'd like to see! Also, please register for the free *Deitel® Buzz Online* e-mail newsletter at:

```
www.deitel.com/newsletter/subscribe.html
```

Computer Science

Functional Programming
Regular Expressions

Programming

ASP.NET 3.5
Adobe Flex
Ajax
Apex
ASP.NET Ajax
ASP.NET
C
C++
C++ Boost Libraries
C++ Game Programming
C#
Code Search Engines and Code Sites
Computer Game Programming
CSS 2.1
Dojo
Facebook Developer Platform
Flash 9
Functional Programming
Java
Java Certification and Assessment Testing
Java Design Patterns
Java EE 5
Java SE 6
Java SE 7 (Dolphin) Resource Center
JavaFX
JavaScript
JSON
Microsoft LINQ
Microsoft Popfly
.NET
.NET 3.0
.NET 3.5
OpenGL
Perl
PHP
Programming Projects
Python
Regular Expressions
Ruby
Ruby on Rails
Silverlight
Visual Basic
Visual C++
Visual Studio Team System
Web 3D Technologies
Web Services
Windows Presentation Foundation
XHTML
XML

Games and Game Programming

Computer Game Programming
Computer Games
Mobile Gaming
Sudoku

Internet Business

Affiliate Programs
Competitive Analysis
Facebook Social Ads
Google AdSense
Google Analytics
Google Services
Internet Advertising
Internet Business Initiative
Internet Public Relations
Link Building
Location-Based Services
Online Lead Generation
Podcasting
Search Engine Optimization
Selling Digital Content
Sitemaps
Web Analytics
Website Monetization
YouTube and AdSense

Java

Java
Java Certification and Assessment Testing
Java Design Patterns
Java EE 5
Java SE 6
Java SE 7 (Dolphin) Resource Center
JavaFX

Microsoft

ASP.NET
ASP.NET 3.5
ASP.NET Ajax
C#
DotNetNuke (DNN)
Internet Explorer 7 (IE7)
Microsoft LINQ
.NET
.NET 3.0
.NET 3.5
SharePoint
Silverlight
Visual Basic
Visual C++
Visual Studio Team System
Windows Presentation Foundation
Windows Vista
Microsoft Popfly

Open Source & LAMP Stack

Apache
DotNetNuke (DNN)
Eclipse
Firefox
Linux
MySQL
Open Source
Perl
PHP
Python
Ruby

Software

Apache
DotNetNuke (DNN)
Eclipse
Firefox
Internet Explorer 7 (IE7)
Linux
MySQL
Open Source
Search Engines
SharePoint
Skype
Web Servers
Wikis
Windows Vista

Web 2.0

Alert Services
Attention Economy
Blogging
Building Web Communities
Community Generated Content
Facebook Developer Platform
Facebook Social Ads
Google Base
Google Video
Google Web Toolkit (GWT)
Internet Video
Joost
Location-Based Services
Mashups
Microformats
Recommender Systems
RSS
Social Graph
Social Media
Social Networking
Software as a Service (SaaS)
Virtual Worlds
Web 2.0
Web 3.0
Widgets

Dive Into® Web 2.0 eBook

Web 2 eBook

Other Topics

Computer Games
Computing Jobs
Gadgets and Gizmos
Ring Tones
Sudoku

Preface

Live in fragments no longer, only connect.
—Edgar Morgan Foster

Welcome to Java and *Java How to Program, Eighth Edition*! This book presents leading-edge computing technologies for students, instructors, software developers and IT professionals.

We use the Deitel signature "live-code approach," presenting most concepts in the context of complete working Java programs, rather than using code snippets. Each code example is immediately followed by one or more sample executions. All the source code is available at `www.deitel.com/books/jhtp8/`.

At Deitel & Associates, we write programming-language textbooks and professional books for Pearson/Prentice Hall, deliver corporate training courses worldwide and develop Web 2.0 Internet businesses. We have updated the previous edition of this book based on recent changes to the Java language and the evolving preferred ways of teaching and learning programming. All of the chapters have been significantly tuned.

New and Updated Features

Here are the updates we've made for *Java How to Program, 8/e*:

- The book has a new interior design that graphically organizes, clarifies and highlights the information and enhances the book's pedagogy.
- We updated the entire book to Java Standard Edition 6 Update 11 and carefully audited the manuscript against the *Java Language Specification.*
- We added the "Making a Difference" exercises set: Students *want* to make a difference. We're encouraging them to associate computers and the Internet with solving problems that really matter to individuals, communities, countries and the world. We hope that our new exercises encourage students to think for themselves as they explore complex social issues. These exercises are not intended to make a political statement. They are meant to increase awareness of important issues the world is facing. Students should approach these issues in the context of their own values, politics and beliefs. Many of the new exercises require students to do research on the web—and weave the results into their problem-solving process. Here's a list of the 34 new "Making a Difference" exercises:

  Test Drive: Carbon Footprint Calculator
  Test Drive: Body Mass Index Calculator
  Attributes of Hybrid Vehicles
  Gender Neutrality
  Body Mass Index Calculator
  World Population Growth Calculator
  Car Pool Savings Calculator
  Target Heart Rate Calculator
  Computerization of Health Records
  Enforcing Privacy with Cryptography
  World Population Growth
  Global Warming Facts Quiz
  Tax Plan Alternatives; The "Fair Tax"
  Computer-Assisted Instruction
  Computer-Assisted Instruction: Reducing Student Fatigue
  Computer-Assisted Instruction: Monitoring Student Performance
  Computer Assisted Instruction: Difficulty Levels
  Computer-Assisted Instruction: Varying the Types of Problems
  Polling
  Air Traffic Control
  Carbon Footprint Interface: Polymorphism
  Ecofont
  Typing Tutor: Tuning a Crucial Skill in the Computer Age
  Large Type Displays for People with Low Vision
  Cooking with Healthier Ingredients
  Spam Scanner
  Phishing Scanner
  Accessibility Project: Speech Synthesis

Accessibility Project: Speech Recognition
Project: Simbad Robotics Simulator
SPAM Scanner Web Service
SMS Web Service
Gender Neutrality Web Service

- We tuned the optional Object-Oriented Design/UML 2 automated teller machine (ATM) case study and reorganized it into two optional chapters (12 and 13) that present the ATM's design and complete code implementation. The ATM is a nice business example that all students can relate to. In our experience, teaching these two chapters as a unit helps students tie together many of the object-oriented concepts they learn in Chapters 1 – 10. A key concept in object-oriented programming is the interactions among objects. In most programming textbooks, the code examples create and use one or two objects. The ATM gives students the opportunity to study interactions of *many* objects that provide the functionality of a substantial system. Chapters 12 and 13 provide complete solutions to *all* of their exercises. Previously, the case study was distributed through Chapters 2 – 8, 10 and an appendix. For instructors who wish to cover the case study in a distributed manner, for each section in Chapters 12 and 13, we indicate after which early chapter that section can be covered.
- We reinforced our early classes and objects pedagogy, paying careful attention to the guidance of the college instructors on our review teams to ensure that we got the conceptual level right. The treatment of OOP is clear and accessible. We introduce the basic concepts and terminology of object technology in Chapter 1. Students develop their first customized classes and objects in Chapter 3. Presenting objects and classes early gets students "thinking about objects" immediately and mastering these concepts more thoroughly.
- We reordered our presentation of data structures. We now begin with generic class `ArrayList` in Chapter 7. Because students will understand basic generics concepts so early in the book, our later data structures discussions provide a deeper treatment of generic collections—showing how to use the built-in collections of the Java API. We then show how to implement generic methods and classes. Finally, we show how to build custom generic data structures.
- We added coverage of Java Web Start and the Java Network Launch Protocol (JNLP), which enable *both* applets *and* applications to be launched via a web browser. In addition, the user can install them as shortcuts on the desktop to execute them in the future without revisiting the website. Programs can also request the user's permission to access local system resources such as files—enabling you to develop more robust applets and applications that execute safely using Java's sandbox security model, which applies to downloaded code.
- We reordered several chapters to facilitate teaching the book in modules. The dependencies chart (page xxx) was updated to reflect the new modularization.
- We've added many links to online documentation where students can learn more about a class or topic. And we've added many links to the Deitel Java-related Resource Centers available at `www.deitel.com/ResourceCenters.html`.
- Chapter 7 now covers class `Arrays`—which contains methods for performing common array manipulations—and class `ArrayList`—which implements a dynamically resizable array-like data structure. This follows our philosophy of *using existing classes before learning how to define your own classes*.
- We now introduce class `BigInteger` for arbitrarily large integer values in Chapter 18, Recursion.
- We carefully tuned all the chapters with a focus on increasing clarity and simplicity, eliminating redundancy, reducing page count (this new edition is 90 in-book pages shorter than the previous one), and improving pedagogy and modular organization.
- We've replaced all uses of `StringTokenizer` with the recommended `String` method `split` throughout the book. Class `StringTokenizer` is still discussed, primarily for backward compatibility with legacy code.
- We include an alphabetized list of the important terms defined in each chapter with the page number of the term's defining occurrence. Defining occurrences are also highlighted in the index with a bold, maroon page number.

All of this has been carefully reviewed by 24 distinguished academics and industry developers who worked with us on *Java How to Program, 8/e*.

We believe that this book and its support materials will give students and professionals an informative, interesting, challenging and entertaining Java educational experience. We provide a suite of ancillary materials that will help instructors maximize their students' learning experience.

As you read the book, if you have questions, send an e-mail to deitel@deitel.com; we'll respond promptly. For updates on this book and the status of all supporting Java software, and for the latest news on all Deitel publications and services, visit www.deitel.com. Sign up at www.deitel.com/newsletter/subscribe.html for the free *Deitel® Buzz Online* e-mail newsletter, and check out our growing list of Java and related Resource Centers at www.deitel.com/ResourceCenters.html. Each week we announce our latest Resource Centers in the newsletter.

Other Features

Other features of *Java How to Program, 8/e*, include:

- We audited the presentation against the ACM/IEEE curriculum recommendations and the Computer Science Advanced Placement Examination.
- The early classes and objects presentation features Time, Employee and GradeBook class case studies that weave their way through multiple sections and chapters, gradually introducing deeper OO concepts.
- Instructors teaching introductory courses have a broad choice of the amount of GUI and graphics to cover—from none, to a ten-brief-sections introductory sequence, to a deep treatment in Chapters 14, 15 and 25, and Appendix I.
- Our object-oriented programming and design presentations use the *UML(Unified Modeling Language)*—the industry-standard graphical language for modeling object-oriented systems.
- We provide several substantial object-oriented web programming case studies.
- Chapter 28, Accessing Databases with JDBC, covers JDBC 4 and uses the Java DB/Apache Derby and MySQL database management systems. The chapter features an OO case study on developing a database-driven address book that demonstrates prepared statements and JDBC 4's automatic driver discovery.
- Chapter 29, JavaServer Faces Web Applications, and Chapter 30, Ajax-Enabled JavaServer Faces Web Applications, introduce JavaServer Faces (JSF) technology and use itwith Netbeans 6.5 to build web applications quickly and easily. Chapter 29 includes examples on building web application GUIs, handling events, validating forms and session tracking. Chapter 30 discusses developing Ajax-enabled web applications, using JavaServer Faces technology. The chapter features a database-driven multitier web address book application that allows users to add and search for contacts. This Ajax-enabled application gives the reader a nice sense of Web 2.0 software development. The application uses Ajax-enabled JSF components to suggest contact names while the user types a name to locate.
- Chapter 31, Web Services, uses a tools-based approach to creating and consuming SOAP-*and* REST-based web services. Case studies include developing blackjack and airline reservation web services.
- We use a new tools-based approach for rapidly developing web applications; all the tools are available free for download.
- We provide 100+ Resource Centers (www.deitel.com/resourcecenters.html) to support our academic and professional readers. Their topics include Java SE 6, Java, Java Assessment and Certification, Java Design Patterns, Java EE 5, Code Search Engines and Code Sites, Game Programming, Programming Projects and many more. Sign up at www.deitel.com/newsletter/subscribe.html for the free *Deitel® Buzz Online* e-mail newsletter—each week we announce our latest Resource Center(s) and include other items of interest to our readers.
- We discuss key software engineering community concepts, such as Web 2.0, Ajax, SaaS (Software as a Service), web services, open-source software, design patterns, mashups, refactoring, agile software development, rapid prototyping and more.
- We completely reworked Chapter 26, Multithreading [special thanks to Brian Goetz and Joseph Bowbeer—co-authors of *Java Concurrency in Practice*, Addison-Wesley, 2006].
- We discuss the SwingWorker class for developing multithreaded user interfaces.
- We discuss the GroupLayout layout manager in the context of the GUI design tool in the NetBeans IDE.
- We present JTable sorting and filtering capabilities which allow the user to resort the data in a JTable and filter it by regular expressions.

- We discuss the `StringBuilder` class, which performs better than `StringBuffer` in non-threaded applications.
- We present annotations, which greatly reduce the amount of code you have to write to build applications.

Optional Case Study: Using the UML 2 to Develop an Object-Oriented ATM Design

UML 2 has become the preferred graphical modeling language for designing object-oriented systems. We use UML activity diagrams (in preference to flowcharts) to demonstrate the flow of control in each of Java's control statements, and we use UML class diagrams to visually represent classes and their inheritance relationships.

We include an optional (but highly recommended) case study on object-oriented design using the UML. The case study has been reviewed through many editions by a distinguished team of OOD/UML academics and industry professionals, including leaders in the field from Rational (the creators of the UML) and the Object Management Group (responsible for evolving the UML). In the case study, we design and fully implement the software for a simple automated teller machine (ATM). The optional Software Engineering Case Study in Chapters 12 and 13 presents a carefully paced introduction to object-oriented design using the UML.

We introduce a simple, concise subset of the UML 2, then guide the reader through a first design experience intended for the novice. The case study is not an exercise; rather, it's an end-to-end learning experience that concludes with a detailed walkthrough of the complete Java code. Chapters 12 and 13 help students develop an object-oriented design to complement the object-oriented programming concepts they've learned in Chapters 1 through 11.

At the end of Chapter 1, we introduce basic concepts and terminology of OOD. In Chapter 12, we consider more substantial issues, as we undertake a challenging problem with the techniques of OOD. We analyze a typical requirements document that specifies a system to be built, determine the objects needed to implement that system, determine the attributes these objects need to have, determine the behaviors these objects need to exhibit, and specify how the objects must interact with one another to meet the system requirements. In Chapter 13, we include a complete Java code implementation of the object-oriented system that we designed in Chapter 12.

This case study helps prepare students for the kinds of substantial projects they'll encounter in industry. We employ a carefully developed, incremental object-oriented design process to produce a UML 2 model for our ATM system. From this design, we produce a substantial working Java implementation using key object-oriented programming notions, including classes, objects, encapsulation, visibility, composition, inheritance and polymorphism.

Dependency Chart

The chart on the next page shows the dependencies among the chapters to help instructors plan their syllabi. *Java How to Program, 8/e*, is appropriate for a variety of programming courses at various levels, most notably CS 1 and CS 2 courses and introductory course sequences in related disciplines. The book has a clearly delineated, modular organization. Chapters 1 – 11 and 14 – 17 form an accessible elementary programming sequence with a solid introduction to object-oriented programming. Optional Chapters 12 – 13 form an accessible introduction to object-oriented design with the UML. The GUI and Graphics Track and Chapters 14, 15, 23, 24 and 25 form a substantial GUI, graphics and multimedia sequence. Chapters 18 – 22 form a nice data-structures sequence. Chapters 26 – 27 form a solid introduction to multithreading and Internet networking. Chapters 28 – 31 form a clear database-intensive web development sequence.

Syllabus Assistance

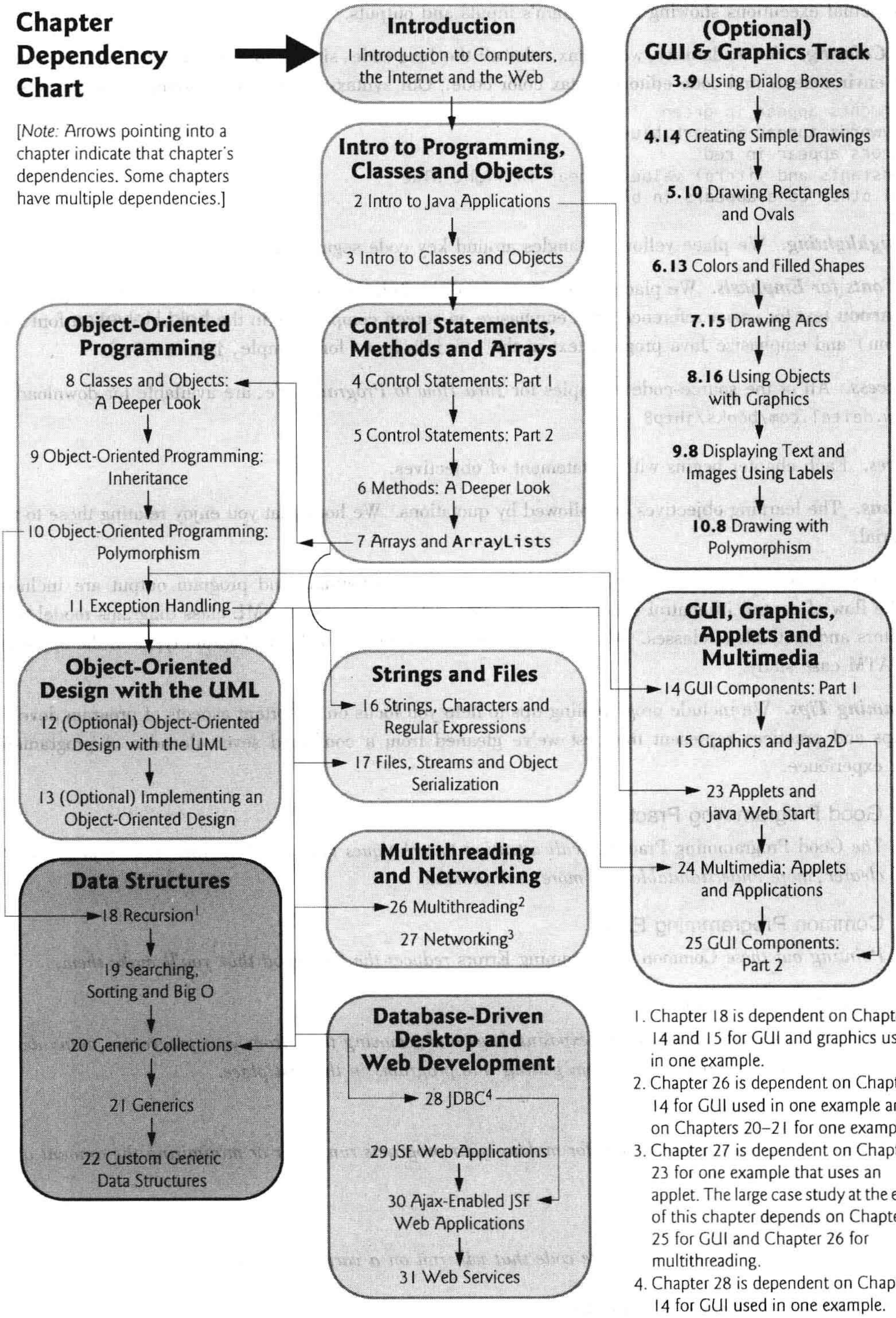
We're happy to assist instructors designing syllabi based on *Java How to Program, 8/e*. You can reach us by email (`deitel@deitel.com`) or phone (1 978 823-0130). We'll respond promptly.

Teaching Approach

Java How to Program, 8/e, contains a rich collection of examples. The book concentrates on the principles of good software engineering and stresses program clarity. We teach by example. We are educators who teach leading-edge programming languages and software-related topics in government, industry, military and academic classrooms worldwide.

Chapter Dependency Chart

[*Note:* Arrows pointing into a chapter indicate that chapter's dependencies. Some chapters have multiple dependencies.]


Introduction
1 Introduction to Computers, the Internet and the Web
Intro to Programming, Classes and Objects
2 Intro to Java Applications
3 Intro to Classes and Objects
Control Statements, Methods and Arrays
4 Control Statements: Part 1
5 Control Statements: Part 2
6 Methods: A Deeper Look
7 Arrays and ArrayLists
Object-Oriented Programming
8 Classes and Objects: A Deeper Look
9 Object-Oriented Programming: Inheritance
10 Object-Oriented Programming: Polymorphism
11 Exception Handling
Object-Oriented Design with the UML
12 (Optional) Object-Oriented Design with the UML
13 (Optional) Implementing an Object-Oriented Design
Data Structures
18 Recursion[1]
19 Searching, Sorting and Big O
20 Generic Collections
21 Generics
22 Custom Generic Data Structures
Strings and Files
16 Strings, Characters and Regular Expressions
17 Files, Streams and Object Serialization
Multithreading and Networking
26 Multithreading[2]
27 Networking[3]
Database-Driven Desktop and Web Development
28 JDBC[4]
29 JSF Web Applications
30 Ajax-Enabled JSF Web Applications
31 Web Services
(Optional) GUI & Graphics Track
3.9 Using Dialog Boxes
4.14 Creating Simple Drawings
5.10 Drawing Rectangles and Ovals
6.13 Colors and Filled Shapes
7.15 Drawing Arcs
8.16 Using Objects with Graphics
9.8 Displaying Text and Images Using Labels
10.8 Drawing with Polymorphism
GUI, Graphics, Applets and Multimedia
14 GUI Components: Part 1
15 Graphics and Java2D
23 Applets and Java Web Start
24 Multimedia: Applets and Applications
25 GUI Components: Part 2

1. Chapter 18 is dependent on Chapters 14 and 15 for GUI and graphics used in one example.
2. Chapter 26 is dependent on Chapter 14 for GUI used in one example and on Chapters 20–21 for one example.
3. Chapter 27 is dependent on Chapter 23 for one example that uses an applet. The large case study at the end of this chapter depends on Chapter 25 for GUI and Chapter 26 for multithreading.
4. Chapter 28 is dependent on Chapter 14 for GUI used in one example.

Live-Code Approach. *Java How to Program, 8/e*, is loaded with "live-code" examples. By this we mean that each new concept is presented in the context of a complete working Java application, followed immediately by one or more actual executions showing the program's inputs and outputs.

Syntax Coloring. For readability, we syntax color all the Java code, similar to the way most Java integrated-development environments and code editors syntax color code. Our syntax-coloring conventions are as follows:

```
comments appear in green
keywords appear in dark blue
errors appear in red
constants and literal values appear in light blue
all other code appears in black
```

Code Highlighting. We place yellow rectangles around key code segments.

Using Fonts for Emphasis. We place the key terms and the index's page reference for each defining occurrence in **bold maroon** text for easier reference. We emphasize on-screen components in the **bold Helvetica** font (e. g., the **File** menu) and emphasize Java program text in the `Lucida` font (for example, `int x = 5;`).

Web Access. All of the source-code examples for *Java How to Program, 8/e*, are available for download from:

`www.deitel.com/books/jhtp8`

Objectives. Each chapter begins with a statement of objectives.

Quotations. The learning objectives are followed by quotations. We hope that you enjoy relating these to the chapter material.

Illustrations/Figures. Abundant charts, tables, line drawings, programs and program output are included. We model the flow of control in control statements with UML activity diagrams. UML class diagrams model the fields, constructors and methods of classes. We make extensive use of six major UML diagram types in the optional OOD/UML 2 ATM case study.

Programming Tips. We include programming tips to help you focus on important aspects of program development. These tips and practices represent the best we've gleaned from a combined seven decades of programming and teaching experience.

Good Programming Practice

The Good Programming Practices *call attention to techniques that will help you produce programs that are clearer, more understandable and more maintainable.*

Common Programming Error

Pointing out these Common Programming Errors *reduces the likelihood that you'll make them.*

Error-Prevention Tip

These tips contain suggestions for exposing bugs and removing them from your programs; many describe aspects of Java that prevent bugs from getting into programs in the first place.

Performance Tip

These tips highlight opportunities for making your programs run faster or minimizing the amount of memory that they occupy.

Portability Tip

The Portability Tips *help you write code that will run on a variety of platforms.*

Software Engineering Observation

The Software Engineering Observations *highlight architectural and design issues that affect the construction of software systems, especially large-scale systems.*