“The chief merit of language is clearness…”
—Galén

Welcome to the Java programming language and Java How to Program, Tenth Edition! This book presents leading-edge computing technologies for students, instructors and software developers. It’s appropriate for introductory academic and professional course sequences based on the curriculum recommendations of the ACM and the IEEE, and for AP Computer Science exam preparation. If you haven’t already done so, please read the back cover and inside back cover—these concisely capture the essence of the book. In this Preface we provide more detail.

We focus on software engineering best practices. At the heart of the book is the Deitel signature “live-code approach”—rather than using code snippets, we present concepts in the context of complete working programs that run on recent versions of Windows®, OS X® and Linux®. Each complete code example is accompanied by live sample executions.

Keeping in Touch with the Authors
As you read the book, if you have questions, send an e-mail to us at
deitel@deitel.com
and we’ll respond promptly. For updates on this book, visit
http://www.deitel.com/books/jhtp10
subscribe to the Deitel® Buzz Online newsletter at
http://www.deitel.com/newsletter/subscribe.html
and join the Deitel social networking communities on
• Facebook® (http://www.deitel.com/deitel1fan)
• Twitter® (@deitel)
• Google+™ (http://google.com/+DeitelFan)
• YouTube® (http://youtube.com/DeitelTV)
• LinkedIn® (http://linkedin.com/company/deitel-&-associates)

Source Code and VideoNotes
All the source code is available at:
http://www.deitel.com/books/jhtp10
and at the book’s Companion Website (which also contains extensive VideoNotes):
http://www.pearsonhighered.com/deitel
Modular Organization

Java How to Program, 10/e, is appropriate for programming courses at various levels, most notably CS 1 and CS 2 courses and introductory course sequences in related disciplines. The book’s modular organization helps instructors plan their syllabi:

Introduction
- Chapter 1, Introduction to Computers, the Internet and Java
- Chapter 2, Introduction to Java Applications; Input/Output and Operators
- Chapter 3, Introduction to Classes, Objects, Methods and Strings

Additional Programming Fundamentals
- Chapter 4, Control Statements: Part 1; Assignment, ++ and -- Operators
- Chapter 5, Control Statements: Part 2; Logical Operators
- Chapter 6, Methods: A Deeper Look
- Chapter 7, Arrays and ArrayLists
- Chapter 14, Strings, Characters and Regular Expressions
- Chapter 15, Files, Streams and Object Serialization

Object-Oriented Programming and Object-Oriented Design
- Chapter 8, Classes and Objects: A Deeper Look
- Chapter 9, Object-Oriented Programming: Inheritance
- Chapter 10, Object-Oriented Programming: Polymorphism and Interfaces
- Chapter 11, Exception Handling: A Deeper Look
- (Online) Chapter 33, ATM Case Study, Part 1: Object-Oriented Design with the UML
- (Online) Chapter 34, ATM Case Study Part 2: Implementing an Object-Oriented Design

Swing Graphical User Interfaces and Java 2D Graphics
- Chapter 12, GUI Components: Part 1
- Chapter 13, Graphics and Java 2D
- Chapter 22, GUI Components: Part 2

Data Structures, Collections, Lambdas and Streams
- Chapter 16, Generic Collections
- Chapter 17, Java SE 8 Lambdas and Streams
- Chapter 18, Recursion
- Chapter 19, Searching, Sorting and Big O
- Chapter 20, Generic Classes and Methods
- Chapter 21, Custom Generic Data Structures

1. The online chapters will be available on the book’s Companion Website for Fall 2014 classes.
**Concurrency; Networking**
- Chapter 23, Concurrency
- (Online) Chapter 28, Networking

**JavaFX Graphical User Interfaces, Graphics and Multimedia**
- Chapter 25, JavaFX GUI: Part 1
- (Online) Chapter 26, JavaFX GUI: Part 2
- (Online) Chapter 27, JavaFX Graphics and Multimedia

**Database-Driven Desktop and Web Development**
- Chapter 24, Accessing Databases with JDBC
- (Online) Chapter 29, Java Persistence API (JPA)
- (Online) Chapter 30, JavaServer™ Faces Web Apps: Part 1
- (Online) Chapter 31, JavaServer™ Faces Web Apps: Part 2
- (Online) Chapter 32, REST-Based Web Services

**New and Updated Features**
Here are the updates we’ve made for *Java How to Program, 10/e*:

**Java Standard Edition: Java SE 7 and the New Java SE 8**
- *Easy to use with Java SE 7 or Java SE 8.* To meet the needs of our audiences, we designed the book for college and professional courses based on Java SE 7, Java SE 8 or a mixture of both. The Java SE 8 features are covered in optional, easy-to-include-or-omit sections. The new Java SE 8 capabilities can dramatically improve the programming process. Figure 1 lists some new Java SE 8 features that we cover.

**Java SE 8 features**
- Lambda expressions
- Type-inference improvements
- @FunctionalInterface annotation
- Parallel array sorting
- Bulk data operations for Java Collections—filter, map and reduce
- Library enhancements to support lambdas (e.g., java.util.stream, java.util.function)
- Date & Time API (java.time)
- Java concurrency API improvements
- static and default methods in interfaces
- Functional interfaces—interfaces that define only one abstract method and can include static and default methods
- JavaFX enhancements

**Fig. 1**  Some new Java SE 8 features.
Java SE 8 lambdas, streams, and interfaces with default and static methods. The most significant new features in Java SE 8 are lambdas and complementary technologies, which we cover in detail in the optional Chapter 17 and optional sections marked “Java SE 8” in later chapters. In Chapter 17, you’ll see that functional programming with lambdas and streams can help you write programs faster, more concisely, more simply, with fewer bugs and that are easier to parallelize (to get performance improvements on multi-core systems) than programs written with previous techniques. You’ll see that functional programming complements object-oriented programming. After you read Chapter 17, you’ll be able to cleverly reimplement many of the Java SE 7 examples throughout the book (Fig. 2).

Java SE 7’s try-with-resources statement and the AutoClosable Interface. AutoClosable objects reduce the likelihood of resource leaks when you use them with the try-with-resources statement, which automatically closes the AutoClosable objects. In this edition, we use try-with-resources and AutoClosable objects as appropriate starting in Chapter 15, Files, Streams and Object Serialization.

Java security. We audited our book against the CERT Oracle Secure Coding Standard for Java as appropriate for an introductory textbook.

See the Secure Java Programming section of this Preface for more information about CERT.

Fig. 2 | Java SE 8 lambdas and streams discussions and examples.

<table>
<thead>
<tr>
<th>Pre-Java-SE-8 topics</th>
<th>Corresponding Java SE 8 discussions and examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 7, Arrays and ArrayLists</td>
<td>Sections 17.3–17.4 introduce basic lambda and streams capabilities that process one-dimensional arrays.</td>
</tr>
<tr>
<td>Chapter 10, Object-Oriented Programming: Polymorphism and Interfaces</td>
<td>Section 10.10 introduces the new Java SE 8 interface features (default methods, static methods and the concept of functional interfaces) that support functional programming with lambdas and streams.</td>
</tr>
<tr>
<td>Chapters 12 and 22, GUI Components: Part 1 and 2, respectively</td>
<td>Section 17.9 shows how to use a lambda to implement a Swing event-listener functional interface.</td>
</tr>
<tr>
<td>Chapter 14, Strings, Characters and Regular Expressions</td>
<td>Section 17.5 shows how to use lambdas and streams to process collections of String objects.</td>
</tr>
<tr>
<td>Chapter 15, Files, Streams and Object Serialization</td>
<td>Section 17.7 shows how to use lambdas and streams to process lines of text from a file.</td>
</tr>
<tr>
<td>Chapter 23, Concurrency</td>
<td>Shows that functional programs are easier to parallelize so that they can take advantage of multi-core architectures to enhance performance. Demonstrates parallel stream processing. Shows that Arrays method parallelSort improves performance on multi-core architectures when sorting large arrays.</td>
</tr>
<tr>
<td>Chapter 25, JavaFX GUI: Part 1</td>
<td>Section 25.5.5 shows how to use a lambda to implement a JavaFX event-listener functional interface.</td>
</tr>
</tbody>
</table>
New and Updated Features

- **Java NIO API.** We updated the file-processing examples in Chapter 15 to use features from the Java NIO (new IO) API.

- **Java Documentation.** Throughout the book, we provide links to Java documentation where you can learn more about various topics that we present. For Java SE 7 documentation, the links begin with

  http://docs.oracle.com/javase/7/

  and for Java SE 8 documentation, the links begin with

  http://download.java.net/jdk8/

  These links could change when Oracle releases Java SE 8—possibly to links beginning with

  http://docs.oracle.com/javase/8/

  For any links that change after publication, we’ll post updates at

  http://www.deitel.com/books/jhtp10

**Swing and JavaFX GUI, Graphics and Multimedia**

- **Swing GUI and Java 2D graphics.** Java’s Swing GUI is discussed in the optional GUI and graphics sections in Chapters 3–10 and in Chapters 12 and 22. Swing is now in maintenance mode—Oracle has stopped development and will provide only bug fixes going forward, however it will remain part of Java and is still widely used. Chapter 13 discusses Java 2D graphics.

- **JavaFX GUI, graphics and multimedia.** Java’s GUI, graphics and multimedia API going forward is JavaFX. In Chapter 25, we use JavaFX 2.2 (released in 2012) with Java SE 7. Our online Chapters 26 and 27—located on the book’s companion website (see the inside front cover of this book)—present additional JavaFX GUI features and introduce JavaFX graphics and multimedia in the context of Java FX 8 and Java SE 8. In Chapters 25–27 we use Scene Builder—a drag-and-drop tool for creating JavaFX GUIs quickly and conveniently. It’s a standalone tool that you can use separately or with any of the Java IDEs.

- **Scalable GUI and graphics presentation.** Instructors teaching introductory courses have a broad choice of the amount of GUI, graphics and multimedia to cover—from none at all, to optional introductory sections in the early chapters, to a deep treatment of Swing GUI and Java 2D graphics in Chapters 12, 13 and 22, and a deep treatment of JavaFX GUI, graphics and multimedia in Chapter 25 and online Chapters 26–27.

**Concurrency**

- **Concurrency for optimal multi-core performance.** In this edition, we were privileged to have as a reviewer Brian Goetz, co-author of *Java Concurrency in Practice* (Addison-Wesley). We updated Chapter 23, with Java SE 8 technology and idiom. We added a parallelSort vs. sort example that uses the Java SE 8 Date/Time API to time each operation and demonstrate parallelSort’s better performance on a multi-core system. We include a Java SE 8 parallel vs. sequential stream processing example, again using the Date/Time API to show performance improvements. Fi-
Finally, we added a Java SE 8 CompletableFuture example that demonstrates sequential and parallel execution of long-running calculations.

- **SwingWorker class.** We use class SwingWorker to create multithreaded user interfaces. In online Chapter 26, we show how JavaFX handles concurrency.

- **Concurrency is challenging.** Programming concurrent applications is difficult and error-prone. There’s a great variety of concurrency features. We point out the ones that most people should use and mention those that should be left to the experts.

### Getting Monetary Amounts Right

- **Monetary amounts.** In the early chapters, for convenience, we use type double to represent monetary amounts. Due to the potential for incorrect monetary calculations with type double, class BigDecimal (which is a bit more complex) should be used to represent monetary amounts. We demonstrate BigDecimal in Chapters 8 and 25.

### Object Technology

- **Object-oriented programming and design.** We use an early objects approach, introducing 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. [For courses that require a late-objects approach, consider Java How to Program, 10/e, Late Objects Version.]

- **Early objects real-world case studies.** The early classes and objects presentation features Account, Student, AutoPolicy, Time, Employee, GradeBook and Card shuffling-and-dealing case studies, gradually introducing deeper OO concepts.

- **Inheritance, Interfaces, Polymorphism and Composition.** We use a series of real-world case studies to illustrate each of these OO concepts and explain situations in which each is preferred in building industrial-strength applications.

- **Exception handling.** We integrate basic exception handling early in the book then present a deeper treatment in Chapter 11. Exception handling is important for building “mission-critical” and “business-critical” applications. Programmers need to be concerned with, “What happens when the component I call on to do a job experiences difficulty? How will that component signal that it had a problem?” To use a Java component, you need to know not only how that component behaves when “things go well,” but also what exceptions that component “throws” when “things go poorly.”

- **Class Arrays and ArrayList.** Chapter 7 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 getting lots of practice using existing classes while learning how to define your own classes. The chapter’s rich selection of exercises includes a substantial project on building your own computer through the technique of software simulation. Chapter 21 includes a follow-on project on building your own compiler that can compile high-level language programs into machine language code that will execute on your computer simulator.
New and Updated Features

- **Optional Online Case Study: Developing an Object-Oriented Design and Java Implementation of an ATM.** Online Chapters 33–34 include an optional case study on object-oriented design using the UML (Unified Modeling Language™)—the industry-standard graphical language for modeling object-oriented systems. We design and implement the software for a simple automated teller machine (ATM). We analyze a typical requirements document that specifies the system to be built. We determine the classes needed to implement that system, the attributes the classes need to have, the behaviors the classes need to exhibit and specify how the classes must interact with one another to meet the system requirements. From the design we produce a complete Java implementation. Students often report having a “light-bulb moment”—the case study helps them “tie it all together” and really understand object orientation.

- **Data Structures and Generic Collections**
  - **Data structures presentation.** We begin with generic class `ArrayList` in Chapter 7. Our later data structures discussions (Chapters 16–21) provide a deeper treatment of generic collections—showing how to use the built-in collections of the Java API. We discuss recursion, which is important for implementing tree-like, data-structure classes. We discuss popular searching and sorting algorithms for manipulating the contents of collections, and provide a friendly introduction to Big O—a means of describing how hard an algorithm might have to work to solve a problem. We then show how to implement generic methods and classes, and custom generic data structures (this is intended for computer-science majors—most programmers should use the pre-built generic collections). Lambdas and streams (introduced in Chapter 17) are especially useful for working with generic collections.

- **Database**
  - **JDBC.** Chapter 24 covers JDBC and uses the Java DB database management system. The chapter introduces Structured Query Language (SQL) and features an OO case study on developing a database-driven address book that demonstrates prepared statements.
  - **Java Persistence API.** The new online Chapter 29 covers the Java Persistence API (JPA)—a standard for object relational mapping (ORM) that uses JDBC “under the hood.” ORM tools can look at a database’s schema and generate a set of classes that enabled you to interact with a database without having to use JDBC and SQL directly. This speeds database-application development, reduces errors and produces more portable code.

- **Web Application Development**
  - **Java Server Faces (JSF).** Online Chapters 30–31 have been updated to introduce the latest JavaServer Faces (JSF) technology, which facilitates building JSF web-based applications. Chapter 30 includes examples on building web application GUIs, validating forms and session tracking. Chapter 31 discusses data-driven, Ajax-enabled JSF applications—the chapter features a database-driven multitier web address book that allows users to add and search for contacts.
  - **Web services.** Chapter 32 now concentrates on creating and consuming REST-based web services. The vast majority of today’s web services now use REST.
Secure Java Programming

It’s difficult to build industrial-strength systems that stand up to attacks from viruses, worms, and other forms of “malware.” Today, via the Internet, such attacks can be instantaneous and global in scope. Building security into software from the beginning of the development cycle can greatly reduce vulnerabilities. We incorporate various secure Java coding practices (as appropriate for an introductory textbook) into our discussions and code examples.

The CERT Coordination Center (www.cert.org) was created to analyze and respond promptly to attacks. CERT—the Computer Emergency Response Team—is a government-funded organization within the Carnegie Mellon University Software Engineering Institute™. CERT publishes and promotes secure coding standards for various popular programming languages to help software developers implement industrial-strength systems that avoid the programming practices which leave systems open to attack.

We’d like to thank Robert C. Seacord, Secure Coding Manager at CERT and an adjunct professor in the Carnegie Mellon University School of Computer Science. Mr. Seacord was a technical reviewer for our book, C How to Program, 7/e, where he scrutinized our C programs from a security standpoint, recommending that we adhere to the CERT C Secure Coding Standard. This experience influenced our coding practices in C++ How to Program, 9/e and Java How to Program, 10/e as well.

Optional GUI and Graphics Case Study

Students enjoy building GUI and graphics applications. For courses that introduce GUI and graphics early, we’ve integrated an optional 10-segment introduction to creating graphics and Swing-based graphical user interfaces (GUIs). The goal of this case study is to create a simple polymorphic drawing application in which the user can select a shape to draw, select the characteristics of the shape (such as its color) and use the mouse to draw the shape. The case study builds gradually toward that goal, with the reader implementing polymorphic drawing in Chapter 10, adding an event-driven GUI in Chapter 12 and enhancing the drawing capabilities in Chapter 13 with Java 2D.

- Section 3.6—Using Dialog Boxes
- Section 4.15—Creating Simple Drawings
- Section 5.11—Drawing Rectangles and Ovals
- Section 6.13—Colors and Filled Shapes
- Section 7.17—Drawing Arcs
- Section 8.16—Using Objects with Graphics
- Section 9.7—Displaying Text and Images Using Labels
- Section 10.11—Drawing with Polymorphism
- Exercise 12.17—Expanding the Interface
- Exercise 13.31—Adding Java2D

Teaching Approach

Java How to Program, 10/e, contains hundreds of complete working examples. We stress program clarity and concentrate on building well-engineered software.
Teaching Approach

VideoNotes. The Companion Website includes extensive VideoNotes in which co-author Paul Deitel explains in detail most of the programs in the book’s core chapters. Students like viewing the VideoNotes for reinforcement of core concepts and for additional insights.

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 can be downloaded from:

- http://www.deitel.com/books/jhtp10
- http://www.pearsonhighered.com/deitel

Objectives. The opening quotes are followed by a list of chapter objectives.

Illustrations/Figures. Abundant tables, line drawings, UML diagrams, programs and program outputs are included.

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.
Summary Bullets. We present a section-by-section bullet-list summary of the chapter. For ease of reference, we include the page number of each key term’s defining occurrence in the text.

Self-Review Exercises and Answers. Extensive self-review exercises and answers are included for self study. All of the exercises in the optional ATM case study are fully solved.

Exercises. The chapter exercises include:

- simple recall of important terminology and concepts
- What’s wrong with this code?
- What does this code do?
- writing individual statements and small portions of methods and classes
- writing complete methods, classes and programs
- major projects
- in many chapters, Making a Difference exercises that encourage you to use computers and the Internet to research and solve significant social problems.

Exercises that are purely SE 8 are marked as such. Check out our Programming Projects Resource Center for lots of additional exercise and project possibilities (www.deitel.com/ProgrammingProjects/).

Index. We’ve included an extensive index. Defining occurrences of key terms are highlighted with a bold maroon page number. The print book index mentions only those terms used in the print book. The online chapters index includes all the print book terms and the online chapter terms.

Software Used in Java How to Program, 10/e

All the software you’ll need for this book is available free for download from the Internet. See the Before You Begin section that follows this Preface for links to each download.

We wrote most of the examples in Java How to Program, 10/e, using the free Java Standard Edition Development Kit (JDK) 7. For the optional Java SE 8 modules, we used the OpenJDK’s early access version of JDK 8. In Chapter 25 and several online chapters, we also used the Netbeans IDE. See the Before You Begin section that follows this Preface for more information. You can find additional resources and software downloads in our Java Resource Centers at:

www.deitel.com/ResourceCenters.html
Instructor Supplements

The following supplements are available to qualified instructors only through Pearson Education’s Instructor Resource Center (www.pearsonhighered.com/irc):

- **PowerPoint® slides** containing all the code and figures in the text, plus bulleted items that summarize key points.
- **Test Item File** of multiple-choice questions (approximately two per book section).
- **Solutions Manual** with solutions to the vast majority of the end-of-chapter exercises. Before assigning an exercise for homework, instructors should check the IRC to be sure it includes the solution.

Please do not write to us requesting access to the Pearson Instructor’s Resource Center which contains the book’s instructor supplements, including the exercise solutions. Access is limited strictly to college instructors teaching from the book. Instructors may obtain access only through their Pearson representatives. Solutions are **not** provided for “project” exercises.

If you’re not a registered faculty member, contact your Pearson representative or visit www.pearsonhighered.com/educator/repllocator/.

Acknowledgments

We’d like to thank Abbey Deitel and Barbara Deitel for long hours devoted to this project. We’re fortunate to have worked on this project with the dedicated team of publishing professionals at Pearson. We appreciate the guidance, wisdom and energy of Tracy Johnson, Executive Editor, Computer Science. Tracy and her team handle all of our academic textbooks. Carole Snyder recruited the book’s academic reviewers and managed the review process. Bob Engelhardt managed the book’s publication. We selected the cover art and Laura Gardner designed the cover.

Reviewers

We wish to acknowledge the efforts of our recent editions reviewers—a distinguished group of academics, Oracle Java team members, Oracle Java Champions and other industry professionals. They scrutinized the text and the programs and provided countless suggestions for improving the presentation.

We appreciate the guidance of Jim Weaver and Johan Vos (co-authors of *Pro JavaFX 2*), and Simon Ritter on the three JavaFX chapters.

**Tenth Edition reviewers:** Lance Andersen (Oracle Corporation), Dr. Danny Coward (Oracle Corporation), Brian Goetz (Oracle Corporation), Evan Golub (University of Maryland), Dr. Huiwei Guan (Professor, Department of Computer & Information Science, North Shore Community College), Manfred Riem (Java Champion), Simon Ritter (Oracle Corporation), Robert C. Seacord (CERT, Software Engineering Institute, Carnegie Mellon University), Khallai Taylor (Assistant Professor, Triton College and Adjunct Professor, Lonestar College—Kingwood), Jorge Vargas (Yumbling and a Java Champion), Johan Vos (LodgON and Oracle Java Champion) and James L. Weaver (Oracle Corporation and author of *Pro JavaFX 2*).

**Previous editions reviewers:** Soundararajan Angusamy (Sun Microsystems), Joseph Bowbeer (Consultant), William E. Duncan (Louisiana State University), Diana Franklin
Preface

(University of California, Santa Barbara), Edward F. Gehringer (North Carolina State University), Ric Heishman (George Mason University), Dr. Heinz Kabutz (JavaSpecialists.eu), Patty Kraft (San Diego State University), Lawrence Premkumar (Sun Microsystems), Tim Margush (University of Akron), Sue McFarland Metzger (Villanova University), Shyamal Mitra (The University of Texas at Austin), Peter Pilgrim (Consultant), Manjeet Rege, Ph.D. (Rochester Institute of Technology), Susan Rodger (Duke University), Amr Sabry (Indiana University), José Antonio González Seco (Parliament of Andalusia), Sang Shin (Sun Microsystems), S. Sivakumar (Astra Infotech Private Limited), Raghavan "Rags" Srinivas (Intuit), Monica Sweat (Georgia Tech), Vinod Varma (Astra Infotech Private Limited) and Alexander Zuev (Sun Microsystems).

A Special Thank You to Brian Goetz

We were privileged to have Brian Goetz, Oracle's Java Language Architect and Specification Lead for Java SE 8's Project Lambda, and co-author of *Java Concurrency in Practice*, do a detailed full-book review. He thoroughly scrutinized every chapter, providing extremely helpful insights and constructive comments. Any remaining faults in the book are our own.

Well, there you have it! As you read the book, we'd appreciate your comments, criticisms, corrections and suggestions for improvement. Please address all correspondence to:

deitel@deitel.com

We'll respond promptly. We hope you enjoy working with *Java How to Program, 10/e*, as much as we enjoyed writing it!

Paul and Harvey Deitel

About the Authors

Paul Deitel, CEO and Chief Technical Officer of Deitel & Associates, Inc., is a graduate of MIT, where he studied Information Technology. He holds the Java Certified Programmer and Java Certified Developer designations, and is an Oracle Java Champion. Through Deitel & Associates, Inc., he has delivered hundreds of programming courses worldwide to clients, including Cisco, IBM, Siemens, Sun Microsystems, Dell, Fidelity, NASA at the Kennedy Space Center, the National Severe Storm Laboratory, White Sands Missile Range, Rogue Wave Software, Boeing, SunGard Higher Education, Nortel Networks, Puma, iRobot, Invensys and many more. He and his co-author, Dr. Harvey M. Deitel, are the world’s best-selling programming-language textbook/professional book/video authors.

Dr. Harvey Deitel, Chairman and Chief Strategy Officer of Deitel & Associates, Inc., has over 50 years of experience in the computer field. Dr. Deitel earned B.S. and M.S. degrees in Electrical Engineering from MIT and a Ph.D. in Mathematics from Boston University. He has extensive college teaching experience, including earning tenure and serving as the Chairman of the Computer Science Department at Boston College before founding Deitel & Associates, Inc., in 1991 with his son, Paul. The Deitels’ publications
have earned international recognition, with translations published in Japanese, German, Russian, Spanish, French, Polish, Italian, Simplified Chinese, Traditional Chinese, Korean, Portuguese, Greek, Urdu and Turkish. Dr. Deitel has delivered hundreds of programming courses to corporate, academic, government and military clients.

About Deitel® & Associates, Inc.

Deitel & Associates, Inc., founded by Paul Deitel and Harvey Deitel, is an internationally recognized authoring and corporate training organization, specializing in computer programming languages, object technology, mobile app development and Internet and web software technology. The company’s training clients include many of the world’s largest companies, government agencies, branches of the military, and academic institutions. The company offers instructor-led training courses delivered at client sites worldwide on major programming languages and platforms, including Java™, Android app development, Objective-C and iOS app development, C++, C, Visual C#®, Visual Basic®, Visual C++®, Python®, object technology, Internet and web programming and a growing list of additional programming and software development courses.

Through its 39-year publishing partnership with Pearson/Prentice Hall, Deitel & Associates, Inc., publishes leading-edge programming textbooks and professional books in print and a wide range of e-book formats, and LiveLessons video courses. Deitel & Associates, Inc. and the authors can be reached at:

deitel@deitel.com

To learn more about Deitel’s Dive-Into® Series Corporate Training curriculum, visit:

http://www.deitel.com/training

To request a proposal for worldwide on-site, instructor-led training at your organization, e-mail deitel@deitel.com.

Individuals wishing to purchase Deitel books and LiveLessons video training can do so through www.deitel.com. Bulk orders by corporations, the government, the military and academic institutions should be placed directly with Pearson. For more information, visit

