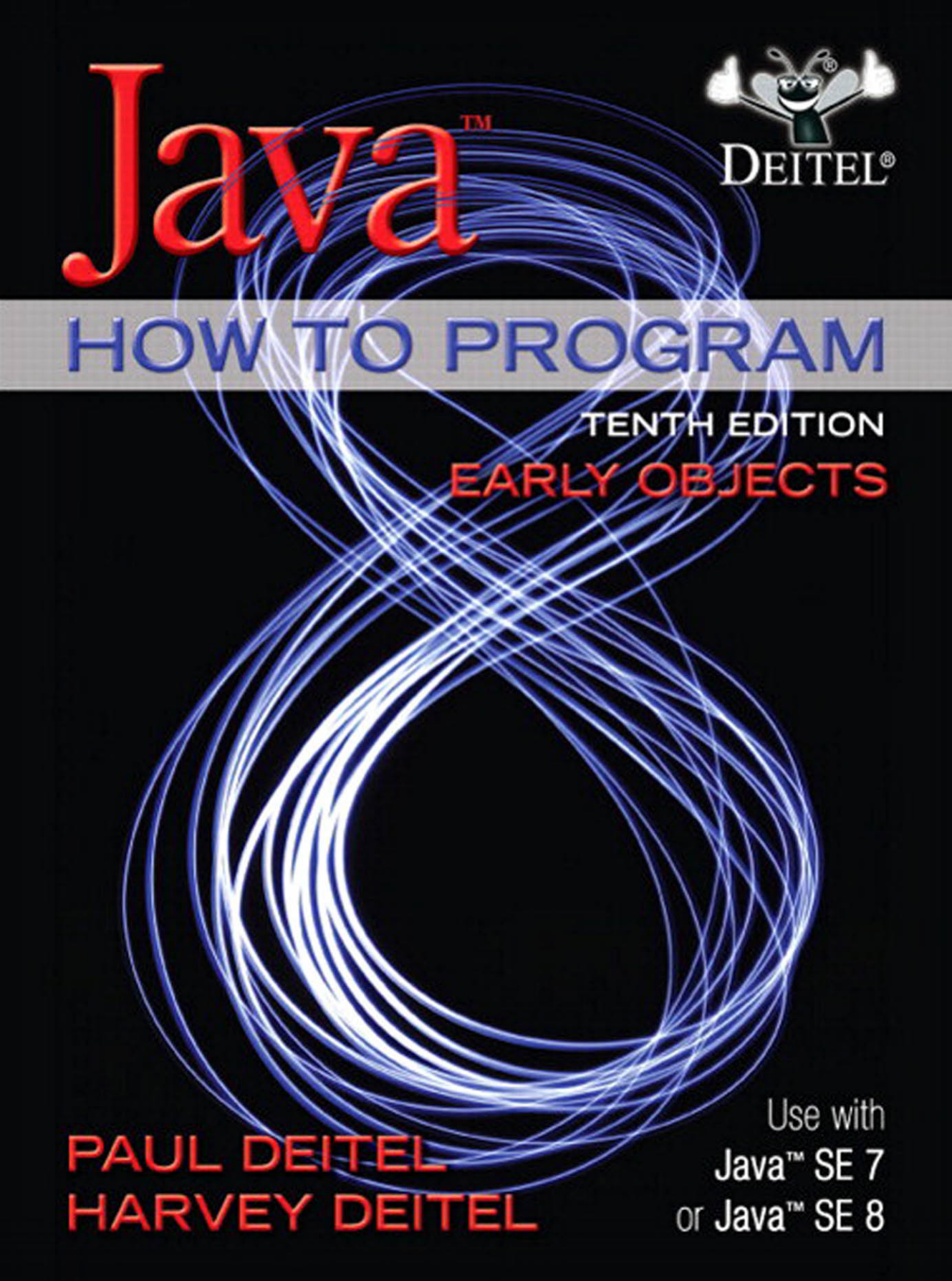

Java™
DEITEL®
HOW TO PROGRAM
TENTH EDITION
EARLY OBJECTS
PAUL DEITEL
HARVEY DEITEL
Use with
Java™ SE 7
or Java™ SE 8

ONLINE ACCESS

Thank you for purchasing a new copy of ***Java™ How to Program, Tenth Edition, Early Objects***. Your textbook includes 12 months of prepaid access to the book's Companion Website. This prepaid subscription provides you with full access to the following student support areas:

- VideoNotes (step-by-step video tutorials specifically designed to enhance the programming concepts presented in this textbook)
- Source code
- Premium web chapters and appendices

Use a coin to scratch off the coating and reveal your student access code.
Do not use a knife or other sharp object as it may damage the code.

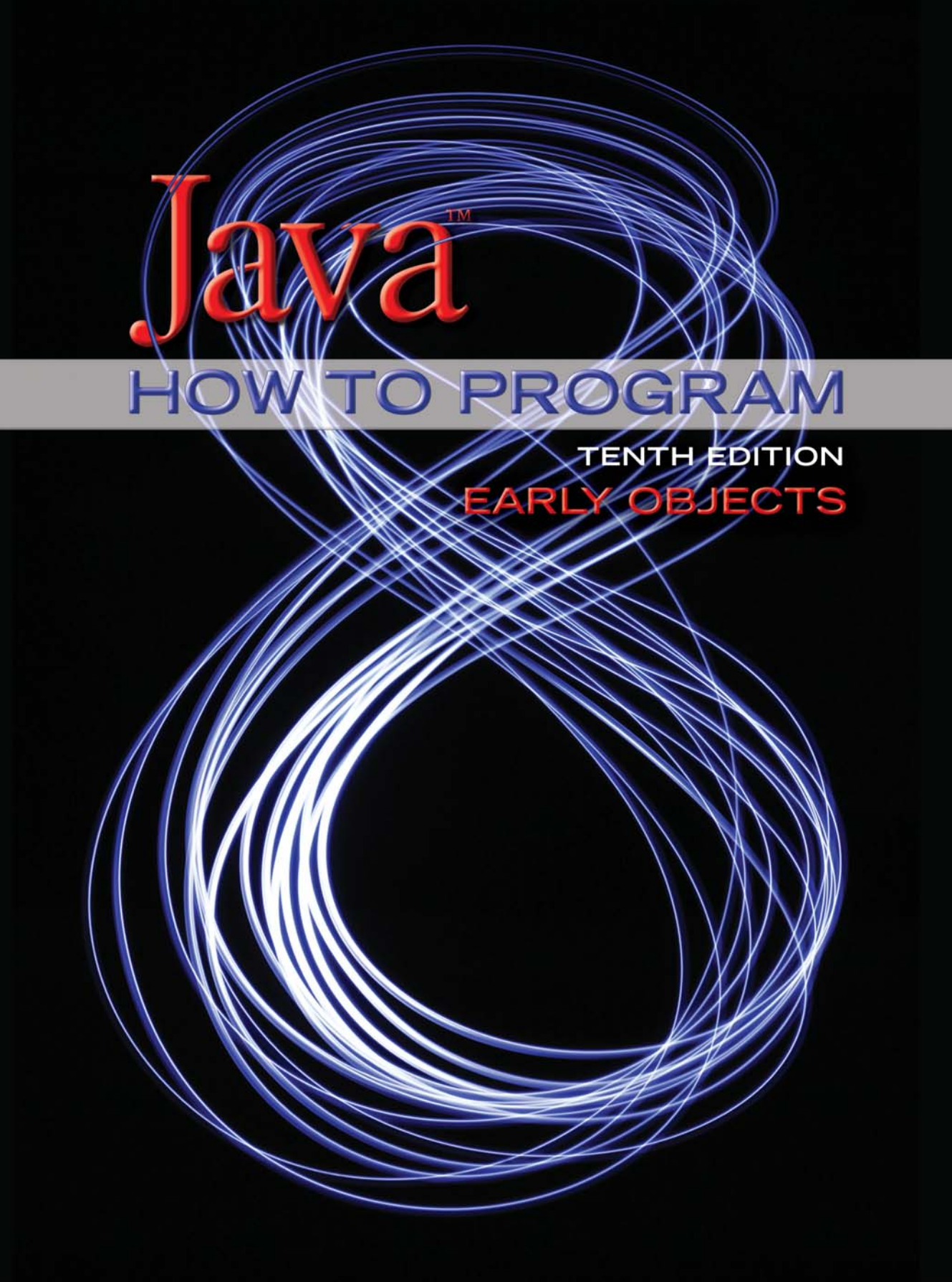
To access the *Java How to Program, Tenth Edition, Early Objects* Companion Website for the first time, you will need to register online using a computer with an Internet connection and a web browser. The process takes just a couple of minutes and only needs to be completed once.

1. Go to **http://www.pearsonhighered.com/deitel/**
2. Click on **Companion Website**.
3. Click on the **Register** button.
4. On the registration page, enter your student access code* found beneath the scratch-off panel. Do not type the dashes. You can use lower- or uppercase.
5. Follow the on-screen instructions. If you need help at any time during the online registration process, simply click the **Need Help?** icon.
6. Once your personal Login Name and Password are confirmed, you can begin using the *Java How to Program, Tenth Edition, Early Objects* Companion Website!

To log in after you have registered:

You only need to register for this Companion Website once. After that, you can log in any time at **http://www.pearsonhighered.com/deitel/** by providing your Login Name and Password when prompted.

*Important: The access code can only be used once. This subscription is valid for 12 months upon activation and is not transferable. If this access code has already been revealed, it may no longer be valid. If this is the case, you can purchase a subscription at **http://www.pearsonhighered.com/deitel/**, by going to the *Java How to Program, Tenth Edition, Early Objects* book and following the on-screen instructions.


Java™
HOW TO PROGRAM
TENTH EDITION
EARLY OBJECTS

Deitel® Series Page

How To Program Series

Android How to Program, 2/E
C++ How to Program, 9/E
C How to Program, 7/E
Java™ How to Program, 10/E
Java™ How to Program, Late Objects Version, 10/E
Internet & World Wide Web How to Program, 5/E
Visual C++® 2008 How to Program, 2/E
Visual Basic® 2012 How to Program, 6/E
Visual C#® 2012 How to Program, 5/E

Simply Series

Simply C++: An App-Driven Tutorial Approach
Simply Java™ Programming: An App-Driven Tutorial Approach
Simply C#: An App-Driven Tutorial Approach
Simply Visual Basic® 2010: An App-Driven Approach, 4/E

CourseSmart Web Books

www.deitel.com/books/CourseSmart/

C++ How to Program, 8/E and 9/E
Simply C++: An App-Driven Tutorial Approach
Java™ How to Program, 9/E and 10/E
Simply Visual Basic® 2010: An App-Driven Approach, 4/E

(continued from previous column)
Visual Basic® 2012 How to Program, 6/E
Visual Basic® 2010 How to Program, 5/E
Visual C#® 2012 How to Program, 5/E
Visual C#® 2010 How to Program, 4/E

Deitel® Developer Series

Android for Programmers: An App-Driven Approach, 2/E, Volume 1
C for Programmers with an Introduction to C11
C++11 for Programmers
C# 2012 for Programmers
Dive Into® iOS 6 for Programmers: An App-Driven Approach
Java™ for Programmers, 3/E
JavaScript for Programmers

LiveLessons Video Learning Products

www.deitel.com/books/LiveLessons/

Android App Development Fundamentals
C++ Fundamentals
Java™ Fundamentals
C# 2012 Fundamentals
C# 2010 Fundamentals
iOS® 6 App Development Fundamentals
JavaScript Fundamentals
Visual Basic® Fundamentals

To receive updates on Deitel publications, Resource Centers, training courses, partner offers and more, please join the Deitel communities on

- Facebook®—facebook.com/DeitelFan
- Twitter®—@deitel
- Google+™—google.com/+DeitelFan
- YouTube™—youtube.com/DeitelTV
- LinkedIn®—linkedin.com/company/deitel-&-associates

and 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 *Dive-Into® Series* on-site seminars offered by Deitel & Associates, Inc. worldwide, write to us at deitel@deitel.com or visit:

www.deitel.com/training/

For continuing updates on Pearson/Deitel publications visit:

www.deitel.com
www.pearsonhighered.com/deitel/

Visit the Deitel Resource Centers that will help you master programming languages, software development, Android and iOS app development, and Internet- and web-related topics:

www.deitel.com/ResourceCenters.html

Java™ HOW TO PROGRAM

TENTH EDITION

EARLY OBJECTS

Paul Deitel
Deitel & Associates, Inc.

Harvey Deitel
Deitel & Associates, Inc.

DEITEL®

PEARSON

Boston Columbus Indianapolis New York San Francisco Upper Saddle River
Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montreal Toronto
Delhi Mexico City Sao Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo

Editorial Director, ECS: ***Marcia Horton***
Executive Editor: ***Tracy Johnson (Dunkelberger)***
Director of Marketing: ***Christy Lesko***
Marketing Manager: ***Yez Alayan***
Marketing Assistant: ***Jon Bryant***
Director of Program Management: ***Erin Gregg***
Program Management—Team Lead: ***Scott Disanno***
Program Manager: ***Carole Snyder***
Project Management-Team Lead: ***Laura Burgess***
Project Manager: ***Robert Engelhardt***
Procurement Specialist: ***Linda Sager***
Cover Design: ***Paul Deitel***, ***Harvey Deitel***, ***Abbey Deitel***, ***Barbara Deitel***, ***Laura Gardner***
Permissions Supervisor: ***Michael Joyce***
Permissions Administrator: ***Jenell Forschler***
Director, Image Asset Services: ***Annie Atherton***
Manager, Visual Research: ***Karen Sanatar***
Cover Art: © ***Nikrub/Shutterstock***
Media Project Manager: ***Renata Butera***

Credits and acknowledgments borrowed from other sources and reproduced, with permission, in this textbook appear on page vi.

Library of Congress Cataloging-in-Publication Data
On file

10 9 8 7 6 5 4 3 2 1
ISBN-10: 0-13-380780-0
ISBN-13: 978-0-13-380780-6

PEARSON

To Brian Goetz,
Oracle's Java Language Architect and
Specification Lead for Java SE 8's Project Lambda:

Your mentorship helped us make a better book.
Thank you for insisting that we get it right.

Paul and Harvey Deitel

Trademarks

DEITEL, the double-thumbs-up bug and DIVE INTO are registered trademarks of Deitel and Associates, Inc.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Microsoft and/or its respective suppliers make no representations about the suitability of the information contained in the documents and related graphics published as part of the services for any purpose. All such documents and related graphics are provided "as is" without warranty of any kind. Microsoft and/or its respective suppliers hereby disclaim all warranties and conditions with regard to this information, including all warranties and conditions of merchantability, whether express, implied or statutory, fitness for a particular purpose, title and non-infringement. In no event shall Microsoft and/or its respective suppliers be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of information available from the services.

The documents and related graphics contained herein could include technical inaccuracies or typographical errors. Changes are periodically added to the information herein. Microsoft and/or its respective suppliers may make improvements and/or changes in the product(s) and/or the program(s) described herein at any time. Partial screen shots may be viewed in full within the software version specified.

Microsoft® and Windows® are registered trademarks of the Microsoft Corporation in the U.S.A. and other countries. Screen shots and icons reprinted with permission from the Microsoft Corporation. This book is not sponsored or endorsed by or affiliated with the Microsoft Corporation.UNIX is a registered trademark of The Open Group.

Apache is a trademark of The Apache Software Foundation.

CSS and XML are registered trademarks of the World Wide Web Consortium.

Firefox is a registered trademark of the Mozilla Foundation.

Google is a trademark of Google, Inc.

Mac and OS X are trademarks of Apple Inc., registered in the U.S. and other countries.

Linux is a registered trademark of Linus Torvalds. All trademarks are property of their respective owners.

Throughout this book, trademarks are used. Rather than put a trademark symbol in every occurrence of a trademarked name, we state that we are using the names in an editorial fashion only and to the benefit of the trademark owner, with no intention of infringement of the trademark.

Contents

Chapters 26–34 and Appendices F–N are PDF documents posted online at the book's Companion Website (located at `www.pearsonhighered.com/deitel/`). See the inside front cover for information on accessing the Companion Website.

Online Chapters and Appendices

Chapters 26–34 and Appendices F–N are PDF documents posted online at the book's Companion Website (located at `www.pearsonhighered.com/deitel/`). See the inside front cover for information on accessing the Companion Website.

This page intentionally left blank

Foreword

I've been enamored with Java even prior to its 1.0 release in 1995, and have subsequently been a Java developer, author, speaker, teacher and Oracle Java Technology Ambassador. In this journey, it has been my privilege to call Paul Deitel a colleague, and to often leverage and recommend his *Java How To Program* book. In its many editions, this book has proven to be a great text for college and professional courses that I and others have developed to teach the Java programming language.

One of the qualities that makes this book a great resource is its thorough and insightful coverage of Java concepts, including those introduced recently in Java SE 8. Another useful quality is its treatment of concepts and practices essential to effective software development.

As a long-time fan of this book, I'd like to point out some of the features of this tenth edition about which I'm most excited:

- An ambitious new chapter on Java lambda expressions and streams. This chapter starts out with a primer on functional programming, introducing Java lambda expressions and how to use streams to perform functional programming tasks on collections.
- Although concurrency has been addressed since the first edition of the book, it is increasingly important because of multi-core architectures. There are timing examples—using the new Date/Time API classes introduced in Java SE 8—in the concurrency chapter that show the performance improvements with multi-core over single-core.
- JavaFX is Java's GUI/graphics/multimedia technology moving forward, so it is nice to see a three-chapter treatment of JavaFX in the Deitel live-code pedagogic style. One of these chapters is in the printed book and the other two are online.

Please join me in congratulating Paul and Harvey Deitel on their latest edition of a wonderful resource for computer science students and software developers alike!

James L. Weaver
Java Technology Ambassador
Oracle Corporation

This page intentionally left blank

Preface

"The chief merit of language is clearness…"
—Galen

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/deitelfan`)
- 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[1]

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 JavaSE 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).

Pre-Java-SE-8 topics	Corresponding Java SE 8 discussions and examples
Chapter 7, Arrays and ArrayLists	Sections 17.3–17.4 introduce basic lambda and streams capabilities that process one-dimensional arrays.
Chapter 10, Object-Oriented Programming: Polymorphism and Interfaces	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.
Chapters 12 and 22, GUI Components: Part 1 and 2, respectively	Section 17.9 shows how to use a lambda to implement a Swing event-listener functional interface.
Chapter 14, Strings, Characters and Regular Expressions	Section 17.5 shows how to use lambdas and streams to process collections of `String` objects.
Chapter 15, Files, Streams and Object Serialization	Section 17.7 shows how to use lambdas and streams to process lines of text from a file.
Chapter 23, Concurrency	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.
Chapter 25, JavaFX GUI: Part 1	Section 25.5.5 shows how to use a lambda to implement a JavaFX event-listener functional interface.

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

- *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.

```
http://bit.ly/CERTOracleSecureJava
```

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

- ***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-

nally, 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.

- ***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.

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.*

Software Engineering Observation

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

Look-and-Feel Observation

The Look-and-Feel Observations *highlight graphical-user-interface conventions. These observations help you design attractive, user-friendly graphical user interfaces that conform to industry norms.*

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/replocator/`.

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

(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

```
http://www.informit.com/store/sales.aspx
```

This page intentionally left blank

Before You Begin

This section contains information you should review before using this book. Any updates to the information presented here will be posted at:

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

In addition, we provide Dive-Into® videos (which will be available in time for Fall 2014 classes) that demonstrate the instructions in this Before You Begin section.

Font and Naming Conventions

We use fonts to distinguish between on-screen components (such as menu names and menu items) and Java code or commands. Our convention is to emphasize on-screen components in a sans-serif bold **Helvetica** font (for example, **File** menu) and to emphasize Java code and commands in a sans-serif `Lucida` font (for example, `System.out.println()`).

Software Used in the Book

All the software you'll need for this book is available free for download from the web. With the exception of the examples that are specific to Java SE 8, all of the examples were tested with the Java SE 7 and Java SE 8 Java Standard Edition Development Kits (JDKs).

Java Standard Edition Development Kit 7 (JDK 7)

JDK 7 for Windows, OS X and Linux platforms is available from:

```
http://www.oracle.com/technetwork/java/javase/downloads/index.html
```

Java Standard Edition Development Kit (JDK) 8

At the time of this publication, the near-final version of JDK 8 for Windows, OS X and Linux platforms was available from:

```
https://jdk8.java.net/download.html
```

Once JDK 8 is released as final, it will be available from:

```
http://www.oracle.com/technetwork/java/javase/downloads/index.html
```

JDK Installation Instructions

After downloading the JDK installer, be sure to carefully follow the JDK installation instructions for your platform at:

```
http://docs.oracle.com/javase/7/docs/webnotes/install/index.html
```

Though these instructions are for JDK 7, they also apply to JDK 8—you'll need to update the JDK version number in any version-specific instructions.

Setting the PATH Environment Variable

The PATH environment variable on your computer designates which directories the computer searches when looking for applications, such as the applications that enable you to compile and run your Java applications (called javac and java, respectively). *Carefully follow the installation instructions for Java on your platform to ensure that you set the PATH environment variable correctly.* The steps for setting environment variables differ by operating system and sometimes by operating system version (e.g., Windows 7 vs. Windows 8). Instructions for various platforms are listed at:

```
http://www.java.com/en/download/help/path.xml
```

If you do not set the PATH variable correctly on Windows and some Linux installations, when you use the JDK's tools, you'll receive a message like:

```
'java' is not recognized as an internal or external command,
operable program or batch file.
```

In this case, go back to the installation instructions for setting the PATH and recheck your steps. If you've downloaded a newer version of the JDK, you may need to change the name of the JDK's installation directory in the PATH variable.

JDK Installation Directory and the `bin` *Subdirectory*

The JDK's installation directory varies by platform. The directories listed below are for Oracle's JDK 7 update 51:

- 32-bit JDK on Windows:
  `C:\Program Files (x86)\Java\jdk1.7.0_51`
- 64-bit JDK on Windows:
  `C:\Program Files\Java\jdk1.7.0_51`
- Mac OS X:
  `/Library/Java/JavaVirtualMachines/jdk1.7.0_51.jdk/Contents/Home`
- Ubuntu Linux:
  `/usr/lib/jvm/java-7-oracle`

Depending on your platform, the JDK installation folder's name might differ if you're using a different update of JDK 7 or using JDK 8. For Linux, the install location depends on the installer you use and possibly the version of Linux that you use. We used Ubuntu Linux. The PATH environment variable must point to the JDK installation directory's **`bin`** subdirectory.

When setting the PATH, be sure to use the proper JDK-installation-directory name for the specific version of the JDK you installed—as newer JDK releases become available, the JDK-installation-directory name changes to include an *update version number*. For example, at the time of this writing, the most recent JDK 7 release was update 51. For this version, the JDK-installation-directory name ends with "_51".

Setting the CLASSPATH Environment Variable

If you attempt to run a Java program and receive a message like

```
Exception in thread "main" java.lang.NoClassDefFoundError: YourClass
```

then your system has a CLASSPATH environment variable that must be modified. To fix the preceding error, follow the steps in setting the PATH environment variable, to locate the CLASSPATH variable, then edit the variable's value to include the local directory—typically represented as a dot (.). On Windows add

```
.;
```

at the beginning of the CLASSPATH's value (with no spaces before or after these characters). On other platforms, replace the semicolon with the appropriate path separator characters—typically a colon (:).

Setting the JAVA_HOME Environment Variable

The Java DB database software that you'll use in Chapter 24 and several online chapters requires you to set the JAVA_HOME environment variable to your JDK's installation directory. The same steps you used to set the PATH may also be used to set other environment variables, such as JAVA_HOME.

Java Integrated Development Environments (IDEs)

There are many Java integrated development environments that you can use for Java programming. For this reason, we used only the JDK command-line tools for most of the book's examples. We provide Dive-Into® videos (which will be available in time for Fall 2014 classes) that show how to download, install and use three popular IDEs—NetBeans, Eclipse and IntelliJ IDEA. We use NetBeans in Chapter 25 and several of the book's online chapters.

NetBeans Downloads

You can download the JDK/NetBeans bundle from:

```
http://www.oracle.com/technetwork/java/javase/downloads/index.html
```

The NetBeans version that's bundled with the JDK is for Java SE development. The online JavaServer Faces (JSF) chapters and web services chapter use the Java Enterprise Edition (Java EE) version of NetBeans, which you can download from:

```
https://netbeans.org/downloads/
```

This version supports both Java SE and Java EE development.

Eclipse Downloads

You can download the Eclipse IDE from:

```
https://www.eclipse.org/downloads/
```

For Java SE development choose the Eclipse IDE for Java Developers. For Java Enterprise Edition (Java EE) development (such as JSF and web services), choose the Eclipse IDE for Java EE Developers—this version supports both Java SE and Java EE development.

IntelliJ IDEA Community Edition Downloads

You can download the free IntelliJ IDEA Community Edition from:

```
http://www.jetbrains.com/idea/download/index.html
```

The free version supports only Java SE development.

Obtaining the Code Examples

The examples for *Java How to Program, 10/e* are available for download at

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

under the heading **Download Code Examples and Other Premium Content.** The examples are also available from

```
http://www.pearsonhighered.com/deitel
```

When you download the ZIP archive file, write down the location where you choose to save it on your computer.

Extract the contents of `examples.zip` using a ZIP extraction tool such as 7-Zip (`www.7-zip.org`), WinZip (`www.winzip.com`) or the built-in capabilities of your operating system. Instructions throughout the book assume that the examples are located at:

- `C:\examples` on Windows
- your user account home folder's `examples` subfolder on Linux
- your `Documents` folders `examples` subfolder on Mac OS X

Java's Nimbus Look-and-Feel

Java comes bundled with a cross-platform look-and-feel known as Nimbus. For programs with Swing graphical user interfaces (e.g., Chapters 12 and 22), we configured our test computers to use Nimbus as the default look-and-feel.

To set Nimbus as the default for all Java applications, you must create a text file named `swing.properties` in the `lib` folder of both your JDK installation folder and your JRE installation folder. Place the following line of code in the file:

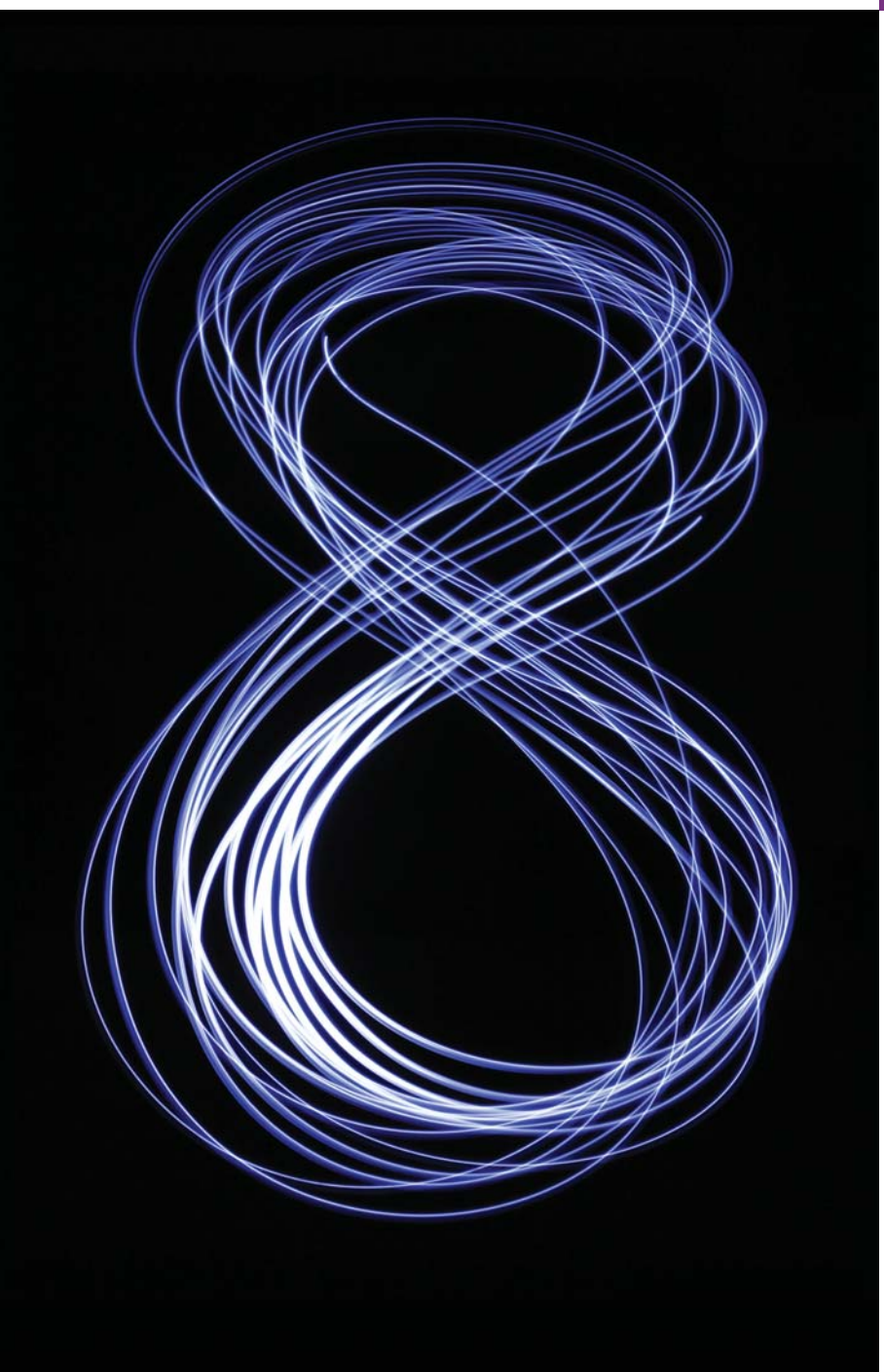
```
swing.defaultlaf=com.sun.java.swing.plaf.nimbus.NimbusLookAndFeel
```

For more information on locating these folders visit `http://docs.oracle.com/javase/7/docs/webnotes/install/index.html`. [*Note:* In addition to the standalone JRE, there's a JRE nested in your JDK's installation folder. If you're using an IDE that depends on the JDK (e.g., NetBeans), you may also need to place the `swing.properties` file in the nested `jre` folder's `lib` folder.]

You're now ready to begin your Java studies with *Java How to Program, 10/e*. We hope you enjoy the book!

Introduction to Computers, the Internet and Java

1



Man is still the most extraordinary computer of all.
—John F. Kennedy

Good design is good business.
—Thomas J. Watson, Founder of IBM

Objectives

In this chapter you'll:

- Learn about exciting recent developments in the computer field.
- Learn computer hardware, software and networking basics.
- Understand the data hierarchy.
- Understand the different types of programming languages.
- Understand the importance of Java and other leading programming languages.
- Understand object-oriented programming basics.
- Learn the importance of the Internet and the web.
- Learn a typical Java program-development environment.
- Test-drive a Java application.
- Learn some key recent software technologies.
- See how to keep up-to-date with information technologies.

Outline

1.1 Introduction

Welcome to Java—one of the world's most widely used computer programming languages. You're already familiar with the powerful tasks computers perform. Using this textbook, you'll write instructions commanding computers to perform those tasks. **Software** (i.e., the instructions you write) controls **hardware** (i.e., computers).

You'll learn *object-oriented programming*—today's key programming methodology. You'll create and work with many *software objects*.

For many organizations, the preferred language for meeting their enterprise programming needs is Java. Java is also widely used for implementing Internet-based applications and software for devices that communicate over a network.

Forrester Research predicts more than two billion PCs will be in use by 2015.[1] According to Oracle, 97% of enterprise desktops, 89% of PC desktops, three billion devices (Fig. 1.1) and 100% of all Blu-ray Disc™ players run Java, and there are over 9 million Java developers.[2]

According to a study by Gartner, mobile devices will continue to outpace PCs as users' primary computing devices; an estimated 1.96 billion smartphones and 388 million tablets will be shipped in 2015—8.7 times the number of PCs.[3] By 2018, the mobile applications

1. http://www.worldometers.info/computers.
2. http://www.oracle.com/technetwork/articles/java/javaone12review-1863742.html.
3. http://www.gartner.com/newsroom/id/2645115.

Devices		
Airplane systems	ATMs	Automobile infotainment systems
Blu-ray Disc™ players	Cable boxes	Copiers
Credit cards	CT scanners	Desktop computers
e-Readers	Game consoles	GPS navigation systems
Home appliances	Home security systems	Light switches
Lottery terminals	Medical devices	Mobile phones
MRIs	Parking payment stations	Printers
Transportation passes	Robots	Routers
Smart cards	Smart meters	Smartpens
Smartphones	Tablets	Televisions
TV set-top boxes	Thermostats	Vehicle diagnostic systems

Fig. 1.1 | Some devices that use Java.

(apps) market is expected to reach $92 billion.[4] This is creating significant career opportunities for people who program mobile applications, many of which are programmed in Java (see Section 1.6.3).

Java Standard Edition

Java has evolved so rapidly that this tenth edition of *Java How to Program*—based on **Java Standard Edition 7 (Java SE 7)** and **Java Standard Edition 8 (Java SE 8)**—was published just 17 years after the first edition. Java Standard Edition contains the capabilities needed to develop desktop and server applications. The book can be used with *either* Java SE 7 *or* Java SE 8 (released just after this book was published). All of the Java SE 8 features are discussed in modular, easy-to-include-or-omit sections throughout the book.

Prior to Java SE 8, Java supported three programming paradigms—*procedural programming*, *object-oriented programming* and *generic programming*. Java SE 8 adds *functional programming*. In Chapter 17, we'll show how to use functional programming to write programs faster, more concisely, with fewer bugs and that are easier to *parallelize* (i.e., perform multiple calculations simultaneously) to take advantage of today's multi-core hardware architectures to enhance application performance.

Java Enterprise Edition

Java is used in such a broad spectrum of applications that it has two other editions. The **Java Enterprise Edition (Java EE)** is geared toward developing large-scale, distributed networking applications and web-based applications. In the past, most computer applications ran on "standalone" computers (computers that were not networked together). Today's applications can be written with the aim of communicating among the world's computers via the Internet and the web. Later in this book we discuss how to build such web-based applications with Java.

4. `https://www.abiresearch.com/press/tablets-will-generate-35-of-this-years-25-billion-`.

Java Micro Edition
The **Java Micro Edition (Java ME)**—a subset of Java SE—is geared toward developing applications for resource-constrained embedded devices, such as smartwatches, MP3 players, television set-top boxes, smart meters (for monitoring electric energy usage) and more.

1.2 Hardware and Software

Computers can perform calculations and make logical decisions phenomenally faster than human beings can. Many of today's personal computers can perform billions of calculations in one second—more than a human can perform in a lifetime. *Supercomputers* are already performing *thousands of trillions (quadrillions)* of instructions per second! China's National University of Defense Technology's Tianhe-2 supercomputer can perform over 33 quadrillion calculations per second (33.86 *petaflops*)![5] To put that in perspective, *the Tianhe-2 supercomputer can perform in one second about 3 million calculations for every person on the planet!* And—these supercomputing "upper limits" are growing quickly.

Computers process data under the control of sequences of instructions called **computer programs**. These software programs guide the computer through ordered actions specified by people called computer **programmers**. In this book, you'll learn a key programming methodology that's enhancing programmer productivity, thereby reducing software development costs—*object-oriented programming*.

A computer consists of various devices referred to as hardware (e.g., the keyboard, screen, mouse, hard disks, memory, DVD drives and processing units). Computing costs are *dropping dramatically*, owing to rapid developments in hardware and software technologies. Computers that might have filled large rooms and cost millions of dollars decades ago are now inscribed on silicon chips smaller than a fingernail, costing perhaps a few dollars each. Ironically, silicon is one of the most abundant materials on Earth—it's an ingredient in common sand. Silicon-chip technology has made computing so economical that computers have become a commodity.

1.2.1 Moore's Law

Every year, you probably expect to pay at least a little more for most products and services. The opposite has been the case in the computer and communications fields, especially with regard to the hardware supporting these technologies. For many decades, hardware costs have fallen rapidly.

Every year or two, the capacities of computers have approximately *doubled* inexpensively. This remarkable trend often is called **Moore's Law**, named for the person who identified it in the 1960s, Gordon Moore, co-founder of Intel—the leading manufacturer of the processors in today's computers and embedded systems. Moore's Law and related observations apply especially to the amount of memory that computers have for programs, the amount of secondary storage (such as disk storage) they have to hold programs and data over longer periods of time, and their processor speeds—the speeds at which they *execute* their programs (i.e., do their work).

5. `http://www.top500.org/`.

Similar growth has occurred in the communications field—costs have plummeted as enormous demand for communications *bandwidth* (i.e., information-carrying capacity) has attracted intense competition. We know of no other fields in which technology improves so quickly and costs fall so rapidly. Such phenomenal improvement is truly fostering the *Information Revolution*.

1.2.2 Computer Organization

Regardless of differences in *physical* appearance, computers can be envisioned as divided into various **logical units** or sections (Fig. 1.2).

Logical unit	Description
Input unit	This "receiving" section obtains information (data and computer programs) from **input devices** and places it at the disposal of the other units for processing. Most user input is entered into computers through keyboards, touch screens and mouse devices. Other forms of input include receiving voice commands, scanning images and barcodes, reading from secondary storage devices (such as hard drives, DVD drives, Blu-ray Disc™ drives and USB flash drives—also called "thumb drives" or "memory sticks"), receiving video from a webcam and having your computer receive information from the Internet (such as when you stream videos from YouTube® or download e-books from Amazon). Newer forms of input include position data from a GPS device, and motion and orientation information from an *accelerometer* (a device that responds to up/down, left/right and forward/backward acceleration) in a smartphone or game controller (such as Microsoft® Kinect® and Xbox®, Wii™ Remote and Sony® PlayStation® Move).
Output unit	This "shipping" section takes information the computer has processed and places it on various **output devices** to make it available for use outside the computer. Most information that's output from computers today is displayed on screens (including touch screens), printed on paper ("going green" discourages this), played as audio or video on PCs and media players (such as Apple's iPods) and giant screens in sports stadiums, transmitted over the Internet or used to control other devices, such as robots and "intelligent" appliances. Information is also commonly output to secondary storage devices, such as hard drives, DVD drives and USB flash drives. A popular recent form of output is smartphone vibration.
Memory unit	This rapid-access, relatively low-capacity "warehouse" section retains information that has been entered through the input unit, making it immediately available for processing when needed. The memory unit also retains processed information until it can be placed on output devices by the output unit. Information in the memory unit is *volatile*—it's typically lost when the computer's power is turned off. The memory unit is often called either **memory**, **primary memory** or **RAM** (Random Access Memory). Main memories on desktop and notebook computers contain as much as 128 GB of RAM. GB stands for gigabytes; a gigabyte is approximately one billion bytes. A **byte** is eight bits. A bit is either a 0 or a 1.

Fig. 1.2 | Logical units of a computer. (Part 1 of 2.)