
Preface

Live in fragments no longer. Only connect.
Edward Morgan Forster

Welcome to *Java How to Program, Fourth Edition* and the exciting world of programming with the *Java™ 2 Platform, Standard Edition*. This book is by an old guy and a young guy. The old guy (HMD; Massachusetts Institute of Technology 1967) has been programming and/or teaching programming for 40 years. The young guy (PJD; MIT 1991) has been programming and/or teaching programming for 22 years, and is both a Sun Certified Java Programmer and a Sun Certified Java Developer. The old guy programs and teaches from experience; the young guy does so from an inexhaustible reserve of energy. The old guy wants clarity; the young guy wants performance. The old guy seeks elegance and beauty; the young guy wants results. We got together to produce a book we hope you will find informative, challenging and entertaining.

In November 1995, we attended an Internet/World Wide Web conference in Boston to hear about Java. A Sun Microsystems representative spoke on Java in a packed convention ballroom. During that presentation, we saw the future of programming unfold. The first edition of *Java How to Program* was born at that moment and was published as the world's first Java computer science textbook.

The world of Java is evolving so rapidly that *Java How to Program: Fourth Edition* is being published less than five years after the first edition. This creates tremendous challenges and opportunities for us as authors, for our publisher—Prentice Hall, for instructors, for students and for professional people.

Before Java appeared, we were convinced that C++ would replace C as the dominant application development language and systems programming language for the next decade. However, the combination of the World Wide Web and Java now increases the prominence of the Internet in information systems strategic planning and implementation. Organizations want to integrate the Internet “seamlessly” into their information systems. Java is more appropriate than C++ for this purpose.

New Features in *Java How to Program: Fourth Edition*

This edition contains many new features and enhancements including:

- **Full-Color Presentation.** The book is now in full color. In the book's earlier two-color editions, the programs were displayed in black and the screen captures appeared in the second color. Full color enables readers to see sample outputs as they would appear on a color monitor. Also, we now syntax color all the Java code, as many of today's Java development environments do. Our syntax-coloring conventions are as follows:

```
comments appear in green
keywords appear in dark blue
constants and literal values appear in light blue
class, method and variable names appear in black
```

- **"Code Washing."** This is our own term for the process we used to convert all the programs in the book to a more open layout with enhanced commenting. We have grouped program code into small, well-documented pieces. This greatly improves code readability—an especially important goal for us given that this new edition contains more than 25,000 lines of code.
- **Tune-Up.** We performed a substantial tune-up of the book's contents based on our own notes from extensive teaching in our professional Java seminars. In addition, a distinguished team of reviewers read the third edition book and provided us with their comments and criticisms. There are literally thousands of fine-tuning improvements over the third edition.
- **Thinking About Objects.** This optional 180-page case study introduces *object-oriented design (OOD)* with the *Unified Modeling Language (the UML)*. Many chapters in this edition end with a "Thinking About Objects" section in which we present a carefully paced introduction to object orientation. Our goal in these sections is to help you develop an object-oriented way of thinking to be able to design and implement more substantial systems. These sections also introduce you to the Unified Modeling Language (UML). The UML is a graphical language that allows people who build systems (e.g., software architects, systems engineers and programmers) to represent their object-oriented designs using a common notation. The "Thinking About Objects" section in Chapter 1 introduces basic concepts and terminology. Chapters 2–13, 15 and 22 (22 is on the CD) and Appendices G, H and I (also on the CD) include optional "Thinking About Objects" sections that present a substantial object-oriented elevator case study that applies the techniques of *object-oriented design (OOD)*. Appendices G, H and I fully implement the case study design in Java code. This case study will help prepare you for the kinds of substantial projects you are likely to encounter in industry. If you are a student and your instructor does not plan to include this case study in your course, you may want to read the case study on your own. We believe it will be well worth your effort to walk through this large and challenging project. The material presented in the case-study sections reinforces the material covered in the corresponding chapters. You will experience a solid introduction to object-oriented design with the UML. Also, you will sharpen your code-reading skills by touring

a carefully written and well-documented 3,465-line Java program that completely solves the problem presented in the case study.

- **Discovering Design Patterns.** These optional sections introduce popular object-oriented design patterns in use today. Most of the examples provided in this book contain fewer than 150 lines of code. Such small examples normally do not require an extensive design process. However, some programs, such as our optional elevator-simulation case study, are more complex—they can require thousands of lines of code. Larger systems, such as automated teller machines or air-traffic control systems, could contain millions, or even hundreds of millions, of lines of code. Effective design is crucial to the proper construction of such complex systems. Over the past decade, the software engineering industry has made significant progress in the field of *design patterns*—proven architectures for constructing flexible and maintainable object-oriented software.¹ Using design patterns can substantially reduce the complexity of the design process. We present several design patterns in Java, but these design patterns can be implemented in any object-oriented language, such as C++, C# or Visual Basic. We describe several design patterns used by Sun Microsystems in the Java API. We use design patterns in many programs in this book, which we will identify in our “Discovering Design Patterns” sections. These programs provide examples of using design patterns to construct reliable, robust object-oriented software.
- **Chapter 22 (on the CD), Java Media Framework (JMF) and JavaSound.** This chapter introduces to Java’s audio and video capabilities, enhancing our Chapter 18 multimedia coverage. With the Java Media Framework, a Java program can play audio and video media, and *capture* audio and video media from devices such as microphones and video cameras. The JMF enables Java developers to create *streaming media* applications, in which a Java program sends live or recorded audio or video feeds across the Internet to other computers, then applications on those other computers play the media as it arrives over the network. The JavaSound APIs enable programs to manipulate MIDI (Musical Instrument Digital Interface) sounds and captured media (i.e., media from a device such as a microphone). The chapter concludes with a substantial MIDI-processing application that enables users to record MIDI files or select MIDI files to play. Users can create their own MIDI music by interacting with the application’s simulated synthesizer keyboard. The application can synchronize playing the notes in a MIDI file with pressing the keys on the simulated synthesizer keyboard—similar to a player piano. [Note: Chapters 18 and 22 both provide substantial sets of exercises. Each chapter also has a special section containing additional interesting and challenging multimedia projects. These are intended only as suggestions for major projects. Solutions are not provided for these additional exercises in either the *Instructor’s Manual* or the *Java 2 Multimedia Cyber Classroom*.]
- **Enhanced TCP/IP-Based Networking.** We include a new capstone example in Chapter 17 that introduces *multicasting* for sending information to groups of network clients. This Deitel Messenger case study emulates many of today’s popular

1. Gamma, Erich, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns; Elements of Reusable Object-Oriented Software*. (Massachusetts: Addison-Wesley, 1995).

instant-messaging applications that enable computer users to communicate with friends, relatives and co-workers over the Internet. This 1130-line, multithreaded, client/server Java program uses most of the techniques presented to this point in the book.

- **Appendix J (on the CD), Career Opportunities.** This detailed appendix introduces career services on the Internet. We explore online career services from the employer and employee's perspective. We suggest sites on which you can submit applications, search for jobs and review applicants (if you are interested in hiring someone). We also review services that build recruiting pages directly into e-businesses. One of our reviewers told us that he had just gone through a job search largely using the Internet and this chapter would have really expanded his search dramatically.
- **Appendix K (on the CD), Unicode.** This appendix overviews the *Unicode Standard*. As computer systems evolved worldwide, computer vendors developed numeric representations of character sets and special symbols for the local languages spoken in different countries. In some cases, different representations were developed for the same languages. Such disparate character sets made communication between computer systems difficult. Java supports the Unicode Standard (maintained by a non-profit organization called the *Unicode Consortium*), which defines a single character set with unique numeric values for characters and special symbols in most spoken languages. This appendix discusses the Unicode Standard, overviews the Unicode Consortium Web site (unicode.org) and shows a Java example that displays "Welcome" in eight different languages!
- **Java 2 Plug-In Moved to Chapter 3, Introduction to Applets.** Students enjoy seeing immediate results as they execute their Java programs. This is difficult if those programs are Java applets that execute in Web browsers. Most of today's Web browsers (with the exception of Netscape Navigator 6) do not support Java 2 applets directly, so students must test their applet programs with the `appletviewer` utility. Sun Microsystems provides the Java 2 Plug-in to enable Java 2 applets to execute in a Web browser that does not support Java 2. The discussion of the Java Plug-in walks the student through the steps necessary to execute an applet in today's Web browsers.
- **Chapter 22 and Appendices E-K on the CD.** There are so many topics covered in this new edition that we could not fit them all in the book! On the CD that accompanies this book, you will find the following chapter and appendices: Chapter 22, Java Media Framework (JMF) and Java Sound; Appendix E, Number Systems; Appendix F, Creating HTML Documentation with `javadoc`; Appendix G, Elevator Events and Listener Interfaces; Appendix H, Elevator Model; Appendix I, Elevator View; Appendix J, Career Opportunities; and Appendix K, Unicode.
- **Chapters Moved to Advanced Java™ 2 Platform How to Program.** Four chapters from *Java How to Program, Third Edition* have been moved to our new book *Advanced Java 2 Platform How to Program* and greatly enhanced. These chapters are: Java Database Connectivity (JDBC), Servlets, Remote Method Invocation and JavaBeans. *Advanced Java 2 Platform How to Program* covers each of these topics in more depth. We present the Table of Contents of *Advanced Java 2 Platform How to Program* shortly.

Some Notes to Instructors

A World of Object Orientation

When we wrote the first edition of *Java How to Program*, universities were still emphasizing procedural programming in languages like Pascal and C. The leading-edge courses were using object-oriented C++, but these courses were generally mixing a substantial amount of procedural programming with object-oriented programming—something that C++ lets you do, but Java does not. By the third edition of *Java How to Program*, many universities were switching from C++ to Java in their introductory curricula, and instructors were emphasizing a pure object-oriented programming approach. In parallel with this activity, the software engineering community was standardizing its approach to modeling object-oriented systems with the UML, and the design-patterns movement was taking shape. *Java How to Program* has many audiences, so we designed the book to be customizable. In particular, we included more than 200 pages of optional material that introduces object-oriented design, the UML and design patterns, and presents a substantial case study in object-oriented design and programming. This material is carefully distributed throughout the book to enable instructors to emphasize “industrial-strength” object-oriented design in their courses.

Students Like Java

Students are highly motivated by the fact that they are learning a leading-edge language (Java) and a leading-edge programming paradigm (object-oriented programming) that will be immediately useful to them while in the university environment and when they head into a world in which the Internet and the World Wide Web have a massive prominence. Students quickly discover that they can do great things with Java, so they are willing to put in the extra effort. Java helps programmers unleash their creativity. We see this in the Java courses Deitel & Associates, Inc. teaches. Once our students enter lab, we can’t hold them back. They eagerly experiment and explore portions of the Java class libraries that we haven’t as yet covered in class. They produce applications that go well beyond anything we’ve ever tried in our introductory C and C++ courses. And they tell us about projects they “can’t wait” to try after the course.

Focus of the Book

Our goal was clear—produce a Java textbook for introductory university-level courses in computer programming for students with little or no programming experience, yet offer the depth and the rigorous treatment of theory and practice demanded by traditional, upper-level courses and that satisfies professionals’ needs. To meet these goals, we produced a comprehensive book, because our text patiently teaches the basics of computer programming and of the Java language (i.e., data types, control structures, methods, arrays, recursion and other “traditional” programming topics); presents key programming paradigms, including object-based programming, object-oriented programming, event-driven programming and concurrent programming; and provides an extensive treatment of the Java class libraries.

Evolution of Java How to Program

Java How to Program (first edition) was the world’s first university computer science textbook on Java. We wrote it fresh on the heels of *C How to Program, Second Edition* and *C++ How to Program*. Hundreds of thousands of university students and professional peo-

ple worldwide have learned C, C++ and Java from these texts. Upon publication in August, 2001 *Java How to Program, Fourth Edition* will be used in hundreds of universities and thousands of corporations and government organizations worldwide. Deitel & Associates, Inc. taught Java courses internationally to thousands of students as we were writing the various editions of *Java How to Program*. We carefully monitored the effectiveness of these courses and tuned the material accordingly.

Conceptualization of Java

We believe in Java. Its conceptualization (and public release in 1995) by Sun Microsystems, the creators of Java, was brilliant. Sun based the new language on two of the world's most widely used implementation languages, C and C++. This immediately gave Java a huge pool of highly skilled programmers who were implementing most of the world's new operating systems, communications systems, database systems, personal computer applications and systems software. Sun removed the messier, more complex and error-prone C/C++ features (such as pointers, operator overloading and multiple inheritance, among others). They kept the language concise by removing special-purpose features that were used by only small segments of the programming community. They made the language truly portable to be appropriate for implementing Internet-based and World-Wide-Web-based applications, and they built in the features people really need such as strings, graphics, graphical user interface components, exception handling, multithreading, multimedia (audio, images, animation and video), file processing, database processing, Internet and World Wide Web-based client/server networking and distributed computing, and prepackaged data structures. Then they made the language available *at no charge* to millions of potential programmers worldwide.

2.5 Million Java Developers

Java was promoted in 1995 as a means of adding “dynamic content” to World-Wide-Web pages. Instead of Web pages with only text and static graphics, people's Web pages could now “come alive” with audios, videos, animations, interactivity—and soon, three-dimensional imaging. But we saw much more in Java than this. Java's features are precisely what businesses and organizations need to meet today's information-processing requirements. So we immediately viewed Java as having the potential to become one of the world's key general-purpose programming languages. In fact, Java has revolutionized software development with multimedia-intensive, platform-independent, object-oriented code for conventional, Internet-, Intranet- and Extranet-based applications and applets. Java now has 2.5 million developers worldwide—a stunning accomplishment given that it has only been available publicly for six years. No other programming language has ever acquired such a large developer base so quickly.

Enabling Multimedia-Based Applications and Communications

The computer field has never seen anything like the Internet/World Wide Web/Java “explosion” occurring today. People want to communicate. People need to communicate. Sure they have been doing that since the dawn of civilization, but computer communications have been mostly limited to digits, alphabetic characters and special characters. Today, we are in the midst of a multimedia revolution. People want to transmit pictures and they want those pictures to be in color. They want to transmit voices, sounds, audio clips and full-motion color video (and they want nothing less than DVD quality). Eventually, people will in-

sist on three-dimensional, moving-image transmission. Our current flat, two-dimensional televisions will eventually be replaced with three-dimensional versions that turn our living rooms into “theaters-in-the-round.” Actors will perform their roles as if we were watching live theater. Our living rooms will be turned into miniature sports stadiums. Our business offices will enable video conferencing among colleagues half a world apart as if they were sitting around one conference table. The possibilities are intriguing and Java is playing a key role in turning many of them into reality.

Teaching Approach

Java How to Program, Fourth Edition contains a rich collection of examples, exercises, and projects drawn from many fields to provide the student with a chance to solve interesting real-world problems. The book concentrates on the principles of good software engineering and stresses program clarity. We avoid arcane terminology and syntax specifications in favor of teaching by example. Our code examples have been tested on popular Java platforms. We are educators who teach edge-of-the-practice topics in industry classrooms worldwide. The text emphasizes good pedagogy.

Learning Java via the Live-Code™ Approach

The book is loaded with live-code™ examples. This is the focus of the way we teach and write about programming, and the focus of each of our multimedia Cyber Classrooms and Web-based training courses as well. Each new concept is presented in the context of a complete, working Java program (application or applet) immediately followed by one or more screen captures showing the program’s output. We call this style of teaching and writing our ***live-code™ approach***. *We use the language to teach the language.* Reading these programs (25,000+ lines of code) is much like entering and running them on a computer.

Java and Swing from Chapter Two!

Java How to Program, Fourth Edition “jumps right in” with object-oriented programming, applications and the Swing-style GUI components from Chapter 2! People tell us this is a “gutsy” move, but Java students really want to “cut to the chase.” There is great stuff to be done in Java so let’s get right to it! Java is not trivial by any means, but it’s fun to program with and students can see immediate results. Students can get graphical, animated, multimedia-based, audio-intensive, multithreaded, database-intensive, network-based programs running quickly through Java’s extensive class libraries of “reusable components.” They can implement impressive projects. They are typically more creative and productive in a one- or two-semester course than in C and C++ introductory courses.

World Wide Web Access

All of the code for *Java How to Program* is on the CD that accompanies this book and is available on the Internet at the Deitel & Associates, Inc. Web site www.deitel.com. Please run each program as you read the text. Make changes to the code examples and see what happens. See how the Java compiler “complains” when you make various kinds of errors. Immediately see the effects of making changes to the code. It’s a great way to learn programming by doing programming. [This is copyrighted material. Feel free to use it as you study Java, but you may not republish any portion of it without explicit permission from the authors and Prentice Hall.]

Objectives

Each chapter begins with a statement of objectives. This tells the student what to expect and gives the student an opportunity, after reading the chapter, to determine if he or she has met these objectives. It is a confidence builder and a source of positive reinforcement.

Quotations

The learning objectives are followed by quotations. Some are humorous, some are philosophical, and some offer interesting insights. Our students enjoy relating the quotations to the chapter material. The quotations are worth a “second look” after you read each chapter.

Outline

The chapter Outline helps the student approach the material in top-down fashion. This, too, helps students anticipate what is to come and set a comfortable and effective learning pace.

25,576 Lines of Code in 197 Example Programs (with Program Outputs)

We present Java features in the context of complete, working Java programs. The programs range from just a few lines of code to substantial examples with several hundred lines of code (and 3,465 lines of code for the optional object-oriented elevator simulator example). Students should use the program code from the CD that accompanies the book or download the code from our Web site (www.deitel.com) and run each program while studying that program in the text.

545 Illustrations/Figures

An abundance of charts, line drawings and program outputs is included. The discussion of control structures, for example, features carefully drawn flowcharts. [Note: We do not teach flowcharting as a program development tool, but we do use a brief, flowchart-oriented presentation to specify the precise operation of each of Java’s control structures.]

605 Programming Tips

We have included programming tips to help students focus on important aspects of program development. We highlight hundreds of these tips in the form of *Good Programming Practices*, *Common Programming Errors*, *Testing and Debugging Tips*, *Performance Tips*, *Portability Tips*, *Software Engineering Observations* and *Look-and-Feel Observations*. These tips and practices represent the best we have gleaned from a combined six decades of programming and teaching experience. One of our students—a mathematics major—told us that she feels this approach is like the highlighting of axioms, theorems, and corollaries in mathematics books; it provides a basis on which to build good software.

**97 Good Programming Practices**

When we teach introductory courses, we state that the “buzzword” of each course is “clarity,” and we highlight as Good Programming Practices techniques for writing programs that are clearer, more understandable, more debuggable, and more maintainable.

**199 Common Programming Errors**

Students learning a language tend to make certain errors frequently. Focusing on these Common Programming Errors helps students avoid making the same errors and shortens lines outside instructors’ offices during office hours!



46 Testing and Debugging Tips

When we first designed this “tip type,” we thought we would use it strictly to tell people how to test and debug Java programs. In fact, many of the tips describe aspects of Java that reduce the likelihood of “bugs” and thus simplify the testing and debugging process.



67 Performance Tips

In our experience, teaching students to write clear and understandable programs is by far the most important goal for a first programming course. But students want to write the programs that run the fastest, use the least memory, require the smallest number of keystrokes, or dazzle in other nifty ways. Students really care about performance. They want to know what they can do to “turbo charge” their programs. So we have included 67 Performance Tips that highlight opportunities for improving program performance—making programs run faster or minimizing the amount of memory that they occupy.



24 Portability Tips

One of Java’s “claims to fame” is “universal” portability, so some programmers assume that if they implement an application in Java, the application will automatically be “perfectly” portable across all Java platforms. Unfortunately, this is not always the case. We include Portability Tips to help students write portable code and to provide insights on how Java achieves its high degree of portability. We had many more portability tips in our books, *C How to Program* and *C++ How to Program*. We needed fewer Portability Tips in *Java How to Program* because Java is designed to be portable top-to-bottom (for the most part)—much less effort is required on the Java programmer’s part to achieve portability than with C or C++.



134 Software Engineering Observations

The object-oriented programming paradigm requires a complete rethinking about the way we build software systems. Java is an effective language for performing good software engineering. The Software Engineering Observations highlight architectural and design issues that affect the construction of software systems, especially large-scale systems. Much of what the student learns here will be useful in upper-level courses and in industry as the student begins to work with large, complex real-world systems.



38 Look-and-Feel Observations

We provide Look-and-Feel Observations to highlight graphical user interface conventions. These observations help students design their own graphical user interfaces in conformance with industry norms.

Summary (983 Summary bullets)

Each chapter ends with additional pedagogical devices. We present a thorough, bullet-list-style summary of the chapter. On average, there are 42 summary bullets per chapter. This helps the students review and reinforce key concepts.

Terminology (2171 Terms)

We include in a *Terminology* section an alphabetized list of the important terms defined in the chapter—again, further reinforcement. On average, there are 95 terms per chapter.

397 Self-Review Exercises and Answers (Count Includes Separate Parts)

Extensive self-review exercises and answers are included for self-study. This gives the student a chance to build confidence with the material and prepare for the regular exercises. Students should be encouraged to do all the self-review exercises and check their answers.

779 Exercises (Count Includes Separate Parts)

Each chapter concludes with a set of exercises including simple recall of important terminology and concepts; writing individual Java statements; writing small portions of Java methods and classes; writing complete Java methods, classes, applications and applets; and writing major term projects. The large number of exercises across a wide variety of areas enables instructors to tailor their courses to the unique needs of their audiences and to vary course assignments each semester. Instructors can use these exercises to form homework assignments, short quizzes and major examinations. The solutions for most of the exercises are included on the *Instructor's Manual CD* that is *available only to instructors* through their Prentice-Hall representatives. **[NOTE: Please do not write to us requesting the instructor's manual. Distribution of this publication is strictly limited to college professors teaching from the book. Instructors may obtain the solutions manual only from their regular Prentice Hall representatives. We regret that we cannot provide the solutions to professionals.]** Solutions to approximately half of the exercises are included on the *Java Multimedia Cyber Classroom, Fourth Edition CD*, which also is part of *The Complete Java 2 Training Course*. For ordering instructions, please see the last few pages of this book or visit www.deitel.com.

Approximately 5300 Index Entries (with approximately 9500 Page References)

We have included an extensive index at the back of the book. This helps the student find any term or concept by keyword. The index is useful to people reading the book for the first time and is especially useful to practicing programmers who use the book as a reference. The terms in the Terminology sections generally appear in the index (along with many more index items from each chapter). Students can use the index with the Terminology sections to be sure they have covered the key material of each chapter.

"Double Indexing" of Java Live-Code™ Examples and Exercises

Java How to Program has 197 live-code™ examples and 1176 exercises (including parts). Many of the exercises are challenging problems or projects requiring substantial effort. We have "double indexed" the live-code™ examples. For every Java source-code program in the book, we took the file name with the `.java` extension, such as `LoadAudioAndPlay.java` and indexed it both alphabetically (in this case under "L") and as a subindex item under "Examples." This makes it easier to find examples using particular features. The more substantial exercises, such as "Maze Generator and Walker," are indexed both alphabetically (in this case under "M") and as subindex items under "Exercises."

Bibliography

An extensive bibliography of books, articles and Sun Microsystems Java 2 documentation is included to encourage further reading.

Software Included with Java How to Program, Fourth Edition

There are a number of for-sale Java products available. However, you do not need them to get started with Java. We wrote *Java How to Program, Fourth Edition* using only the *Java 2 Software Development Kit (J2SDK)*. For your convenience, Sun's J2SDK version 1.3.1 is included on the CD that accompanies this book. The J2SDK also can be downloaded from the Sun Microsystems Java Web site java.sun.com. With Sun's cooperation, we

also were able to include on the CD a powerful Java integrated development environment (IDE)—Sun Microsystem’s *Forté for Java Community Edition*.

Forté for Java Community Edition is a professional IDE written in Java that includes a graphical user interface designer, code editor, compiler, visual debugger and more. J2SDK 1.3.1 must be installed before installing *Forté for Java Community Edition*. If you have any questions about using this software, please read the introductory *Forté* documentation on the CD. We will provide additional information on our Web site www.deitel.com.

The CD also contains the book’s examples and an HTML Web page with links to the Deitel & Associates, Inc. Web site, the Prentice Hall Web site and the many Web sites listed in the appendices. If you have access to the Internet, this Web page can be loaded into your Web browser to give you quick access to all the resources. Finally, the CD contains Chapter 22 and Appendices E–K.

Ancillary Package for *Java How to Program, Fourth Edition*

Java How to Program, Fourth Edition has extensive ancillary materials for instructors teaching from the book. The Instructor’s Manual CD contains solutions to the vast majority of the end-of-chapter exercises and a test bank of multiple choice questions (approximately 2 per book section). In addition, we provide PowerPoint® slides containing all the code and figures in the text. You are free to customize these slides to meet your own classroom needs. Prentice Hall provides a *Companion Web Site* (www.prenhall.com/deitel) that includes resources for instructors and students. For instructors, the Web site has a Syllabus Manager for course planning, links to the PowerPoint slides and reference materials from the appendices of the book (such as the operator precedence chart, character sets and Web resources). For students, the Web site provides chapter objectives, true/false exercises with instant feedback, chapter highlights and reference materials. **[NOTE: Please do not write to us requesting the instructor’s manual. Distribution of this publication is strictly limited to college professors teaching from the book. Instructors may obtain the solutions manual only from their regular Prentice Hall representatives. We regret that we cannot provide the solutions to professionals.]**

Java 2 Multimedia Cyber Classroom, Fourth Edition (CD and Web-Based Training Versions) and The Complete Java 2 Training Course, Fourth Edition

We have prepared an interactive, CD-based, software version of *Java How to Program, Fourth Edition* called the *Java 2 Multimedia Cyber Classroom, Fourth Edition*. It is loaded with features for learning and reference. The *Cyber Classroom* is wrapped with the textbook at a discount in *The Complete Java 2 Training Course, Fourth Edition*. If you already have the book and would like to purchase the *Java 2 Multimedia Cyber Classroom, Fourth Edition* separately, please visit www.informit.com/cyberclassrooms. The ISBN# for the *Java 2 Multimedia Cyber Classroom, Fourth Edition* is 0-13-064935-x. All Deitel *Cyber Classrooms* are generally available in CD and Web-based training formats.

The CD has an introduction with the authors overviews the *Cyber Classroom*’s features. The 197 live-code™ example Java programs in the textbook truly “come alive” in the *Cyber Classroom*. If you are viewing a program and want to execute it, you simply click

on the lightning bolt icon and the program will run. You will immediately see—and hear for the audio-based multimedia programs—the program’s outputs. If you want to modify a program and see and hear the effects of your changes, simply click the floppy-disk icon that causes the source code to be “lifted off” the CD and “dropped into” one of your own directories so you can edit the text, recompile the program and try out your new version. Click the audio icon and Paul Deitel will talk about the program and “walk you through” the code.

The *Cyber Classroom* also provides navigational aids including extensive hyperlinking. The *Cyber Classroom* is browser based, so it remembers recent sections you have visited and allows you to move forward or backward among these sections. The thousands of index entries are hyperlinked to their text occurrences. You can key in a term using the “find” feature and the *Cyber Classroom* will locate its occurrences throughout the text. The Table of Contents entries are “hot”—so clicking a chapter name takes you to that chapter.

Students tell us that they particularly like the hundreds of solved problems from the textbook that are included with the *Cyber Classroom*. Studying and running these extra programs is a great way for students to enhance their learning experience.

Students and professional users of our Cyber Classrooms tell us they like the interactivity and that the Cyber Classroom is an effective reference because of the extensive hyperlinking and other navigational features. We received an email from a person who said that he lives “in the boonies” and cannot take a live course at a university, so the Cyber Classroom was the solution to his educational needs.

Professors tell us that their students enjoy using the *Cyber Classroom*, spend more time on the course and master more of the material than in textbook-only courses. We have published (and will be publishing) many other *Cyber Classroom* and *Complete Training Course* products. For a complete list of the available and forthcoming *Cyber Classrooms* and *Complete Training Courses*, see the *Deitel™ Series* page at the beginning of this book or the product listing and ordering information at the end of this book. You can also visit www.deitel.com or www.prenhall.com/deitel for more information.

Advanced Java™ 2 Platform How to Program

Our companion book—*Advanced Java 2 Platform How to Program*—focuses on the *Java 2 Platform, Enterprise Edition (J2EE)*, presents advanced Java 2 Platform Standard Edition features and introduces the *Java 2 Platform, Micro Edition (J2ME)*. This book is intended for developers and upper-level university students in advanced courses who already know Java and want a deeper treatment and understanding of the language. The book features our signature live-code™ approach of complete working programs and contains over 37,000 lines of code. The programs are more substantial than those presented in *Java How to Program, Fourth Edition*. The book expands the coverage of Java Database Connectivity (JDBC), remote method invocation (RMI), servlets and JavaBeans from *Java How to Program, Fourth Edition*. The book also covers emerging and more advanced Java technologies of concern to enterprise application developers. The Table of Contents for *Advanced Java 2 Platform How to Program* is: **Chapters**—Introduction; Advanced Swing Graphical User Interface Components; Model-View-Controller; Graphics Programming with Java 2D and Java 3D; Case Study: A Java2D Application; JavaBeans Component Model; Security; Java Database Connectivity (JDBC); Servlets; Java Server Pages (JSP); Case Study: Servlet and JSP Bookstore; Java 2 Micro Edition (J2ME) and Wireless Internet; Remote Method Invocation (RMI); Session Enterprise JavaBeans (EJBs) and Distributed Transac-

tions; Entity EJBs; Java Message Service (JMS) and Message-Driven EJBs; Enterprise Java Case Study: Architectural Overview; Enterprise Java Case Study: Presentation and Controller Logic; Enterprise Java Case Study: Business Logic Part 1; Enterprise Java Case Study: Business Logic Part 2; Application Servers; Jini; JavaSpaces; Jiro; Java Management Extensions (JMX); Common Object Request Broker Architecture (CORBA): Part 1; Common Object Request Broker Architecture (CORBA): Part 2; Peer-to-Peer Networking; **Appendices**—Creating Markup with XML; XML Document Type Definitions; XML Document Object Model (DOM); XSL: Extensible Stylesheet Language Transformations; Downloading and Installing J2EE 1.2.1; Java Community Process (JCP); Java Native Interface (JNI); Career Opportunities; Unicode.

Acknowledgments

One of the great pleasures of writing a textbook is acknowledging the efforts of the many people whose names may not appear on the cover, but whose hard work, cooperation, friendship, and understanding were crucial to the production of the book.

Other people at Deitel & Associates, Inc. devoted long hours to this project. We would like to acknowledge the efforts of our full-time Deitel & Associates, Inc. colleagues Tem Nieto, Sean Santry, Jonathan Gadzik, Kate Steinbuhler, Rashmi Jayaprakash and Laura Treibick.

- Tem Nieto is a graduate of the Massachusetts Institute of Technology. Tem teaches XML, Java, Internet and Web, C, C++ and Visual Basic seminars and works with us on textbook writing, course development and multimedia authoring efforts. He is co-author with us of *Internet & World Wide Web How to Program (Second Edition)*, *XML How to Program*, *Perl How to Program* and *Visual Basic 6 How to Program*. In *Java How to Program, Fourth Edition* Tem co-authored Chapters 11, 12, 13 and 21 and the Special Section entitled “Building Your Own Compiler” in Chapter 19.
- Sean Santry, a graduate of Boston College (Computer Science and Philosophy) and co-author of *Advanced Java 2 Platform How to Program*, edited Chapter 22 (Java Media Framework and Java Sound), helped update the programs in Chapter 15 (Multithreading), designed and implemented the Deitel Messenger networking application in Chapter 17 (Networking), helped design the optional case study on OOD/UML, reviewed the optional design patterns case study and reviewed the implementation of the elevator simulation for the OOD/UML case study.
- Jonathan Gadzik, a graduate of the Columbia University School of Engineering and Applied Science (BS in Computer Science) co-authored the optional OOD/UML case study and the optional “Discovering Design Patterns” sections. He also implemented the 3,465-line Java program that completely solves the object-oriented elevator simulation exercise presented in the OOD/UML case study.
- Kate Steinbuhler, a graduate of Boston College with majors in English and Communications, co-authored Appendix J, Career Opportunities, and managed the permissions process. Kate is moving on to law school at the University of Pittsburgh—good luck Kate! Thank you for your contributions to three Deitel publications.

- Rashmi Jayaprakash, a graduate of Boston University with a major in Computer Science, co-authored Appendix K, Unicode.
- Laura Treibick, a graduate of University of Colorado at Boulder with a major in Photography and Multimedia, created the delightful animated bug character for the implementation of the OOD/UML case study.

We would also like to thank the participants in our Deitel & Associates, Inc. College Internship Program.²

- Susan Warren, a Junior in Computer Science at Brown University, and Eugene Izumo, a Sophomore in Computer Science at Brown University, reviewed the entire *Fourth Edition*; reviewed and updated Chapter 22, Java Media Framework and Java Sound; and updated Appendix A (Java Demos) and Appendix B (Java Resources). Susan and Eugene also worked on many of the books's ancillary materials, including the solutions to the exercises, true/false questions for the companion Web site (www.prenhall.com/deitel), true/false questions for the Java 2 Multimedia Cyber Classroom and multiple choice questions for the Instructor's test bank.
- Vincent He, a Senior in Management and Computer Science at Boston College, co-authored Chapter 22, Java Media Framework and Java Sound—one of the most exciting and fun chapters in the book! We are sure you will enjoy the multimedia extravaganza Vincent created for you.
- Liz Rockett, a Senior in English at Princeton University edited and updated Chapter 22, Java Media Framework and Java Sound.
- Chris Henson, a graduate of Brandeis University (Computer Science and History), reviewed Chapter 22, Java Media Framework and Java Sound.
- Christina Carney, a Senior in Psychology and Business at Framingham State College, researched and updated the bibliography, helped prepare the Preface and performed the URL research for the OOD/UML case study and design patterns.
- Amy Gips, a Sophomore in Marketing and Finance at Boston College, updated and added URLs for applets, graphics, Java 2D and Multimedia in Appendices A and B. Amy also researched quotes for Chapter 22 and helped prepare the Preface.
- Varun Ganapathi, a Sophomore in Computer Science and Electrical Engineering at Cornell University, updated Appendix F, Creating HTML Documentation with **javadoc**.
- Reshma Khilnani, a Junior in Computer Science and Mathematics at the Massachusetts Institute of Technology, worked with Rashmi on the Unicode Appendix

We are fortunate to have been able to work on this project with the talented and dedicated team of publishing professionals at Prentice Hall. We especially appreciate the

2. The *Deitel & Associates, Inc. College Internship Program* offers a limited number of salaried positions to Boston-area college students majoring in Computer Science, Information Technology or Marketing. Students work at our corporate headquarters in Sudbury, Massachusetts full-time in the summers and part-time during the academic year. Full-time positions are available to college graduates. For more information about this competitive program, please contact Abbey Deitel at deitel@deitel.com and check our Web site, www.deitel.com.

extraordinary efforts of our computer science editor, Petra Recter and her boss—our mentor in publishing—Marcia Horton, Editor-in-Chief of Prentice-Hall’s Engineering and Computer Science Division. Camille Trentacoste did a marvelous job as production manager.

The *Java 2 Multimedia Cyber Classroom, Fourth Edition* was developed in parallel with *Java How to Program, Fourth Edition*. We sincerely appreciate the “new media” insight, savvy and technical expertise of our e-media editor-in-chief, mentor and friend Mark Taub. He and our e-media editor, Karen Mclean, did a remarkable job bringing the *Java 2 Multimedia Cyber Classroom, Fourth Edition* to publication under a tight schedule. *Michael Ruel* did a marvelous job as Cyber Classroom project manager.

We owe special thanks to the creativity of Tamara Newnam Cavallo (smart_art@earthlink.net) who did the art work for our programming tips icons and the cover. She created the delightful bug creature who shares with you the book’s programming tips. Michelle Gopen contributed the cover bug’s names.

We sincerely appreciate the efforts of our fourth edition reviewers:

Java How to Program, Fourth Edition Reviewers

Dibyendu Baksi (Sun Microsystems)
Tim Boudreau (Sun Microsystems)
Michael Bundschuh (Sun Microsystems)
Gary Ginstling (Sun Microsystems)
Tomas Pavek (Sun Microsystems)
Rama Roberts (Sun Microsystems)
Terry Hull (Sera Nova)
Ralph Johnson (“gang-of-four” co-author of the seminal book, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison Wesley, 1995)
Cameron Skinner (Embarcadero Technologies; OMG)
Michael Chonoles (Lockheed Martin Adv. Concepts; OMG)
Brian Cook (The Technical Resource Connection; OMG)
Akram Al-Rawi (Zayed University)
Charley Bay (Fronte Range Community College)
Clint Bickmore (Fronte Range Community College)
Ron Braithwaite (Nutriware)
Columbus Brown (IBM)
Larry Brown (co-author of *Core Web Programming*)
Dan Corkum (Trillium Software)
Jonathan Earl (Technical Training and Consulting)
Karl Frank (togethersoft.com)
Charles Fry (thesundancekid.org)
Kyle Gabhart (Objective Solutions)
Felipe Gaucho (Softexport)
Rob Gordon (SuffolkSoft, Inc.)
Michelle Guy (XOR)
Christopher Green (Colorado Springs Technical Consulting Group)
Kevlin Henney (Curbralan Limited)
Ethan Henry (Sitraka Software)
Faisal Kaleem (Florida International University)
Rob Kelly (SUNY)

Scott Kendall (Consultant, UML author)
 Sachin Khana (Freelance Java Programmer)
 Michael-Franz Mannion (Java Developer)
 Julie McVicar (Oakland Community College)
 Matt Mitton (Consultant)
 Dan Moore (XOR)
 Simon North (Synopsis)
 Chetan Patel (Lexisnexis)
 Brian Pontarelli (Consultant)
 Kendall Scott (Consultant, UML author)
 Craig Shofding (CAS Training Corp)
 Spencer Roberts (Titus Corporation)
 Toby Steel (CertaPay)
 Stephen Tockey (Construx Software)
 Kim Topley (Author of *Core Java Foundation Classes* and *Core Swing: Advanced Programming*, both published by Prentice Hall)
 Gustavo Toretti (Java Programmer; Campinas University)
 Michael Van Kleeck (Director of Technology, Learning.com)
 Dave Wagstaff (Sungard)

Java How to Program, Third Edition Post-Publication Reviewers

Jonathan Earl (Technical Training Consultants)
 Harry Foxwell (Sun Microsystems)
 Terry Hull (Sera Nova)
 Ron McCarty (Penn State University Behrend Campus)
 Bina Ramamurthy (SUNY Buffalo)
 Vadim Tkachenko (Sera Nova)

Under a tight time schedule, they scrutinized every aspect of the text and made countless suggestions for improving the accuracy and completeness of the presentation.

We would sincerely appreciate your comments, criticisms, corrections, and suggestions for improving the text. Please address all correspondence to:

deitel@deitel.com

We will respond immediately. Well, that's it for now. Welcome to the exciting world of Java programming. We hope you enjoy this look at leading-edge computer applications development. Good luck!

*Dr. Harvey M. Deitel
Paul J. Deitel*

About the Authors

Dr. Harvey M. Deitel, CEO of Deitel & Associates, Inc., has 40 years experience in the computing field including extensive industry and academic experience. He is one of the world's leading computer science instructors and seminar presenters. Dr. Deitel earned B.S. and M.S. degrees from the Massachusetts Institute of Technology and a Ph.D. from

Boston University. He has 20 years of college teaching experience including earning tenure and serving as the Chairman of the Computer Science Department at Boston College before founding Deitel & Associates, Inc. with his son Paul J. Deitel. He is author or co-author of several dozen books and multimedia packages and is currently writing many more. With translations published in Japanese, Russian, Spanish, Italian, Basic Chinese, Traditional Chinese, Korean, French, Polish and Portuguese, Dr. Deitel's texts have earned international recognition. Dr. Deitel has delivered professional seminars internationally to major corporations, government organizations and various branches of the military.

Paul J. Deitel, Chief Technical Officer of Deitel & Associates, Inc., is a graduate of the Massachusetts Institute of Technology's Sloan School of Management where he studied Information Technology. Through Deitel & Associates, Inc. he has delivered Internet and World Wide Web courses and programming language classes for industry clients including Sun Microsystems, EMC², IBM, BEA Systems, Visa International, Progress Software, Boeing, Fidelity, Hitachi, Cap Gemini, Compaq, Art Technology, White Sands Missile Range, NASA at the Kennedy Space Center, the National Severe Storm Laboratory, Rogue Wave Software, Lucent Technologies, Computervision, Cambridge Technology Partners, Adra Systems, Entergy, CableData Systems, Banyan, Stratus, Concord Communications and many other organizations. He has lectured on Java and C++ for the Boston Chapter of the Association for Computing Machinery, and has taught satellite-based courses through a cooperative venture of Deitel & Associates, Inc., Prentice Hall and the Technology Education Network. He and his father, Dr. Harvey M. Deitel, are the world's best-selling Computer Science textbook authors.

About Deitel & Associates, Inc.

Deitel & Associates, Inc. is an internationally recognized corporate training and content-creation organization specializing in Internet/World Wide Web software technology, e-business/e-commerce software technology and computer programming languages education. Deitel & Associates, Inc. is a member of the World Wide Web Consortium. The company provides courses on Internet and World Wide Web programming, object technology and major programming languages. The founders of Deitel & Associates, Inc. are Dr. Harvey M. Deitel and Paul J. Deitel. The company's clients include many of the world's largest computer companies, government agencies, branches of the military and business organizations. Through its publishing partnership with Prentice Hall, Deitel & Associates, Inc. publishes leading-edge programming textbooks, professional books, interactive CD-ROM-based multimedia *Cyber Classrooms*, satellite courses and Web-based training courses. Deitel & Associates, Inc. and the authors can be reached via e-mail at

deitel@deitel.com

To learn more about Deitel & Associates, Inc., its publications and its worldwide corporate on-site curriculum, see the last few pages of this book and visit:

www.deitel.com

Individuals wishing to purchase Deitel books, Cyber Classrooms, Complete Training Courses and Web-based training courses can do so through

www.deitel.com

© Copyright 2002 by Prentice Hall. All Rights Reserved.

Bulk orders by corporations and academic institutions should be placed directly with Prentice Hall. See the last few pages of this book for worldwide ordering details.

The World Wide Web Consortium (W3C)

W3C[®] Deitel & Associates, Inc. is a member of the *World Wide Web Consortium (W3C)*. The W3C was founded in 1994 “to develop common protocols for the evolution of the World Wide Web.” As a W3C member, we hold a seat on the W3C Advisory Committee (our Advisory Committee representative is our Chief Technology Officer, Paul Deitel). Advisory Committee members help provide “strategic direction” to the W3C through meetings around the world. Member organizations also help develop standards recommendations for Web technologies (such as HTML, XML and many others) through participation in W3C activities and groups. Membership in the W3C is intended for companies and large organizations. For information on becoming a member of the W3C visit www.w3.org/Consortium/Prospectus/Joining.