INTRODUCTION TO JAVA PROGRAMMING

THIRD EDITION

Y. DANIEL LIANG

Introduction to Java™ *Programming*

Third Edition

Y. Daniel Liang
Armstrong Atlantic State University
Department of Computer Science

Prentice Hall

Prentice Hall
Upper Saddle River, New Jersey 07458
http://www.prenhall.com

Library of Congress Cataloging-in-Publication Data

Liang, Y. Daniel.
Introduction to Java Programming/ Y. Daniel Liang.—3rd ed,
p. cm.
Includes bibliographical references and index.
ISBN 0-13-031997-X
1. Java (Computer program language) I. Title
QA76.73.J38 L52 2001
005.13'3—dc21 00-061970
CIP

Vice president and editorial director, ECS: *Marcia Horton*
Publisher: *Alan R. Apt*
Associate editor: *Toni D. Holm*
Editorial assistant: *Amy Todd*
Marketing manager: *Jennie Burger*
Production editor: *Pine Tree Composition*
Executive managing editor: *Vince O'Brien*
Managing editor: *David A. George*
Art director: *Heather Scott*
Art Editor: *Adam Velthaus*
Assistant art director: *John Christiana*
Manufacturing manager: *Trudy Pisciotti*
Manufacturing buyer: *Dawn Murrin*
Vice president and director of production and manufacturing, ESM: *David W. Riccardi*
Cover image: *Photodisc PD*

Printed in the United States of America
10 9 8 7 6

ISBN 0-13-031997-X

Prentice-Hall International (UK) Limited, *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall Hispanoamericana, S.A., *Mexico*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Pearson Education Asia Pte. Ltd., *Singapore*
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

About the Author

Y. Daniel Liang is the author of four successful Java books. He has taught over 40 Java courses to the university students and corporate employees.

Dr. Liang is currently a Yamacyaw professor of software engineering in the Department of Computer Science at Armstrong Atlantic State University, Savannah, Georgia. He can be reached via the Internet at **liangjava@yahoo.com,** or **liang@armstrong.edu.**

Acknowledgments

I am grateful to the readers of the previous editions, both students and instructors, who offered comments, suggestions, bug reports, and praise. Their enthusiastic support prompted the third edition, which incorporates many of their fine contributions. Without supportive readers and instructors, this edition would not have been possible.

Students and faculty at Armstrong Atlantic State University and at Purdue University at Fort Wayne have been great supporters of this ongoing project. I would like to thank Ray Greenlaw of Armstrong Atlantic State University for his support of incorporating Java into the CS curriculum, and for his tireless efforts of building a first-class computer science undergraduate program.

This book has benefited from the previous editions of my *Introduction to Java Programming*. I would like to acknowledge the following people who helped produce the previous editions: Hao Wu, Greg Geller, Michael Willig, Russell Minnich, Balaram Nair, Ben Stonebraker, C-Y Tang, Bertrand I-P Lin, Maw-Shang Chang, Ruay-Shiung Chang, Mike Sunderman, Fen English, James Silver, Mark Temte, Bob Sanders, Marta Partington, Tom Cirtin, Songlin Qiu, Tim Tate, Carolyn Linn, Alfonso Hermida, Nathan Clement, Eric Miller, Chris Barrick, John Etchison, Louisa Klucznik, Angela Denny, Randy Haubner, Robin Drake, Betsy Brown, and Susan Kindel.

For this edition, I would like to thank Alan Apt, Toni Holm, Jennie Burger, Cindy Szollose, Jake Warde, and their colleagues at Prentice-Hall for organizing and managing this project, and Patty Donovan, Robert Milch, Dan Boilard, Dana Smith, and their colleagues at Pine Tree Composition for helping to produce the book.

As always, I am indebted to my wife, Samantha, for love, support, and encouragement.

To Samantha, Michael, and Michelle

PREFACE

To the Instructor

There are three popular strategies in teaching Java. The first is to mix Java applets and graphics programming with object-oriented programming concepts. The second is to introduce object-oriented programming from the start. The third strategy is a step-by-step approach, first laying a sound foundation on programming concepts, control statements, and methods, then introducing object-oriented programming, and finally moving on to graphical user interface (GUI), applets, internationalization, multithreading, multimedia, I/O, and networking.

The first strategy, starting with GUI and applets, seems attractive, but requires substantial knowledge of object-oriented programming and a good understanding of the Java event-handling model; thus, students may never fully understand what they are doing. The second strategy is based on the notion that the objects should be introduced first because Java is an object-oriented programming language. This notion, however, does not strike a chord with students. From the more than 40 Java courses I have taught, I have concluded that introducing primary data types, control structures, and methods prepares students to learn object-oriented programming. Therefore, this text adopts the third strategy, first proceeding at a steady pace through all the necessary and important basic concepts, then moving to object-oriented programming, and then to using the object-oriented approach to build interesting GUI applications and applets with multithreading, multimedia, I/O, and networking.

The book is suited for both beginning and advanced students, depending on how it is used. This book has been used in a two-semester freshman programming course and a one-semester course in Java as a second language. It has also been used for a short training course for experienced programmers. Computer science departments, engineering departments, and management information systems departments around the world have used this book at various levels. For students with no programming experience, the emphasis should be on Part I of the book, particularly on writing the loops. You could spend the entire semester of four credit hours just on the first six chapters of the book.

The Instructor's Manual on CD-ROM is available for instructors using this book. It contains the following resources:

- Microsoft PowerPoint slides for lectures.
- Answers to review questions.
- Source code for the examples in the book.
- The new edition provides 45% more programming exercises. Instructor's Solutions are provided in the Instructor's Manual. Students will have access to the solutions of even-numbered exercises.

To obtain the Instructor's Manual, contact your Prentice-Hall sales representative.

Instructors may request sample exams directly from the author.

Microsoft PowerPoint slides, answers to review questions, solutions to even-numbered programming exercises, and source code for the examples in the book are also available at the book's companion Web site at **www.cs.armstrong.edu/liang/intro3e.html** and **www.prenhall.com/liang**

Pedagogical Features of the Book

Introduction to Java Programming Third Edition uses the following elements to get the most out of the material:

- **Objectives** lists what students should have learned from the chapter. This will help them to determine whether they have met the objectives after completing the chapter.
- **Introduction** opens the discussion with a brief overview of what to expect from the chapter.
- Programming concepts are taught by representative **Examples**, carefully chosen and presented in an easy-to-follow style. Each example is described, and includes the source code, a sample run, and an example review. The source code of the examples is contained in the companion CD-ROM.
- Each program is complete and ready to be compiled and executed. The sample run of the program is captured from the screen to give students a live presentation of the example. Reading these examples is much like entering and running them on a computer.
- **Chapter Summary** reviews the important subjects that students should understand and remember. It helps them to reinforce the key concepts they have learned in the chapter.
- **Review Questions** help students to track their progress and evaluate their learning.
- **Programming Exercises** at the end of each chapter provide students with opportunities to apply the skills on their own. The trick of learning programming is practice, practice, and practice. To that end, the book provides a large number of exercises.
- **Notes** and **Tips** are inserted throughout the text to offer valuable advice and insight on important aspects of program development.

NOTE
Provides additional information on the subject and reinforces important concepts.

TIP
Teaches good programming style and practice, and helps students steer away from the pitfalls of programming errors.

What's New in This Edition

This book expands and improves upon the second edition of *Introduction to Java Programming*. The major changes are as follows:

- The use of UML enhances the treatment of object-oriented design and programming. UML graphical notations are used to describe classes and their relationships throughout the book.
- Beginning with Chapter 8, "Getting Started with Graphics Programming," all the AWT user interface components are replaced with state-of-the-art Swing components.
- Chapter 12, "Internationalization," is brand-new. It introduces date, time, and number formatting, and the development of Java programs for international audiences.
- Several new case studies are provided to give more examples for learning the fundamentals of programming, such as writing loops.
- Nonessential sections are marked optional and can be skipped or covered later with no adverse effect on the students' understanding of subsequent chapters. These sections include such topics as recursion, advanced layout managers, and resource bundles.
- Several new appendices provide readers with additional background information.

To the Student

There is nothing more important to the future of computing than the Internet. There is nothing more exciting on the Internet than Java. A revolutionary programming language developed by Sun Microsystems, Java has become the de facto standard for cross-platform applications and programming on the World Wide Web.

Before Java, the Web was used primarily for viewing static information on the Internet using HTML, a marked-up language for document layout and for linking documents over the Internet. Java programs can be embedded in an HTML page and downloaded by Web browsers to bring live animation and interactive applications to Web clients.

Java is a full-featured, general-purpose programming language that is capable of developing robust mission-critical applications. In the last three years, Java has gained enormous popularity and has quickly become the most popular and successful programming language. Today, it is used not only for Web programming, but also for developing standalone applications. Many companies that once considered Java to be more hype than substance are now using it to create distributed applications accessed by customers and partners across the Internet. For every new project being developed today, companies are asking how they can use Java to make their work easier.

Java's Design and Advantages

Java is an object-oriented programming language. Object-oriented programming is a favored programming approach that has replaced traditional procedure-based programming techniques. An object-oriented language uses abstraction, encapsulation, inheritance, and polymorphism to provide great flexibility, modularity, and reusability for developing software.

Java is platform-independent. Its programs can run on any platform with a Java Virtual Machine, a software component that interprets Java instructions and carries out associated actions.

Java is distributed. Networking is inherently built-in. Simultaneous processing can occur on multiple computers on the Internet. Writing network programs is treated as simple data input and output.

Java is multithreaded. Multithreading is the capability of a program to perform several tasks simultaneously, such as downloading a video file while playing the video at the same time. Multithreading is particularly useful in graphical user interfaces (GUI) and network programming. Multithread programming is smoothly integrated in Java. In other languages, you can only enable multithreading by calling procedures that are specific to the operating system.

Java is secure. Computers become vulnerable when they are connected with other computers. Viruses and malicious programs can damage your computer. Java is designed with multiple layers of security that ensure proper access to private data and restrict access to disk files.

Java's Versatility

Stimulated by the promise of writing programs once and running them anywhere, Java has become the most important programming language. IBM, Sun, and Apple, and many other vendors are working to integrate the Java Virtual Machine with their operating systems so that Java programs can run directly and efficiently on the native machine. Java programs run on full-featured computers, and also on consumer electronics and appliances.

Because of its great potential to unite existing legacy applications written on different platforms so that they can run together, Java has been perceived as a universal front end for the enterprise database. The leading database companies, IBM, Oracle, Sybase, and Informix, have extended their commitment to Java by integrating it into their products. Oracle, for example, enables Java applications to run on its server, and to deliver a complete set of Java-based development tools supporting the integration of current applications with the Web.

Learning Java

To the first time programmers, learning Java is like learning any high-level programming languages. The foundation of learning programming is to develop critical skills of formulating programmatic solutions for the programs and to translate the solutions into programs using the selection statements, loops, and methods.

Applying the concept of abstraction in the design and implementation of software projects is the key to developing software. The overriding objective of this book, therefore, is to teach students to use many levels of abstraction in solving problems and to see problems in small and in large.

Students with no programming experience should take a slow-pace approach in Part I of the book. I recommend you to complete all the exercises in Chapter 3 and 4 before moving to Chapter 5. Students new to object-oriented programming may need some time to become familiar with the concept of objects and classes. Once the principles are mastered, programming in Java is easy and productive. Students who know object-oriented programming languages like C++ and Smalltalk will find it easier to learn Java. In fact, you will find that Java is simpler than C++ and Smalltalk in many aspects.

Organization of the Book

This book is divided into four parts that, taken together, form a comprehensive introductory course on Java programming. Because knowledge is cumulative, the early chapters provide the conceptual basis for understanding Java and guide students through simple examples and exercises; subsequent chapters progressively present Java programming in detail, culminating with the development of comprehensive Java applications. The appendixes contain a mixed bag of topics, including an HTML tutorial.

Part I: Fundamentals of Programming

The first part of the book is a stepping stone that will prepare you to embark on the journey of learning Java. You will begin to know Java, and will learn how to write simple Java programs with primitive data types, control statements, and methods.

Chapter 1, "Introduction to Java," gives an overview of the major features of Java: object-oriented programming, platform-independence, Java bytecode, security, performance, multithreading, and networking. The chapter also introduces how to create, compile, and run Java applications and applets. Simple examples of writing applications and applets are provided, along with a brief anatomy of programming structures.

Chapter 2, "Primitive Data Types and Operations," introduces primitive data types, operators, and expressions. Important topics include identifiers, variables, constants, assignment statements, primitive data types, operators, and shortcut operators. Java programming style and documentation are also addressed.

Chapter 3, "Control Statements," introduces decision and repetition statements. Java decision statements include various forms of `if` statements, and the `switch` statement. Java repetition statements include the `while` loop, the `do` loop, and the `for` loop. The keywords `break` and `continue` are discussed.

Chapter 4, "Methods," introduces method creation, calling methods, passing parameters, returning values, method overloading, and recursion. Applying the concept of abstraction is the key to developing software. The chapter also introduces the use of method abstraction in problem-solving. The `Math` class for performing basic math operations is introduced.

Part II: Object-Oriented Programming

In the book's second part, object-oriented programming is introduced. Java is a class-centric, object-oriented programming language that uses abstraction, encapsulation, inheritance, and polymorphism to provide great flexibility, modularity, and reusability in developing software. You will learn programming with objects and classes, arrays and strings, and class inheritance.

Chapter 5, "Programming with Objects and Classes," begins with objects and classes. The important topics include defining classes, creating objects, using constructors, passing objects to methods, instance and class variables, and instance and class methods, analyzing relationships among classes, and using the UML graphical notations to describe classes. Several examples are provided to demonstrate the power of the object-oriented programming approach. Students will learn the benefits (abstraction, encapsulation, and modularity) of object-oriented programming from these examples. There are more than 500 predefined Java classes grouped in several packages. Starting with this chapter, students will gradually learn how to use Java classes to develop their own programs. The classes `String`, `StringBuffer`, and `StringTokenizer` for storing and processing strings are introduced.

Chapter 6, "Class Inheritance," teaches how an existing class can be extended and modified as needed. Inheritance is an extremely powerful programming technique, further extending software reusability. Java programs are all built by extending predefined Java classes. The major topics include defining subclasses, using the keywords `super` and `this`, using the modifiers `protected`, `final` and `abstract`, polymorphism and dynamic binding, and casting objects. This chapter introduces the `Object` class, which is the root of all Java classes. You will learn to use abstract classes, and interfaces.

Chapter 7, "Arrays and Vectors," explores an important structure: arrays for processing data in lists and tables. You will learn how to declare, initialize, and copy arrays. You will also learn how to use wrapper classes to encapsulate primitive data type values in objects for use in generic programming. This chapter introduces the `Vector` class, which can be used to store an unspecified number of elements.

Part III: Graphics Programming

The third part of the book introduces Java graphics programming. Major topics include event-driven programming, creating graphical user interfaces, and writing applets. You will learn the architecture of Java graphics programming API and use the user interface components to develop graphics applications and applets.

Chapter 8, "Getting Started with Graphics Programming," introduces the concepts of Java graphics programming using Swing components. Topics include the Swing class hierarchy, event-driven programming, frames, panels, and simple layout managers (`FlowLayout`, `GridLayout`, and `BorderLayout`). The chapter also introduces drawing geometric figures in the graphics context.

Chapter 9, "Creating User Interfaces," introduces the user interface components: buttons, labels, text fields, text areas, combo boxes, lists, check boxes, radio buttons, menus, scrollbars, and scroll panes. Today's client/server and Web-based applications use a graphical user interface (GUI, pronounced "goo-ee"). Java has a rich set of classes to help you build GUIs.

Chapter 10, "Applets and Advanced Graphics," takes an in-depth look at applets, discussing applet behavior and the relationship between applets and other Swing classes. Applets are a special kind of Java class that can be executed from the Web browser. Students will learn how to convert applications to applets, and vice versa, and how to run programs both as applications and as applets. The chapter also introduces two advanced layout mangers (`CardLayout` and `GridBagLayout`) and the use of no layout. Advanced examples of handling mouse and keyboard events are provided. You will learn how to package and deploy Java applications and applets.

Part IV: Developing Comprehensive Projects

The book's final part is devoted to several advanced features of Java programming. You will learn how to use these features to develop comprehensive programs; for example, using exception handling to make your program robust, using multithreading to make your program more responsive and interactive, incorporating sound and images to make your program user-friendly, using input and output to manage and process a large quantity of data, and creating client/server applications with Java networking support.

Chapter 11, "Exception Handling," teaches students how to define exceptions, throw exceptions, and handle exceptions so that their programs can either continue to run or terminate gracefully in the event of runtime errors. The chapter discusses predefined exception classes, and gives examples of creating user-defined exception classes.

Chapter 12, "Internationalization," introduces the development of Java programs for international audiences. You will learn how to format dates, numbers, currencies, and percentages for different regions, countries, and languages. You will also learn how to use resource bundles to define which images and strings are used by a component depending on the user's locale and preferences.

Chapter 13, "Multithreading," introduces threads, which enable the running of multiple tasks simultaneously in one program. Students will learn how to use the `Thread` class and the `Runnable` interface to launch separate threads. The chapter also discusses thread states, thread priority, thread groups, and the synchronization of conflicting threads.

Chapter 14, "Multimedia," teaches how to incorporate sound and images to bring live animation to Java programs. Various techniques for smoothing animation are introduced.

Chapter 15, "Input and Output," introduces input and output streams. Students will learn the class structures of I/O streams, byte and character streams, file I/O streams, data I/O streams, print streams, delimited I/O, random file access, and interactive I/O.

Chapter 16, "Networking," introduces network programming. Students will learn the concept of network communication, stream sockets, client/server programming, and reading data files from the Web server.

Appendixes

This part of the book covers a mixed bag of topics. Appendix A lists Java keywords. Appendix B gives tables of ASCII characters and their associated codes in decimal and in hex. Appendix C shows the operator precedence. Appendix D summarizes Java modifiers and their usage. Appendix E introduces number systems and conversions among binary, decimal, and hex numbers. Appendix F introduces HTML basics. Appendix G lists UML Graphical Notations for describing classes and their relationships. Appendix H introduces packages for grouping classes. Appendix I gives a JBuilder 3.5 tutorial. Finally, Appendix J provides a glossary of key terms used in the text.

Companion Web Site for the Book

The companion Web site for the book can be accessed from **www.cs.armstrong.edu/liang/intro3e.html** or **www.prenhall.com/liang**. The Web site contains the following resources:

- Microsoft PowerPoint slides for lectures
- Answers to review questions
- Solutions to even-numbered programming exercises
- Source code for the examples in the book
- Interactive on-line self test
- How to obtain and install Java 2 SDK v1.3
- A JBuilder 4 Tutorial
- Java Data Structures
- Errata
- FAQs

CONTENTS AT A GLANCE

Table of Contents