Introduction to Java™ Programming

Second Edition

Y. Daniel Liang

Series in Programming and Development

Introduction to Java™ Programming

Second Edition

Y. Daniel Liang

QUE E&T

An Imprint of Macmillan Computer Publishing
201 W. 103rd St., Indianapolis, IN 46290

INTRODUCTION TO JAVA PROGRAMMING, SECOND EDITION

Library of Congress Catalog No.: 98-88887

ISBN: 1-58076-255-7

This book is sold as is, without warranty of any kind, either express or implied, respecting the contents of this book, including but not limited to implied warranties for the book's quality, performance, merchantability, or fitness for any particular purpose. Neither Que Corporation nor its dealers or distributors shall be liable to the purchaser or any other person or entity with respect to any liability, loss, or damage caused or alleged to be caused directly or indirectly by this book.

02 01 00 4 3

Interpretation of the printing code: the rightmost double-digit number is the year of the book's printing; the rightmost single-digit number, the number of the book's printing. For example, a printing code of 99-1 shows that the first printing of the book occurred in 1999.

Composed in *AGaramond* and *MCPdigital* by Que® Education and Training.

Trademark Acknowledgments

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Que Education & Training cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Java is a registered trademark of Sun Microsystems. Microsoft and Windows are registered trademarks of Microsoft Corporation.

Screens reproduced in this book were created using Collage Plus from Inner Media, Inc., Hollis, NH.

PUBLISHER:
Robert Linsky

EXECUTIVE EDITOR:
Randy Haubner

DIRECTOR OF PRODUCT MARKETING:
Susan L. Kindel

OPERATIONS MANAGER:
Christine Moos

DEVELOPMENT EDITOR:
Songlin Qiu

PROJECT EDITOR:
Tim Tate

COPY EDITOR:
Carolyn Linn

TECHNICAL EDITOR:
Alfonso Hermida

SOFTWARE SPECIALIST:
Angela Denny

COVER DESIGNER:
Nathan Clement

BOOK DESIGNER:
Louisa Klucznik

INDEXER:
Chris Barrick

PROOFREADER:
John Etchison

LAYOUT TECHNICIAN:
Eric S. Miller

ABOUT THE AUTHOR

Y. Daniel Liang holds B.S. and M.S. degrees in computer science from Fudan University in Shanghai and a Ph.D. degree in computer science from the University of Oklahoma. He has published numerous papers in international journals and has taught Java courses nationally and internationally. He has consulted in the areas of algorithm design, client/server computing, and database management.

Dr. Liang is currently an associate professor in the Department of Computer Science at Indiana University-Purdue University at Fort Wayne, where he twice received the Excellence in Research Award from the School of Engineering, Technology, and Computer Science. He can be reached via the Internet at **liangy@ipfw.edu**.

ACKNOWLEDGMENTS

I am grateful to the readers of the first edition who offered comments, suggestions, bug reports, and praise. Their enthusiastic support prompted the second edition, which incorporates many of their fine contributions.

Many students helped to prepare this book. Hao Wu drew all the diagrams in the early draft and helped test some examples. Michael Willig, Russell Minnich, Balaram Nair, and Ben Stonebraker helped to correct many typos and errors for the first edition. Mike Sunderman of Raytheon Systems offered several corrections for the second edition.

Dr. Fen English, Dr. James Silver, and many colleagues in the Department of Computer Science have supported this project and offered valuable suggestions and assistance. It was a pleasure to teach the introductory Java course with Mark Temte and Bob Sanders.

I was fortunate to work with an excellent team at Que Education & Training, whose contributions greatly improved the content and presentation of this book. For the first edition, I learned many valuable tips from Marta Partington, Development Editor, who guided me through the revision of the text, and Tom Cirtin, who did a super job as Production Editor. For the second edition, Songlin Qiu, the Development Editor, coordinated the entire development and production process and offered good advice to improve and enhance the text. Tim Tate, the Project Editor, helped in producing the book. Carolyn Linn, the Copy Editor, helped polish my English. Alfonso Hermida was the Technical Editor for both editions; he reviewed all source code and offered good suggestions. Once again, the entire development and production teams for both editions have my thanks for developing and producing the finished product.

I would also like to thank Randy Haubner, the Executive Editor, who organized and oversaw the project; Robin Drake, who helped develop the instructor's manual for the first edition; and Betsy Brown and Susan Kindel for their support of the project.

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

To Samantha, Michael, and Michelle,
with love

INTRODUCTION

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 since its inception in May 1995.

Before Java, the Web was used primarily for viewing static information on the Internet using HTML, a script 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, which is capable of developing robust and 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, Java is used not only for Web programming, but also to develop standalone applications. Many companies that considered Java to be more hype than substance before are now using Java to create distributed applications accessed by customers and partners across the Internet. For every new project being developed today, companies are asking how Java could be used 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 today that replaces the 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. Java programs can run on any machine with any operating system that supports the 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 in Java.

Java is multithreaded. Multithreading is the capability of a program to perform several tasks simultaneously; for example, a program can download a video file while playing the video at the same time. Multithreading is particularly useful in graphical user interface (GUI) and network programming. Multithread programming is smoothly integrated in Java. In other languages, you have to call procedures that are specific to the operating system to enable multithreading.

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

Java's Versatility

Stimulated by the promise of writing programs once and running them anywhere, the computer industry gave Java unqualified endorsement. IBM, Sun, Apple, Microsoft, 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 not only on full-featured computers, but also on consumer electronics and appliances.

There is a great potential for Java to unite existing legacy applications written on different platforms to run together. Java has been perceived as a universal front end for the enterprise database. The leading database companies, such as IBM, Oracle, Sybase, and Informix, have extended their commitment to Java by integrating Java into their products. Oracle, for example, plans to enable native 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

The key to developing software is applying the concept of abstraction in the design and implementation of the software project. The overriding objective of this book is, therefore, to teach how to solve problems using many levels of abstraction, and to teach how to see problems in small and in large.

Students were the source of inspiration for this book. I learned to teach programming from my students. Students told me they wanted a book that uses easy-to-follow examples to teach programming concepts. In the summer of 1996, I was looking for a Java text. I encountered many reference books and several books converted from C and C++ texts on the market. I could not find the kind of book that I was looking for, so the idea was born to write a book to teach basic Java concepts using good examples.

This book covers major topics in Java programming, including programming structures, methods, objects, classes, inheritance, AWT, applets, exception handling, multithreading, multimedia, I/O, and networking. For students new to object-oriented programming, it takes some time to become familiar with the concept of objects and classes. Once you master the principles, programming in Java is easy and productive. For students who know object-oriented programming languages, such as C++ and Smalltalk, learning Java is easier. In fact, Java is simpler than C++ and Smalltalk in many aspects.

The book is completely based on JDK 1.1 and is compatible with JDK 1.2. All the examples were tested using JDK 1.1 and JDK 1.2 on Windows 95 and on Sun Solaris using no deprecated API. The companion CD-ROM contains the JDK 1.2

beta 4 and the source code of all the examples in the text. You can download the latest version of JDK 1.2 from JavaSoft at **www.javasoft.com**.

Coming soon are *Introduction to Java Programming with Visual J++ 6.0* and *Introduction to Java Programming with JBuilder 2.* Both books are written by Y. Daniel Liang and will be published by Que Education and Training.

To the Instructor

There are three popular strategies in teaching Java. The first strategy is to mix Java applets and graphics programming with object-oriented programming concepts. The second strategy 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 elements, control structures, and methods, and then moving on to the graphical user interface, applets, multithreading, multimedia, I/O, and networking.

The first strategy, starting with GUI and applets, seems attractive, but requires substantial knowledge of OOP and full understanding of the JDK event-handling model; thus, students may never fully understand what they are doing. The second strategy is based on the perception that the objects should be introduced first because Java is an object-oriented programming language. That perception, however, does not strike a chord with the students. From more than 20 Java courses I taught in the past, I found that introducing primitive data types, control structures, and methods helps prepare students to learn object-oriented programming. Therefore, this text adopts the third strategy, proceeding at a steady pace through all necessary and important basic concepts first, then quickly moving to object-oriented programming and then to building interesting GUI applications and applets with multimedia and networking using the object-oriented approach.

This book is primarily intended for introductory programming courses; however, it can also be used for teaching Java as a second language, or for a short training course for experienced programmers. The book contains more material than can be covered in a single semester. You can cover the first 12 chapters, then use the remaining chapters as time permits.

The *Instructor's Resource* CD-ROM is available for adopters of this book. It contains the following resources:

- Lecture notes with suggested teaching strategies and activities
- Microsoft PowerPoint slides for lectures
- Answers to chapter reviews
- Solutions to programming exercises
- Over 400 multiple choice and true or false questions and answers covering all of the chapters of this book in sequence

To receive the Instructor's CD-ROM, contact your Macmillan sales representative.

Key Features of This Book

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

- **Objectives** lists what students learn from the chapter. This helps students to determine whether they have met these 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 also helps students reinforce the key concepts they have learned in the current chapter.
- **Chapter Review** helps students to track progress and evaluate learning.
- **Programming Exercises** at the end of each chapter provide students opportunities to apply the skills on their own. The trick of learning programming is practice, practice, and practice. To that end, this book provides a large number of exercises.
- **Notes**, **Tips**, and **Cautions** are inserted throughout the text to offer students 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.

CAUTION
Helps students steer away from the pitfalls of programming errors.

What's New in the Second Edition

The second edition expands and improves upon the first edition. The major changes are as follows:

- The text has been updated to conform to the JDK 1.2 standards. All the source code can compile and run in JDK 1.2 with no deprecated methods.
- A new section on inner classes was added in Chapter 7, "Class Inheritance." A new section on using `Scrollbar` was added in Chapter 9, "Creating User Interfaces." A new section on using `FileDialog` was added in Chapter 14, "Input and Output."
- New naming conventions for AWT objects are introduced and used consistently to improve code readability.
- A new appendix, E, titled "An HTML Tutorial," was added to introduce basics of writing HTML pages. A new appendix, F, titled "Using the Companion CD-ROM," was added to guide the students in installing JDK 1.2 and in using the source code in the text.
- A new glossary is provided to summarize key terms and definitions used in the text.
- The appendixes on JBuilder in the first edition were removed. Interested readers should refer to the author's new book, *Introduction to Java Programming with JBuilder 2.*

Organization of This Book

This book is divided into four Parts that, taken together, form a comprehensive 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 and culminate in teaching the development of comprehensive Java applications. The appendixes contain a mixed bag of topics that include an HTML tutorial.

Part I: Fundamentals of Java Programming

This Part is a steppingstone to prepare you to embark on the journey of learning Java. You will start to know Java and learn to write simple Java programs with primitive data types, control structures, 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. This chapter also gives a brief introduction to Java applications and applets, and discusses their similarities and differences. Simple examples for writing applications and applets are provided, along with a brief anatomy of programming structures.

Chapter 2, "Java Building Elements," 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 Structures," introduces decision and repetition statements. The Java decision statements include various forms of `if` statements, the `switch` statement, and the shortcut `if` statement. The repetition statements include the `for` loop, the `while` loop, and the `do` loop. Students will also learn the role of the keywords `break` and `continue`, which are used in the control structures.

Chapter 4, "Methods," introduces method creation, calling methods, passing parameters, returning values, method overloading, and recursion. The key to developing software is applying the concept of abstraction. This chapter also introduces the concept of method abstraction in problem solving.

Part II: Object-Oriented Programming

This Part introduces object-oriented programming. Java is a class-centric, object-oriented programming language that uses abstraction, encapsulation, inheritance, and polymorphism to provide great flexibility, modularity, and reusability for 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. Many 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 to use Java classes to develop their own programs. The `Math` class for performing basic math operations is introduced in this chapter.

Chapter 6, "Arrays and Strings," explores two important structures: arrays for processing data in lists and tables, and strings using the `String`, `StringBuffer`, and `StringTokenizer` classes. Arrays are treated as objects in Java. Strings are treated as a special kind of array in many high-level languages, but strings are not related to arrays in Java. Strings and arrays are used very differently in Java.

Chapter 7, "Class Inheritance," teaches how to extend an existing class and modify it 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 `final` and `abstract`, and casting objects and interfaces. This chapter introduces the `Object` class, which is the root of all Java classes. Students will learn primitive data type wrapper classes to encapsulate primitive data type values in objects and learn `Date`, `Calendar`, `TimeZone`, `Locale`, and `DateFormat` and their subclasses to process dates and times.

Part III: Graphics Programming

This Part 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 the AWT (Abstract Window Toolkit). Topics include the AWT class hierarchy, event-driven programming, frames, panels, canvases, and simple layout managers (`FlowLayout`, `GridLayout`, and `BorderLayout`). This chapter also introduces drawing geometric figures in the graphics context.

Chapter 9, "Creating User Interfaces," introduces the user interface components, including buttons, labels, text fields, text areas, choices, lists, check boxes, check box groups, menus and scrollbars. 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, and discusses applet behaviors and an applet's relationship with other AWT classes. Applets are a special kind of Java class that can be executed from the Web browser. Students will learn to convert applications to applets and vice versa, and learn to run programs both as applications and as applets. This chapter also introduces two advanced layout managers (`CardLayout` and `GridBagLayout`) and the use of no layout. Advanced examples on handling mouse and keyboard events are also provided.

Part IV: Developing Comprehensive Projects

This Part is devoted to several advanced features of Java programming. You will learn to develop comprehensive programs using these features, such as 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 the programs can continue to run or terminate gracefully in the event of runtime errors. The chapter discusses predefined exception classes, and gives examples for creating user-defined exception classes.

Chapter 12, "Multithreading," introduces threads, which enable the running of multiple tasks simultaneously in one program. Students will learn 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 13, "Multimedia," teaches students to incorporate sound and images to bring live animation to Java programs. The chapter also discusses various techniques for smoothing animation.

Chapter 14, "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 15, "Networking," introduces network programming. Students will learn the concepts of network communication, stream sockets, client/server programming, and reading data files from the Web server.

Appendixes

This section 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 HTML. Appendix F contains information for using the companion CD-ROM. Finally, Appendix G provides a glossary of key terms found in the text.

What's on the Companion CD-ROM

The companion CD-ROM contains JDK 1.2 beta 4 for Windows 95, 98, and NT, and the source code for all the examples in the text. Please refer to Appendix F, "Using the Companion CD-ROM," for instructions on installing JDK 1.2 and using the example source code.

CONTENTS AT A GLANCE

TABLE OF CONTENTS

PART I FUNDAMENTALS OF JAVA PROGRAMMING