



Preface

This book is a desktop quick reference for Java™ programmers, designed to sit faithfully by your keyboard while you program. Part I of the book is a fast-paced, “no-fluff” introduction to the Java programming language and the core APIs of the Java platform. Part II is a quick-reference section that succinctly details most classes and interfaces of those core APIs. The book covers Java 1.0, 1.1, 1.2, 1.3, and 1.4.

Changes in the Fourth Edition

Once again, the Java platform has grown dramatically larger with the release of Java 1.4, and this book has grown in response. Some of the important new features of Java 1.4 (and of this book) are:

Assert statement

The Java language has been extended to support assertions with the `assert` statement. This new statement is documented in Chapter 2.

JavaBeans persistence

JavaBeans and related objects can now be serialized to XML documents. See `java.beans.XMLEncoder` in Chapter 9 for more information.

New I/O API

Java 1.4 includes a new API for high-performance, nonblocking file and network input and output. See the `java.nio` package and its subpackages in Chapter 14. Chapter 4 contains a number of examples of this important new API.

Certification path API

The `java.security.cert` package has been extended with new classes and interfaces for creating certificate chains, or “certification paths,” which are commonly used in network authentication.

Logging API

The new `java.util.logging` package defines a powerful and flexible logging framework for Java applications.

Preferences API

`java.util.prefs` defines an API that allows applications to persistently store and query user preference values and systemwide configuration options.

Pattern matching with regular expressions

One more new utility package, `java.util.regex`, provides support for textual pattern matching with Perl-style regular expressions.

Secure network sockets

The Java Secure Sockets Extension (JSSE) API defined by the new `javax.net` and `javax.net.ssl` packages provides support for secure networking with the SSL and TLS protocols.

Network authentication and authorization

The Java Authentication and Authorization Service (JAAS) is defined by the `javax.security.auth` package and its subpackages. JAAS enables a Java application to securely establish the identity of a user and run code under a security policy based on the set of permissions granted to that user.

XML parsing and transformations

The Java API for XML Processing (JAXP) is defined by the `javax.xml.parsers` package and the `javax.xml.transform` package and subpackages. JAXP provides facilities for parsing XML documents using the SAX and DOM APIs and for transforming the content of those documents using XSLT. Along with JAXP, the DOM and SAX APIs have also been made part of the Java 1.4 platform. You'll find them in the `org.w3c.dom` package in Chapter 23 and in the `org.xml.sax` package and subpackages in Chapter 24.

You'll find examples illustrating how to use most of these new APIs in Chapter 4.

In addition to all the new content, there have been a few organizational changes to the book. In previous editions, the quick reference was organized with one package to a chapter. This edition documents 46 distinct packages, which would make for an excessive number of chapters. In this edition, therefore, related packages (those with a common prefix) are grouped into a single chapter, shortening the quick reference to a more manageable 15 chapters. Because the quick reference has a purely alphabetical organization, however, the chapter boundaries are largely irrelevant, and you can find what you need simply by flipping through the quick reference as you would flip through a dictionary or phone book.

Another change caused by the dramatically increased number of packages is that I was forced to cut the package hierarchy figures that appeared at the start of each chapter in previous editions. These figures were all carefully hand-drawn and have become an increasingly large burden on the technical illustration staff at O'Reilly & Associates, Inc. Furthermore, the figures simply haven't proven to be as useful as they once seemed. In this edition, I decided that the figures' benefit simply didn't justify their cost. If you are one of the minority of readers who was fond of those diagrams, I apologize for their removal.

There are two new features of the quick reference that should compensate for the loss of the package hierarchy diagrams. First, the reference entry for each package now includes a listing of all interfaces and classes in the package. The entries in this list are grouped by category (interfaces, classes, and exceptions, for example) and by hierarchy. This listing, while not graphical, provides exactly the same information as the old hierarchy diagrams. Second, the class hierarchy subsection of each class and interface quick reference has been converted from an awkward textual format to an improved graphical format.

Contents of This Book

The first eight chapters of this book document the Java language, the Java platform, and the Java development tools that are supplied with Sun's Java SDK (software development kit). The first four chapters are essential; the next four cover topics of interest to some, but not all Java programmers.

Chapter 1: Introduction

This chapter is an overview of the Java language and the Java platform that explains the important features and benefits of Java. It concludes with an example Java program and walks the new Java programmer through it line by line.

Chapter 2: Java Syntax From the Ground Up

This chapter explains the details of the Java programming language. It is a long and detailed chapter. Experienced Java programmers can use it as a language reference. Programmers with substantial experience with languages such as C and C++ should be able to pick up Java syntax by reading this chapter. The chapter does not assume years of programming experience, however, and does not even require familiarity with C or C++. Even beginning programmers, with only a modest amount of experience should be able to learn Java programming by studying this chapter carefully.

Chapter 3: Object-Oriented Programming in Java

This chapter describes how the basic Java syntax documented in Chapter 2 is used to write object-oriented programs in Java. The chapter assumes no prior experience with OO programming. It can be used as a tutorial by new programmers or as a reference by experienced Java programmers.

Chapter 4: The Java Platform

This chapter is an overview of the essential Java APIs covered in this book. It contains numerous short examples that demonstrate how to perform common tasks with the classes and interfaces that comprise the Java platform. Programmers who are new to Java, and especially those who learn best by example, should find this a valuable chapter.

Chapter 5: Java Security

This chapter explains the Java security architecture that allows untrusted code to run in a secure environment from which it cannot do any malicious damage to the host system. It is important for all Java programmers to have at least a passing familiarity with Java security mechanisms.

Chapter 6: JavaBeans

This chapter documents the JavaBeans™ component framework and explains what programmers need to know to create and use the reusable, embeddable Java classes known as beans.

Chapter 7: Java Programming and Documentation Conventions

This chapter documents important and widely adopted Java programming conventions and also explains how you can make your Java code self-documenting by including specially formatted documentation comments.

Chapter 8: Java Development Tools

The Java SDK shipped by Sun includes a number of useful Java development tools, most notably the Java interpreter and the Java compiler. This chapter documents those tools.

These first eight chapters teach you the Java language and get you up and running with the Java APIs. The bulk of the book, however, is the API quick reference, Chapters 9 through 24, which is a succinct but detailed API reference formatted for optimum ease of use. Please be sure to read Chapter 1, which appears at the beginning of the reference section; it explains how to get the most out of this section. Also, please note that the quick-reference chapters are followed by one final chapter entitled “Class, Method, and Field Index”. This special index allows you to look up the name of a class and find the package it is defined in or look up the name of a method or field and find the class it is defined in.

Related Books

O'Reilly publishes an entire series of books on Java programming, including several companion books to this one. The companion books are:

Java Enterprise in a Nutshell

This book is a succinct tutorial and quick reference for the Java “Enterprise” APIs such as JDBC, RMI, JNDI, and CORBA.

Java Foundation Classes in a Nutshell

This book is a tutorial and quick reference for the graphics, graphical user interface, and related APIs of the Java platform. It includes coverage of Applets, AWT, Java2D, and Swing.

Java Examples in a Nutshell

This book contains hundreds of complete, working examples illustrating many common Java programming tasks using the core, enterprise, and foundation classes APIs. *Java Examples in a Nutshell* is like Chapter 4 of this book, greatly expanded in breadth and depth, and with all the code snippets fully fleshed out into working examples. This is a particularly valuable book for readers who learn well by experimenting with existing code.

J2ME in a Nutshell

This book is a tutorial and quick reference for the graphics, networking, and database APIs of the Java 2 Micro Edition (J2ME) platform.

You can find a complete list of Java books from O'Reilly at <http://java.oreilly.com/>. Books that focus on the core Java APIs, as this one does, include:

Learning Java, by Pat Niemeyer and Jonathan Knudsen

A comprehensive tutorial introduction to Java, with an emphasis on client-side Java programming.

Java Threads, by Scott Oaks and Henry Wong

Java makes multithreaded programming easy, but doing it right can still be tricky. This book explains everything you need to know.

Java I/O, by Elliotte Rusty Harold

Java's stream-based input/output architecture is a thing of beauty. This book covers it in the detail it deserves.

Java Network Programming, by Elliotte Rusty Harold

This book documents the Java networking APIs in detail.

Java Security, by Scott Oaks

This book explains the Java access-control mechanisms in detail and also documents the authentication mechanisms of digital signatures and message digests.

Java Cryptography, by Jonathan Knudsen

Thorough coverage of the Java Cryptography Extension, the `javax.crypto.*` packages, and everything you need to know about cryptography in Java.

Developing Java Beans, by Robert Englander

A complete guide to writing components that work with the JavaBeans API.

Java Programming Resources Online

This book is a quick reference designed for speedy access to frequently needed information. It does not, and cannot, tell you everything you need to know about Java. In addition to the books listed earlier, there are several valuable (and free) electronic sources of information about Java programming.

Sun's main web site for all things related to Java is <http://java.sun.com/>. The web site specifically for Java developers is <http://developer.java.sun.com/>. Much of the content on this developer site is password-protected, and access to it requires (free) registration.

Sun distributes electronic documentation for all Java classes and methods in its *javadoc* HTML format. Although this documentation is somewhat difficult to navigate and is rough or outdated in places, it is still an excellent starting point when you need to know more about a particular Java package, class, method, or field. If you do not already have the *javadoc* files with your Java distribution, see <http://java.sun.com/docs/> for a link to the latest available version. Sun also distributes its excellent *Java Tutorial* online. You can browse and download it from <http://java.sun.com/docs/books/tutorial/>.

For Usenet discussion (in English) about Java, try the *comp.lang.java.programmer* and related *comp.lang.java.** newsgroups. You can find the very comprehensive

comp.lang.java.programmer FAQ by Peter van der Linden at <http://www.afu.com/javafaq.btm>.

Finally, don't forget O'Reilly's Java web site. <http://java.oreilly.com/> contains Java news and commentary. The O'Reilly Network (www.oreillynet.com) includes the *onjava.com* site which has a focus on Enterprise Java.

Examples Online

The examples in this book are available online and can be downloaded from the home page for the book at <http://www.oreilly.com/catalog/javanut4>. You also may want to visit this site to see if any important notes or errata about the book have been published there.

Conventions Used in This Book

We use the following formatting conventions in this book:

Italic

Used for emphasis and to signify the first use of a term. Italic is also used for commands, email addresses, web sites, FTP sites, file and directory names, and newsgroups.

Bold

Occasionally used to refer to particular keys on a computer keyboard or to portions of a user interface, such as the **Back** button or the **Options** menu.

Constant Width

Used in all Java code and generally for anything that you would type literally when programming, including keywords, data types, constants, method names, variables, class names, and interface names.

Constant Width Italic

Used for the names of function arguments and generally as a placeholder to indicate an item that should be replaced with an actual value in your program.

Franklin Gothic Book Condensed

Used for the Java class synopses in the quick-reference section. This very narrow font allows us to fit a lot of information on the page without a lot of distracting line breaks. This font is also used for code entities in the descriptions in the quick-reference section.

Franklin Gothic Demi Condensed

Used for highlighting class, method, field, property, and constructor names in the quick-reference section, which makes it easier to scan the class synopses.

Franklin Gothic Book Condensed Italic

Used for method parameter names and comments in the quick-reference section.

Request for Comments

Please address comments and questions concerning this book to the publisher:

O'Reilly & Associates, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
(800) 998-9938 (in the United States or Canada)
(707) 829-0515 (international or local)
(707) 829-1014 (fax)

There is a web page for this book, which lists errata, examples, and any additional information. You can access this page at:

<http://www.oreilly.com/catalog/javanut4/>

To ask technical questions or comment on this book, send email to:

bookquestions@oreilly.com

For more information about books, conferences, Resource Centers, and the O'Reilly Network, see the O'Reilly web site at:

<http://www.oreilly.com/>

How the Quick Reference Is Generated

For the curious reader, this section explains a bit about how the quick-reference material in *Java in a Nutshell* and related books is created.

As Java has evolved, so has my system for generating Java quick-reference material. The current system is part of a larger commercial documentation browser system I'm developing (visit <http://www.davidflanagan.com/jude/> for more information about it). The program works in two passes: the first pass collects and organizes the API information, and the second pass outputs that information in the form of quick-reference chapters.

The first pass begins by reading the class files for all of the classes and interfaces to be documented. Almost all of the API information in the quick reference is available in these class files. The notable exception is the names of method arguments, which are not stored in class files. These argument names are obtained by parsing the Java source file for each class and interface. Where source files are not available, I obtain method argument names by parsing the API documentation generated by *javadoc*. The parsers I use to extract API information from the source files and *javadoc* files are created using the Antlr parser generator developed by Terrence Parr of the Magelang Institute. (See <http://www.antlr.org/> for details on this very powerful programming tool.)

Once the API information has been obtained by reading class files, source files, and *javadoc* files, the program spends some time sorting and cross-referencing everything. Then it stores all the API information into a single large data file.

The second pass reads API information from that data file and outputs quick-reference chapters using a custom XML doctype. Once I've generated the XML output, I

hand it off to the production team at O'Reilly. They process it and convert it to troff source code. The troff source is processed with the GNU *groff* program ([ftp://ftp.gnu.org/gnu/groff/](http://ftp.gnu.org/gnu/groff/)) and a custom set of troff macros to produce PostScript output that is shipped directly to the printer.

Acknowledgments

Many people helped in the creation of this book, and I am grateful to them all. I am indebted to the many, many readers of the first three editions who wrote in with comments, suggestions, bug reports, and praise. Their many small contributions are scattered throughout the book. Also, my apologies to those who made the many good suggestions that could not be incorporated into this edition.

Paula Ferguson, a friend and colleague, was the editor of the first three editions of this book. Her careful reading and always-practical suggestions have made the book stronger, clearer, and more useful. Paula's editorial duties have moved her away from Java books and into Web programming books, and this fourth edition was edited by Bob Eckstein, a careful editor with a great sense of humor.

The new material I wrote for this edition has been reviewed by a number of engineers at Sun, and often these engineers were the very ones who created or worked on the APIs for which they were reviewers. I am fortunate to have been able to go "straight to the source" for these reviews, and am very grateful to these engineers, who made time in their very busy schedules to read and comment on my drafts. In alphabetical order, the reviewers were:

- Josh Bloch, author of the excellent book *Effective Java Programming Language Guide*, reviewed the new material on assertions and the Preferences API.
- Graham Hamilton reviewed the Logging API material.
- Jonathan Knudsen (who is also an O'Reilly author) reviewed the JSSE and Certification Path material.
- Charlie Lai reviewed the JAAS material.
- Ram Marti reviewed the JGSS material.
- Philip Milne, a former Sun employee, now at Dresdner Kleinwort Wasserstein, reviewed the material on the Java Beans persistence mechanism.
- Mark Reinhold reviewed the `java.nio` material. Mark deserves special thanks for having been a reviewer for the second, third, and fourth editions of this book.
- Andreas Sterbenz and Brad Wetmore reviewed the JSSE material.

In addition to these reviewers from Sun, Ron Hitchens reviewed my New I/O material, and my editor, Bob Eckstein, did double duty as the technical reviewer for the XML material. My sincere thanks to each of these gentlemen for their careful work. Any mistakes that remain in this book are, of course, my own.

The third edition also benefited greatly from the contributions of reviewers who are intimately familiar with the Java platform. Joshua Bloch, one of the primary authors of the Java collections framework, reviewed my descriptions of the collections classes and interfaces. Joshua was also helpful in discussing the `Timer` and `TimerTask` classes of Java 1.3 with me. Mark Reinhold, creator of the `java.lang.ref` package, explained the package to me and reviewed my documentation of it. Scott Oaks reviewed my descriptions of the Java security and cryptography classes and interfaces. Joshua, Mark, and Scott are all engineers with Sun Microsystems, and I'm very grateful for their time. The documentation of the `javax.crypto` package and its subpackages was also reviewed by Jon Eaves. Jon worked on a clean-room implementation of the Java Cryptography Extension (which is available from <http://www.aba.net.au/>), and his comments were quite helpful. Jon now works for Fluent Technologies (<http://www.fluent.com.au/>) consulting in Java and electronic commerce. Finally, Chapter 1 was improved by the comments of reviewers who were *not* already familiar with the Java platform: Christina Byrne reviewed it from the standpoint of a novice programmer, and Judita Byrne of Virginia Power offered her comments as a professional COBOL programmer.

For the second edition, John Zukowski reviewed my Java 1.1 AWT quick-reference material, and George Reese reviewed most of the remaining new material. The second edition was also blessed with a “dream team” of technical reviewers from Sun. John Rose, the author of the Java inner class specification, reviewed the chapter on inner classes. Mark Reinhold, author of the new character stream classes in `java.io`, reviewed my documentation of these classes. Nakul Saraiya, the designer of the new Java Reflection API, reviewed my documentation of the `java.lang.reflect` package. I am very grateful to these engineers and architects; their efforts made this a stronger, more accurate book.

Mike Loukides provided high-level direction and guidance for the first edition of the book. Eric Raymond and Troy Downing reviewed that first edition—they helped spot my errors and omissions and offered good advice on making the book more useful to Java programmers.

The O'Reilly production team has done its usual fine work of creating a book out of the electronic files I submit. My thanks to them all.

As always, my thanks and love to Christie.

David Flanagan
<http://www.davidflanagan.com/>
January 2002

