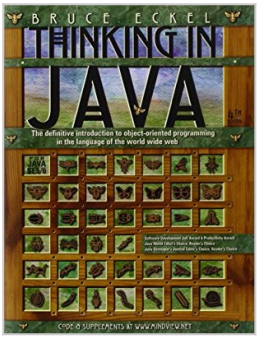

Thinking in Java (4th Edition)

By Bruce Eckel

Thinking in Java (4th Edition) By Bruce Eckel

"*Thinking in Java* should be read cover to cover by every Java programmer, then kept close at hand for frequent reference. The exercises are challenging, and the chapter on Collections is superb! Not only did this book help me to pass the Sun Certified Java Programmer exam; it's also the first book I turn to whenever I have a Java question."
—Jim Pleger, Loudoun County (Virginia) Government

"*Much* better than any other Java book I've seen. Make that 'by an order of magnitude'.... Very complete, with excellent right-to-the-point examples and intelligent, not dumbed-down, explanations.... In contrast to many other Java books I found it to be unusually mature, consistent, intellectually honest, well-written, and precise. IMHO, an ideal book for studying Java."
—Anatoly Vorobey, Technion University, Haifa, Israel

"Absolutely one of the best programming tutorials I've seen for any language."
—Joakim Ziegler, FIX sysop

"Thank you again for your awesome book. I was really floundering (being a non-C programmer), but your book has brought me up to speed as fast as I could read it. It's really cool to be able to understand the underlying principles and concepts from the start, rather than having to try to build that conceptual model through trial and error. Hopefully I will be able to attend your seminar in the not-too-distant future."
—Randall R. Hawley, automation technician, Eli Lilly & Co.

"This is one of the best books I've read about a programming language.... The best book ever written on Java."
—Ravindra Pai, Oracle Corporation, SUNOS product line

"Bruce, your book is wonderful! Your explanations are clear and

direct. Through your fantastic book I have gained a tremendous amount of Java knowledge. The exercises are also *fantastic* and do an excellent job reinforcing the ideas explained throughout the chapters. I look forward to reading more books written by you. Thank you for the tremendous service that you are providing by writing such great books. My code will be much better after reading *Thinking in Java.* I thank you and I'm sure any programmers who will have to maintain my code are also grateful to you."
—Yvonne Watkins, Java artisan, Discover Technologies, Inc.

"Other books cover the *what* of Java (describing the syntax and the libraries) or the *how* of Java (practical programming examples). *Thinking in Java* is the only book I know that explains the *why* of Java: Why it was designed the way it was, why it works the way it does, why it sometimes doesn't work, why it's better than C++, why it's not. Although it also does a good job of teaching the what and how of the language, *Thinking in Java* is definitely the thinking person's choice in a Java book."
—Robert S. Stephenson

Awards for *Thinking in Java*

2003 *Software Development Magazine* Jolt Award for Best Book
2003 *Java Developer's Journal* Reader's Choice Award for Best Book
2001 *JavaWorld* Editor's Choice Award for Best Book
2000 *JavaWorld* Reader's Choice Award for Best Book
1999 *Software Development Magazine* Productivity Award
1998 *Java Developer's Journal* Editor's Choice Award for Best Book

Thinking in Java has earned raves from programmers worldwide for its extraordinary clarity, careful organization, and small, direct programming examples. From the fundamentals of Java syntax to its most advanced features, ***Thinking in Java*** is designed to teach, one simple step at a time.

- The classic object-oriented introduction for beginners and experts alike, fully updated for Java SE5/6 with many new examples and chapters!
- Test framework shows program output.
- Design patterns are shown with multiple examples throughout: Adapter, Bridge, Chain of Responsibility, Command, Decorator, Facade, Factory Method, Flyweight, Iterator, Data Transfer Object, Null Object, Proxy, Singleton, State, Strategy, Template Method, and Visitor.
- Introduction to XML for data transfer; SWT, Flash for user interfaces.
- Completely rewritten concurrency chapter gives you a solid grasp of threading fundamentals.
- 500+ working Java programs in 700+ compiling files, rewritten for this edition and Java SE5/6.
- Companion web site includes all source code, annotated solution guide,

weblog, and multimedia seminars.

- Thorough coverage of fundamentals; demonstrates advanced topics.
- Explains sound object-oriented principles.
- *Hands-On Java Seminar CD* available online, with full multimedia seminar by Bruce Eckel.
- Live seminars, consulting, and reviews available. *See* www.MindView.net

Download seven free sample chapters from ***Thinking in Java, Fourth Edition.***
Visit http://mindview.net/Books/TIJ4.

Thinking in Java (4th Edition)

By Bruce Eckel

Thinking in Java (4th Edition) By Bruce Eckel

"*Thinking in Java* should be read cover to cover by every Java programmer, then kept close at hand for frequent reference. The exercises are challenging, and the chapter on Collections is superb! Not only did this book help me to pass the Sun Certified Java Programmer exam; it's also the first book I turn to whenever I have a Java question."
—Jim Pleger, Loudoun County (Virginia) Government

"*Much* better than any other Java book I've seen. Make that 'by an order of magnitude'.... Very complete, with excellent right-to-the-point examples and intelligent, not dumbed-down, explanations.... In contrast to many other Java books I found it to be unusually mature, consistent, intellectually honest, well-written, and precise. IMHO, an ideal book for studying Java."
—Anatoly Vorobey, Technion University, Haifa, Israel

"Absolutely one of the best programming tutorials I've seen for any language."
—Joakim Ziegler, FIX sysop

"Thank you again for your awesome book. I was really floundering (being a non-C programmer), but your book has brought me up to speed as fast as I could read it. It's really cool to be able to understand the underlying principles and concepts from the start, rather than having to try to build that conceptual model through trial and error. Hopefully I will be able to attend your seminar in the not-too-distant future."
—Randall R. Hawley, automation technician, Eli Lilly & Co.

"This is one of the best books I've read about a programming language.... The best book ever written on Java."
—Ravindra Pai, Oracle Corporation, SUNOS product line

"Bruce, your book is wonderful! Your explanations are clear and direct. Through your fantastic book I have gained a tremendous amount of Java knowledge. The exercises are also *fantastic* and do an excellent job reinforcing the ideas explained throughout the chapters. I look forward to reading more books written by you. Thank you for the tremendous service that you are providing by writing such great books. My code will be much better after reading *Thinking in Java*. I thank you and I'm sure any programmers who will have to maintain my code are also grateful to you."
—Yvonne Watkins, Java artisan, Discover Technologies, Inc.

"Other books cover the *what* of Java (describing the syntax and the libraries) or the *how* of Java (practical programming examples). *Thinking in Java* is the only book I know that explains the *why* of Java: Why it was designed the way it was, why it works the way it does, why it sometimes doesn't work, why it's better than C++, why it's not. Although it also does a good

job of teaching the what and how of the language, *Thinking in Java* is definitely the thinking person's choice in a Java book."
—*Robert S. Stephenson*

Awards for *Thinking in Java*

2003 *Software Development Magazine* Jolt Award for Best Book
2003 *Java Developer's Journal* Reader's Choice Award for Best Book
2001 *JavaWorld* Editor's Choice Award for Best Book
2000 *JavaWorld* Reader's Choice Award for Best Book
1999 *Software Development Magazine* Productivity Award
1998 *Java Developer's Journal* Editor's Choice Award for Best Book

Thinking in Java has earned raves from programmers worldwide for its extraordinary clarity, careful organization, and small, direct programming examples. From the fundamentals of Java syntax to its most advanced features, ***Thinking in Java*** is designed to teach, one simple step at a time.

- The classic object-oriented introduction for beginners and experts alike, fully updated for Java SE5/6 with many new examples and chapters!
- Test framework shows program output.
- Design patterns are shown with multiple examples throughout: Adapter, Bridge, Chain of Responsibility, Command, Decorator, Facade, Factory Method, Flyweight, Iterator, Data Transfer Object, Null Object, Proxy, Singleton, State, Strategy, Template Method, and Visitor.
- Introduction to XML for data transfer; SWT, Flash for user interfaces.
- Completely rewritten concurrency chapter gives you a solid grasp of threading fundamentals.
- 500+ working Java programs in 700+ compiling files, rewritten for this edition and Java SE5/6.
- Companion web site includes all source code, annotated solution guide, weblog, and multimedia seminars.
- Thorough coverage of fundamentals; demonstrates advanced topics.
- Explains sound object-oriented principles.
- *Hands-On Java Seminar CD* available online, with full multimedia seminar by Bruce Eckel.
- Live seminars, consulting, and reviews available. *See* www.MindView.net

Download seven free sample chapters from ***Thinking in Java, Fourth Edition.*** Visit http://mindview.net/Books/TIJ4.

Thinking in Java (4th Edition) By Bruce Eckel Bibliography

- Sales Rank: #60269 in Books
- Brand: Eckel, Bruce
- Published on: 2006-02-20
- Ingredients: Example Ingredients
- Original language: English
- Number of items: 1
- Dimensions: 9.10" h x 1.90" w x 7.00" l, 4.53 pounds
- Binding: Paperback
- 1150 pages

Editorial Review

From the Back Cover

""Thinking in Java" should be read cover to cover by every Java programmer, then kept close at hand for frequent reference. The exercises are challenging, and the chapter on Collections is superb! Not only did this book help me to pass the Sun Certified Java Programmer exam; it's also the first book I turn to whenever I have a Java question."
"--Jim Pleger, Loudoun County (Virginia) Government""Much" better than any other Java book I've seen. Make that 'by an order of magnitude'.... Very complete, with excellent right-to-the-point examples and intelligent, not dumbed-down, explanations.... In contrast to many other Java books I found it to be unusually mature, consistent, intellectually honest, well-written, and precise. IMHO, an ideal book for studying Java."
"--Anatoly Vorobey, Technion University, Haifa, Israel""Absolutely one of the best programming tutorials I've seen for any language."
"--Joakim Ziegler, FIX sysop""Thank you again for your awesome book. I was really floundering (being a non-C programmer), but your book has brought me up to speed as fast as I could read it. It's really cool to be able to understand the underlying principles and concepts from the start, rather than having to try to build that conceptual model through trial and error. Hopefully I will be able to attend your seminar in the not-too-distant future."
"--Randall R. Hawley, automation technician, Eli Lilly & Co.""This is one of the best books I've read about a programming language.... The best book ever written on Java."
"--Ravindra Pai, Oracle Corporation, SUNOS product line""Bruce, your book is wonderful! Your explanations are clear and direct. Through your fantastic book I have gained a tremendous amount of Java knowledge. The exercises are also "fantastic" and do an excellent job reinforcing the ideas explained throughout the chapters. I look forward to reading more books written by you. Thank you for the tremendous service that you are providing by writing such great books. My code will be much better after reading "Thinking in Java." I thank you and I'm sure any programmers who will have to maintain my code are also grateful to you."
"--Yvonne Watkins, Java artisan, Discover Technologies, Inc.""Other books cover the "what" of Java (describing the syntax and the libraries) or the "how" of Java (practical programming examples). "Thinking in Java" is the only book I know that explains the "why" of Java: Why it was designed the way it was, why it works the way it does, why it sometimes doesn't work, why it's better than C++, why it's not. Although it also does a good job of teaching the what and how of the language, "Thinking in Java" is definitely the thinking person's choice in a Java book."
"--Robert S. Stephenson"Awards for "Thinking in Java"2003 "Software Development Magazine" Jolt Award for Best Book
2003 "Java Developer's Journal" Reader's Choice Award for Best Book
2001 "JavaWorld" Editor's Choice Award for Best Book
2000 "JavaWorld" Reader's Choice Award for Best Book
1999 "Software Development Magazine" Productivity Award
1998 "Java Developer's Journal" Editor's Choice Award for Best Book

"Thinking in Java" has earned raves from programmers worldwide for its extraordinary clarity, careful organization, and small, direct programming examples. From the fundamentals of Java syntax to its most advanced features, "Thinking in Java" is designed to teach, one simple step at a time.The classic object-oriented introduction for beginners and experts alike, fully updated for Java SE5/6 with many new examples and chapters!Test framework shows program output.Design patterns are shown with multiple examples

throughout: Adapter, Bridge, Chain of Responsibility, Command, Decorator, Facade, Factory Method, Flyweight, Iterator, Data Transfer Object, Null Object, Proxy, Singleton, State, Strategy, Template Method, and Visitor.Introduction to XML for data transfer; SWT, Flash for user interfaces.Completely rewritten concurrency chapter gives you a solid grasp of threading fundamentals.500+ working Java programs in 700+ compiling files, rewritten for this edition and Java SE5/6.Companion web site includes all source code, annotated solution guide, weblog, and multimedia seminars.Thorough coverage of fundamentals; demonstrates advanced topics.Explains sound object-oriented principles."Hands-On Java Seminar CD" available online, with full multimedia seminar by Bruce Eckel.Live seminars, consulting, and reviews available. "See" www.MindView.net

Download seven free sample chapters from "Thinking in Java, Fourth Edition." Visit http: //mindview.net/Books/TIJ4.

About the Author

Bruce Eckel is president of MindView, Inc. (www.MindView.net), which provides public and private training seminars, consulting, mentoring, and design reviews in object-oriented technology and design patterns. He is the author of several books, has written more than fifty articles, and has given lectures and seminars throughout the world for more than twenty years. Bruce has served as a voting member of the C++ Standards Committee. He holds a B.S. in applied physics and an M.S. in computer engineering.

I originally approached Java as “just another programming language,” which in many senses it is.

But as time passed and I studied it more deeply, I began to see that the fundamental intent of this language was different from other languages I had seen up to that point.

Programming is about managing complexity: the complexity of the problem you want to solve, laid upon the complexity of the machine in which it is solved. Because of this complexity, most of our programming projects fail. And yet, of all the programming languages of which I am aware, almost none have gone all out and decided that their *main* design goal would be to conquer the complexity of developing and maintaining programs.[1] Of course, many language design decisions were made with complexity in mind, but at some point there were always other issues that were considered essential to be added into the mix. Inevitably, those other issues are what cause programmers to eventually “hit the wall” with that language. For example, C++ had to be backwards-compatible with C (to allow easy migration for C programmers), as well as efficient. Those are both very useful goals and account for much of the success of C++, but they also expose extra complexity that prevents some projects from being finished (certainly, you can blame programmers and management, but if a language can help by catching your mistakes, why shouldn’t it?). As another example, Visual BASIC (VB) was tied to BASIC, which wasn’t really designed to be an extensible language, so all the extensions piled upon VB have produced some truly unmaintainable syntax. Perl is backwards-compatible with awk, sed, grep, and other Unix tools it was meant to replace, and as a result it is often accused of producing “write-only code” (that is, after a while you can’t read it). On the other hand, C++, VB, Perl, and other languages like Smalltalk had *some* of their design efforts focused on the issue of complexity and as a result are remarkably successful in solving certain types of problems.

What has impressed me most as I have come to understand Java is that somewhere in the mix of Sun’s design objectives, it seems that there was a goal of reducing complexity *for the programmer*. As if to say,

“We care about reducing the time and difficulty of producing robust code.” In the early days, this goal resulted in code that didn’t run very fast (although this has improved over time), but it has indeed produced amazing reductions in development time—half or less of the time that it takes to create an equivalent C++ program. This result alone can save incredible amounts of time and money, but Java doesn’t stop there. It goes on to wrap many of the complex tasks that have become important, such as multithreading and network programming, in language features or libraries that can at times make those tasks easy. And finally, it tackles some really big complexity problems: cross-platform programs, dynamic code changes, and even security, each of which can fit on your complexity spectrum anywhere from “impediment” to “show-stopper.” So despite the performance problems that we’ve seen, the promise of Java is tremendous: It can make us significantly more productive programmers.

In all ways—creating the programs, working in teams, building user interfaces to communicate with the user, running the programs on different types of machines, and easily writing programs that communicate across the Internet—Java increases the communication bandwidth *between people.*

I think that the results of the communication revolution may not be seen from the effects of moving large quantities of bits around. We shall see the true revolution because we will all communicate with each other more easily: one-on-one, but also in groups and as a planet. I’ve heard it suggested that the next revolution is the formation of a kind of global mind that results from enough people and enough interconnectedness. Java may or may not be the tool that foments that revolution, but at least the possibility has made me feel like I’m doing something meaningful by attempting to teach the language.

Java SE5 and SE6

This edition of the book benefits greatly from the improvements made to the Java language in what Sun originally called JDK 1.5, and then later changed to JDK5 or J2SE5, then finally they dropped the outdated “2” and changed it to Java SE5. Many of the Java SE5 language changes were designed to improve the experience of the programmer. As you shall see, the Java language designers did not completely succeed at this task, but in general they made large steps in the right direction.

One of the important goals of this edition is to completely absorb the improvements of Java SE5/6, and to introduce and use them throughout this book. This means that this edition takes the somewhat bold step of being “Java SE5/6-only,” and much of the code in the book will not compile with earlier versions of Java; the build system will complain and stop if you try. However, I think the benefits are worth the risk.

If you are somehow fettered to earlier versions of Java, I have covered the bases by providing free downloads of previous editions of this book via www.MindView.net. For various reasons, I have decided not to provide the current edition of the book in free electronic form, but only the prior editions.

Java SE6

This book was a monumental, time-consuming project, and before it was published, Java SE6 (code-named *mustang*) appeared in beta form. Although there were a few minor changes in Java SE6 that improved some of the examples in the book, for the most part the focus of Java SE6 did not affect the content of this book; the features were primarily speed improvements and library features that were outside the purview of this text.

The code in this book was successfully tested with a release candidate of Java SE6, so I do not expect any changes that will affect the content of this book. If there are any important changes by the time Java SE6 is officially released, these will be reflected in the book’s source code, which is downloadable from

www.MindView.net.

The cover indicates that this book is for “Java SE5/6,” which means “written for Java SE5 and the very significant changes that version introduced into the language, but is equally applicable to Java SE6.”

The 4th edition

The satisfaction of doing a new edition of a book is in getting things “right,” according to what I have learned since the last edition came out. Often these insights are in the nature of the saying “A learning experience is what you get when you don’t get what you want,” and my opportunity is to fix something embarrassing or simply tedious. Just as often, creating the next edition produces fascinating new ideas, and the embarrassment is far outweighed by the delight of discovery and the ability to express ideas in a better form than what I have previously achieved.

There is also the challenge that whispers in the back of my brain, that of making the book something that owners of previous editions will want to buy. This presses me to improve, rewrite and reorganize everything that I can, to make the book a new and valuable experience for dedicated readers.

Changes

The CD-ROM that has traditionally been packaged as part of this book is not part of this edition. The essential part of that CD, the *Thinking in C* multimedia seminar (created for MindView by Chuck Allison), is now available as a downloadable Flash presentation. The goal of that seminar is to prepare those who are not familiar enough with C syntax to understand the material presented in this book. Although two of the chapters in this book give decent introductory syntax coverage, they may not be enough for people without an adequate background, and *Thinking in C* is intended to help those people get to the necessary level.

The *Concurrency* chapter (formerly called “Multithreading”) has been completely rewritten to match the major changes in the Java SE5 concurrency libraries, but it still gives you a basic foundation in the core ideas of concurrency. Without that core, it’s hard to understand more complex issues of threading. I spent many months working on this, immersed in that netherworld called “concurrency,” and in the end the chapter is something that not only provides a basic foundation but also ventures into more advanced territory.

There is a new chapter on every significant new Java SE5 language feature, and the other new features have been woven into modifications made to the existing material. Because of my continuing study of design patterns, more patterns have been introduced throughout the book as well.

The book has undergone significant reorganization. Much of this has come from the teaching process together with a realization that, perhaps, my perception of what a “chapter” was could stand some rethought. I have tended towards an unconsidered belief that a topic had to be “big enough” to justify being a chapter. But especially while teaching design patterns, I find that seminar attendees do best if I introduce a single pattern and then we immediately do an exercise, even if it means I only speak for a brief time (I discovered that this pace was also more enjoyable for me as a teacher). So in this version of the book I’ve tried to break chapters up by topic, and not worry about the resulting length of the chapters. I think it has been an improvement.

I have also come to realize the importance of code testing. Without a built-in test framework with tests that are run every time you do a build of your system, you have no way of knowing if your code is reliable or not. To accomplish this in the book, I created a test framework to display and validate the output of each program. (The framework was written in Python; you can find it in the downloadable code for this book at www.MindView.net.) Testing in general is covered in the supplement you will find at

http://MindView.net/Books/BetterJava, which introduces what I now believe are fundamental skills that all programmers should have in their basic toolkit.

In addition, I've gone over every single example in the book and asked myself, "Why did I do it this way?" In most cases I have done some modification and improvement, both to make the examples more consistent within themselves and also to demonstrate what I consider to be best practices in Java coding (at least, within the limitations of an introductory text). Many of the existing examples have had very significant redesign and reimplementation. Examples that no longer made sense to me were removed, and new examples have been added.

Readers have made many, many wonderful comments about the first three editions of this book, which has naturally been very pleasant for me. However, every now and then, someone will have complaints, and for some reason one complaint that comes up periodically is "The book is too big." In my mind it is faint damnation indeed if "too many pages" is your only gripe. (One is reminded of the Emperor of Austria's complaint about Mozart's work: "Too many notes!" Not that I am in any way trying to compare myself to Mozart.) In addition, I can only assume that such a complaint comes from someone who is yet to be acquainted with the vastness of the Java language itself and has not seen the rest of the books on the subject. Despite this, one of the things I have attempted to do in this edition is trim out the portions that have become obsolete, or at least nonessential. In general, I've tried to go over everything, remove what is no longer necessary, include changes, and improve everything I could. I feel comfortable removing portions because the original material remains on the Web site (www.MindView.net), in the form of the freely downloadable 1st through 3rd editions of the book, and in the downloadable supplements for this book. For those of you who still can't stand the size of the book, I do apologize. Believe it or not, I have worked hard to keep the size down.

Note on the cover design

The cover of *Thinking in Java* is inspired by the American Arts & Crafts Movement that began near the turn of the century and reached its zenith between 1900 and 1920. It began in England as a reaction to both the machine production of the Industrial Revolution and the highly ornamental style of the Victorian era. Arts & Crafts emphasized spare design, the forms of nature as seen in the art nouveau movement, hand-crafting, and the importance of the individual craftsperson, and yet it did not eschew the use of modern tools. There are many echoes with the situation we have today: the turn of the century, the evolution from the raw beginnings of the computer revolution to something more refined and meaningful, and the emphasis on software craftsmanship rather than just manufacturing code.

I see Java in this same way: as an attempt to elevate the programmer away from an operating system mechanic and toward being a "software craftsman."

Both the author and the book/cover designer (who have been friends since childhood) find inspiration in this movement, and both own furniture, lamps, and other pieces that are either original or inspired by this period.

The other theme in this cover suggests a collection box that a naturalist might use to display the insect specimens that he or she has preserved. These insects are objects that are placed within the box objects. The box objects are themselves placed within the "cover object," which illustrates the fundamental concept of aggregation in object-oriented programming. Of course, a programmer cannot help but make the association with "bugs," and here the bugs have been captured and presumably killed in a specimen jar, and finally confined within a small display box, as if to imply Java's ability to find, display, and subdue bugs (which is truly one of its most powerful attributes).

In this edition, I created the watercolor painting that you see as the cover background.

Acknowledgements

First, thanks to associates who have worked with me to give seminars, provide consulting, and develop teaching projects: Dave Bartlett, Bill Venners, Chuck Allison, Jeremy Meyer, and Jamie King. I appreciate your patience as I continue to try to develop the best model for independent folks like us to work together.

Recently, no doubt because of the Internet, I have become associated with a surprisingly large number of people who assist me in my endeavors, usually working from their own home offices. In the past, I would have had to pay for a pretty big office space to accommodate all these folks, but because of the Net, FedEx, and the telephone, I'm able to benefit from their help without the extra costs. In my attempts to learn to "play well with others," you have all been very helpful, and I hope to continue learning how to make my own work better through the efforts of others. Paula Steuer has been invaluable in taking over my haphazard business practices and making them sane (thanks for prodding me when I don't want to do something, Paula). Jonathan Wilcox, Esq., has sifted through my corporate structure and turned over every possible rock that might hide scorpions, and frog-marched us through the process of putting everything straight, legally. Thanks for your care and persistence. Sharlynn Cobaugh has made herself an expert in sound processing and an essential part of creating the multimedia training experiences, as well as tackling other problems. Thanks for your perseverance when faced with intractable computer problems. The folks at Amaio in Prague have helped me out with several projects. Daniel Will-Harris was the original work-by-Internet inspiration, and he is of course fundamental to all my graphic design solutions.

Over the years, through his conferences and workshops, Gerald Weinberg has become my unofficial coach and mentor, for which I thank him.

Ervin Varga was exceptionally helpful with technical corrections on the 4th edition—although other people helped on various chapters and examples, Ervin was my primary technical reviewer for the book, and he also took on the task of rewriting the solution guide for the 4th edition. Ervin found errors and made improvements to the book that were invaluable additions to this text. His thoroughness and attention to detail are amazing, and he's far and away the best technical reader I've ever had. Thanks, Ervin.

My weblog on Bill Venners' www.Artima.com has been a source of assistance when I've needed to bounce ideas around. Thanks to the readers that have helped me clarify concepts by submitting comments, including James Watson, Howard Lovatt, Michael Barker, and others, in particular those who helped with generics.

Thanks to Mark Welsh for his continuing assistance.

Evan Cofsky continues to be very supportive by knowing off the top of his head all the arcane details of setting up and maintaining Linux-based Web servers, and keeping the MindView server tuned and secure.

A special thanks to my new friend, coffee, who generated nearly boundless enthusiasm for this project. Camp4 Coffee in Crested Butte, Colorado, has become the standard hangout when people have come up to take MindView seminars, and during seminar breaks it is the best catering I've ever had. Thanks to my buddy Al Smith for creating it and making it such a great place, and for being such an interesting and entertaining part of the Crested Butte experience. And to all the Camp4 barristas who so cheerfully dole out beverages.

Thanks to the folks at Prentice Hall for continuing to give me what I want, putting up with all my special requirements, and for going out of their way to make things run smoothly for me.

Certain tools have proved invaluable during my development process and I am very grateful to the creators every time I use these. Cygwin (www.cygwin.com) has solved innumerable problems for me that Windows can't/won't and I become more attached to it each day (if I only had this 15 years ago when my brain was still hard-wired with Gnu Emacs). IBM's Eclipse (www.eclipse.org) is a truly wonderful contribution to the development community, and I expect to see great things from it as it continues to evolve (how did IBM become hip? I must have missed a memo). JetBrains IntelliJ Idea continues to forge creative new paths in development tools.

I began using Enterprise Architect from Sparxsystems on this book, and it has rapidly become my UML tool of choice. Marco Hunsicker's Jalopy code formatter (www.triemax.com) came in handy on numerous occasions, and Marco was very helpful in configuring it to my particular needs. I've also found Slava Pestov's JEdit and plug-ins to be helpful at times (www.jedit.org) and it's quite a reasonable beginner's editor for seminars.

And of course, if I don't say it enough everywhere else, I use Python (www.Python.org) constantly to solve problems, the brainchild of my buddy Guido Van Rossum and the gang of goofy geniuses with whom I spent a few great days sprinting (Tim Peters, I've now framed that mouse you borrowed, officially named the "TimBotMouse"). You guys need to find healthier places to eat lunch. (Also, thanks to the entire Python community, an amazing bunch of people.)

Lots of people sent in corrections and I am indebted to them all, but particular thanks go to (for the 1st edition): Kevin Raulerson (found tons of great bugs), Bob Resendes (simply incredible), John Pinto, Joe Dante, Joe Sharp (all three were fabulous), David Combs (many grammar and clarification corrections), Dr. Robert Stephenson, John Cook, Franklin Chen, Zev Griner, David Karr, Leander A. Stroschein, Steve Clark, Charles A. Lee, Austin Maher, Dennis P. Roth, Roque Oliveira, Douglas Dunn, Dejan Ristic, Neil Galarneau, David B. Malkovsky, Steve Wilkinson, and a host of others. Prof. Ir. Marc Meurrens put in a great deal of effort to publicize and make the electronic version of the 1st edition of the book available in Europe.

Thanks to those who helped me rewrite the examples to use the Swing library (for the 2nd edition), and for other assistance: Jon Shvarts, Thomas Kirsch, Rahim Adatia, Rajesh Jain, Ravi Manthena, Banu Rajamani, Jens Brandt, Nitin Shivaram, Malcolm Davis, and everyone who expressed support.

In the 4th edition, Chris Grindstaff was very helpful during the development of the SWT section, and Sean Neville wrote the first draft of the Flex section for me.

Kraig Brockschmidt and Gen Kiyooka have been some of the smart technical people in my life who have become friends and have also been both influential and unusual in that they do yoga and practice other forms of spiritual enhancement, which I find quite inspirational and instructional.

It's not that much of a surprise to me that understanding Delphi helped me understand Java, since there are many concepts and language design decisions in common. My Delphi friends provided assistance by helping me gain insight into that marvelous programming environment. They are Marco Cantu (another Italian—perhaps being steeped in Latin gives one aptitude for programming languages?), Neil Rubenking (who used to do the yoga/vegetarian/Zen thing until he discovered computers), and of course Zack Urlocker (the original Delphi product manager), a long-time pal whom I've traveled the world with. We're all indebted to the brilliance of Anders Hejlsberg, who continues to toil away at C# (which, as you'll learn in this book, was a major inspiration for Java SE5).

My friend Richard Hale Shaw's insights and support have been very helpful (and Kim's, too). Richard and I spent many months giving seminars together and trying to work out the perfect learning experience for the

attendees.

The book design, cover design, and cover photo were created by my friend Daniel Will-Harris, noted author and designer (www.Will-Harris.com), who used to play with rub-on letters in junior high school while he awaited the invention of computers and desktop publishing, and complained of me mumbling over my algebra problems. However, I produced the camera-ready pages myself, so the typesetting errors are mine. Microsoft® Word XP for Windows was used to write the book and to create camera-ready pages in Adobe Acrobat; the book was created directly from the Acrobat PDF files. As a tribute to the electronic age, I happened to be overseas when I produced the final versions of the 1st and 2nd editions of the book—the 1st edition was sent from Cape Town, South Africa, and the 2nd edition was posted from Prague. The 3rd and 4th came from Crested Butte, Colorado. The body typeface is *Georgia* and the headlines are in *Verdana*. The cover typeface is *ITC Rennie Mackintosh*.

A special thanks to all my teachers and all my students (who are my teachers as well).

Molly the cat often sat in my lap while I worked on this edition, and thus offered her own kind of warm, furry support.

The supporting cast of friends includes, but is not limited to: Patty Gast (Masseuse extraordinaire), Andrew Binstock, Steve Sinofsky, JD Hildebrandt, Tom Keffer, Brian McElhinney, Brinkley Barr, Bill Gates at *Midnight Engineering Magazine*, Larry Constantine and Lucy Lockwood, Gene Wang, Dave Mayer, David Intersimone, Chris and Laura Strand, the Almquists, Brad Jerbic, Marilyn Cvitanic, Mark Mabry, the Robbins families, the Moelter families (and the McMillans), Michael Wilk, Dave Stoner, the Cranstons, Larry Fogg, Mike Sequeira, Gary Entsminger, Kevin and Sonda Donovan, Joe Lordi, Dave and Brenda Bartlett, Patti Gast, Blake, Annette & Jade, the Rentschlers, the Sudeks, Dick, Patty, and Lee Eckel, Lynn and Todd, and their families. And of course, Mom and Dad.

Note

1. However, I believe that the Python language comes closest to doing exactly that. See www.Python.org.

Users Review

From reader reviews:

Steven Slaughter:

The book Thinking in Java (4th Edition) give you a sense of feeling enjoy for your spare time. You need to use to make your capable considerably more increase. Book can to become your best friend when you getting anxiety or having big problem together with your subject. If you can make examining a book Thinking in Java (4th Edition) to get your habit, you can get much more advantages, like add your own capable, increase your knowledge about several or all subjects. You may know everything if you like open and read a book Thinking in Java (4th Edition). Kinds of book are several. It means that, science book or encyclopedia or others. So , how do you think about this publication?

John Oliver:

The e-book untitled Thinking in Java (4th Edition) is the book that recommended to you to study. You can see the quality of the publication content that will be shown to you. The language that publisher use to explained their ideas are easily to understand. The article author was did a lot of study when write the book, and so the information that they share for you is absolutely accurate. You also could get the e-book of Thinking in Java (4th Edition) from the publisher to make you more enjoy free time.

Ramona Wegener:

Thinking in Java (4th Edition) can be one of your nice books that are good idea. Most of us recommend that straight away because this reserve has good vocabulary that will increase your knowledge in language, easy to understand, bit entertaining however delivering the information. The article author giving his/her effort that will put every word into enjoyment arrangement in writing Thinking in Java (4th Edition) but doesn't forget the main level, giving the reader the hottest along with based confirm resource data that maybe you can be among it. This great information can drawn you into brand-new stage of crucial considering.

James Martin:

Do you like reading a book? Confuse to looking for your selected book? Or your book seemed to be rare? Why so many concern for the book? But any people feel that they enjoy intended for reading. Some people likes examining, not only science book and also novel and Thinking in Java (4th Edition) or perhaps others sources were given know-how for you. After you know how the truly great a book, you feel wish to read more and more. Science book was created for teacher or students especially. Those publications are helping them to include their knowledge. In additional case, beside science book, any other book likes Thinking in Java (4th Edition) to make your spare time much more colorful. Many types of book like this.

Read Thinking in Java (4th Edition) By Bruce Eckel for online ebook

Thinking in Java (4th Edition) By Bruce Eckel Free PDF d0wnl0ad, audio books, books to read, good books to read, cheap books, good books, online books, books online, book reviews epub, read books online, books to read online, online library, greatbooks to read, PDF best books to read, top books to read Thinking in Java (4th Edition) By Bruce Eckel books to read online.

Online Thinking in Java (4th Edition) By Bruce Eckel ebook PDF download

Thinking in Java (4th Edition) By Bruce Eckel Doc

Thinking in Java (4th Edition) By Bruce Eckel Mobipocket

Thinking in Java (4th Edition) By Bruce Eckel EPub