Book review: Core Java, Volumes I and II, Twelfth Edition

Andrew Binstock | April 7, 2022 | 5 minute read

This is the definitive reference and instructional work for Java and the Java ecosystem.

When Java was first released in 1996, there soon followed a long list of introductory books vying with each other to explain the new language, the new runtime, and the libraries. Most of those books vanished in the next few years, displaced by the high quality of Core Java—originally a single volume coauthored by Cay Horstmann, who is both a Java Champion and a professor of computer science at San Jose State University, and by Gary Cornell.

Since those early days, Horstmann has become the sole author, and Core Java has evolved through twelve editions and expanded into two volumes weighing in at
This review covers the twelfth edition in the series, with each edition coinciding roughly to a major release of Java. (I'll discuss the effect of Java's long-term support schedule later in the review.) Volume I is available now and volume II will appear shortly; I based this review of volume II on a final pre-press version of the book. These two volumes are up to date through the current long-term support (LTS) platform release, Java 17.

Core Java is written for experienced developers, that is, programmers who already know how to write code in some language and who want practical information without the need to be taught basic concepts or entertained by code samples of little value.

With the intelligence and experience of his audience in mind, Horstmann explores the Java language in great depth. Due to his long association with Java, he provides insight as well as historical context that other authors cannot match.

For example, the following comment came from volume II, section 2.1.5:

In Java 1.0, the PrintStream class simply truncated all Unicode characters to ASCII characters by dropping the top byte. (At the time, Unicode was still a 16-bit encoding.) Clearly, that was not a clean or portable approach, and it was fixed with the introduction of readers and writers in Java 1.1. For compatibility with existing code, System.in, System.out, and System.err are still input/output streams, not readers and writers. But now the PrintStream class internally converts Unicode characters to the default host encoding...

Explanations are always detailed and in depth, and they are frequently supported with clear, brief code that uncannily foresees what a reader using the books as reference volumes would be looking for. Sprinkled throughout these explanations are tips, cautionary advice, and occasional comparisons of Java features to counterparts in C++. Even for seasoned Java developers, these added items provide unexpected insights, similar to the kind revealed by the quizzes in Java Magazine.

Permeating the expositions is Horstmann's own point of view, which offers a light commentary on the utility of various features, as well as recommendations for alternative solutions within the language.

For example, in volume I, section 4.8.5, Horstmann comments on the default visibility of classes versus variables in those classes.

For classes, [public access] is a reasonable default. However, for variables, this was an unfortunate choice. Variables must explicitly be marked private, or they will default to having package access. This, of course, breaks encapsulation. The problem is that it is awfully easy to forget to type the private keyword. Here is an example from the Window class in the java.awt package...

By using an example from a standard Java library, Horstmann cements the understanding of the various access levels and how to use them correctly.

In addition, Horstmann includes small caveats and warnings of problems that might affect readers coming...
I recently reviewed Horstmann’s books, Core Java, volumes 1 and 2. Many programmers who might be thinking of going to Java for the first time or from earlier editions of the JDK. For example, his discussion of the new HTTP client includes a helpful note for users of Java 9 and Java 10 on the needed command-line option to access the feature. I doubt many users are on those two versions, but his thoroughness accommodating them is pervasive throughout these two books.

The writing is consistently clear. For many topics, I find Horstmann’s explanations clearer and more useful than any other source, and rarely, if ever, do I come to the end of a topic and think to myself “Yeah, but what about X?” To sample Horstmann’s writing in these books, consult several extracts in the form of articles that have appeared in this magazine.

**Core Java is big**

The voluminous page count might deter someone from recommending these two volumes as resources for learning Java—but should not. A student new to the language would surely be able to map a path through the content that would reduce the reading load. For example, there are sections on Swing, running a JavaScript scripting engine, writing a classloader, calling C functions, using the compiler API, and parsing XML, which are unlikely topics for a neophyte to explore. The benefit of these sections, though, is that these volumes make a definitive reference work.

A good practice when you have a question about the language or the principal libraries is to check Horstmann rather than Stack Overflow or the other online forums. In this way, you’ll get the correct answer on the first hit, along with full context and any associated caveats and reminders.

I strongly advise serious Java developers to purchase both volumes and to update them with each new release, which—Horstmann told me in private correspondence—will appear after every LTS release from Oracle. Until recently, that meant a three-year cadence, but it’s now a two-year cycle.

Despite my strong recommendation of these two volumes, I do experience one key frustration: the choice of topics to include in each volume. For example, volume I, which covers much of the syntax of the language, lacks a section on streams, which are consigned to volume II. However, volume I does include 150 pages on user interface programming, primarily using Swing. I/O is likewise found in volume II. The result of this peculiar division of content is that even though volume II is labeled “Advanced Features,” you need both volumes to cover the basics. This is fine for a reference work, but suboptimal for a student of the language.

That one issue aside, Core Java is the definitive reference and instructional work on Java and the Java ecosystem. Don’t even fool with anything else—you’d be wasting your time and your money.

**Dig deeper**

- 12 handy debugging tips from Cay Horstmann’s *Core Java*
- *Synchronization in Java, Part 1: Race conditions, locks, and conditions*
- *Synchronization in Java, Part 2: The synchronized keyword*
- *Synchronization in Java, Part 3: Atomic operations and deadlocks*
Andrew Binstock

Andrew Binstock (@platypusguy) was formerly the editor in chief of Java Magazine. Previously, he was the editor of Dr. Dobb's Journal. He co-founded the company behind the open-source iText PDF library, which was acquired in 2015. His book on algorithm implementation in C went through 16 printings before joining the long tail. Previously, he was the editor in chief of UNIX Review and, earlier, the founding editor of the C Gazette. He lives in Silicon Valley with his wife. When not coding or editing, he studies piano.