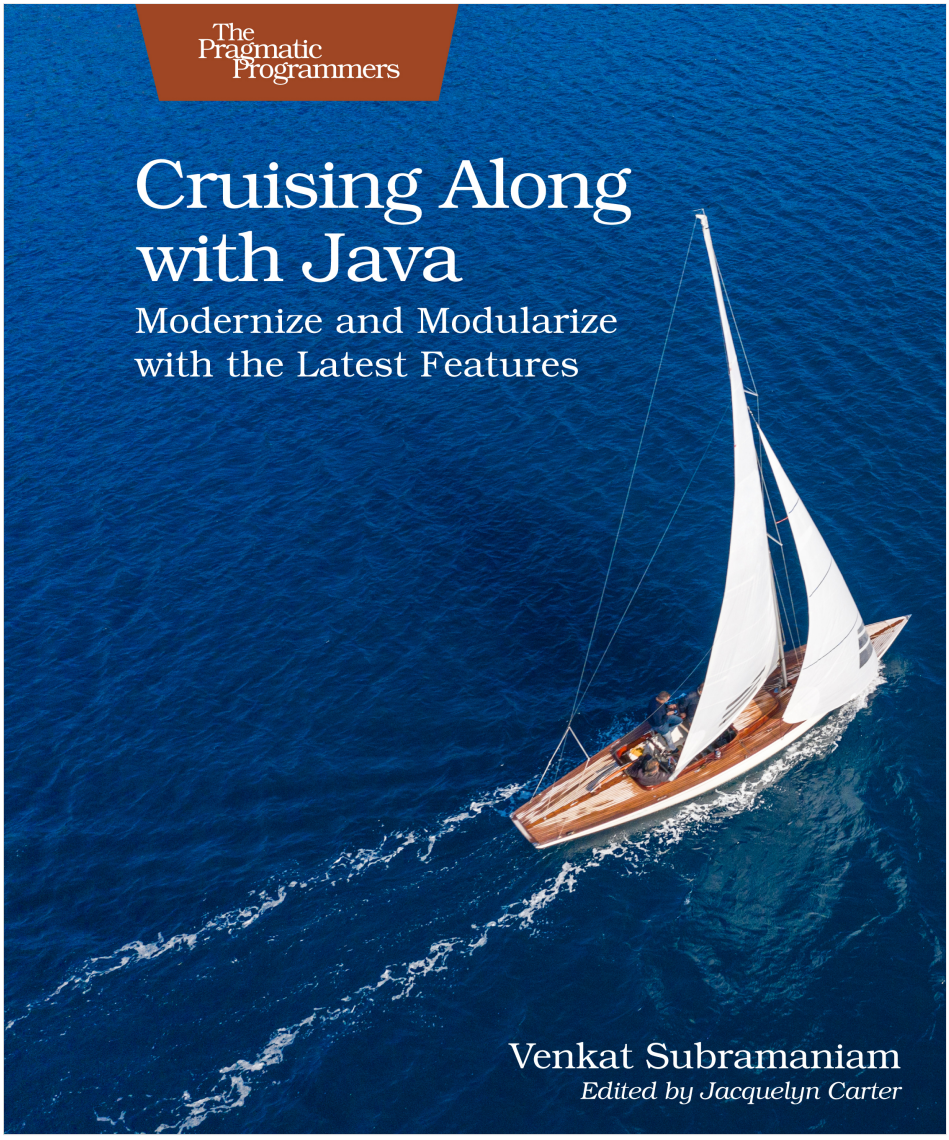

This extract shows the online version of this title, and may contain features (such as hyperlinks and colors) that are not available in the print version.

For more information, or to purchase a paperback or ebook copy, please visit https://www.pragprog.com.

Copyright © The Pragmatic Programmers, LLC.

Preface

Do you have a new-found love for Java? If so, you're not alone. I once complained that Java was stagnant and its days were over. The team behind Java proved the naysayers like me wrong in the most brilliant way—by making Java a highly vibrant language.

Truly, the first time I did a double take at Java was when the language introduced the functional programming ability in version 8. I even wrote a book about it: *Functional Programming in Java, Second Edition [Sub23]*. Every release since then has only gotten better, more interesting, and more exciting. And those who know me also know that I can't keep my excitement quiet. The result—the book you're reading.

A number of my clients were eager to get trained on modularizing Java. They were keen to learn about the developments in the language, how to make good use of records, the concept of sealed classes, the benefits of pattern matching, and so on. Ongoing discussions and the continuous demand for such content prompted me to invest my time and effort to write in detail about the amazing capabilities of Java from version 9 onward.

Thank you for reading this book. Get ready to dive deep into the features that were recently added to the Java language.

What's in This Book?

In Chapter 1, The Evolution of Java, on page ?, we start with a quick introduction. Then, we'll group the changes in Java into these categories:

- *Syntax Sugar*: Some of the features can be classified as syntax sugar; type Inference and text blocks make us productive but have no footprint in the bytecode. These are purely compiler-level features and don't permeate into the JVM ecosystem. These are covered in Chapter 2, Using Type Inference, on page ?, and Chapter 3, Reducing Clutter with Text Blocks, on page ?.

• Click HERE to purchase this book now. discuss

- *Design Aid*: Features such as records and sealed classes/interfaces help us with designing better object-oriented code. We'll see in Chapter 4, Programming with Records, on page ?, how records can help to better model data and in Chapter 5, Designing with Sealed Classes and Interfaces, on page ?, how sealed classes can be used to better model and manage inheritance hierarchies.
- *Fluent Expressions*: No one wants to write verbose code, and no one ever enjoys maintaining them. Java has upped a notch in fluency, and we can write highly expressive code that's less error-prone using switch as an expression—see Chapter 6, Switching to Switch Expression, on page ?. We can take it further and benefit from Pattern Matching, as you'll see in Chapter 7, Using Powerful Pattern Matching, on page ?.
- *Modularization*: The JDK has been finally split into manageable pieces and we can benefit from the same techniques used in Java to modularize our own applications. In Chapter 8, Modularizing Your Java Applications, on page ?, we'll discuss the need to modularize and the steps to take. In Chapter 9, Working with Modules, on page ?, we'll look at the practical considerations of working with multiple modules. Then, in Chapter 10, Creating Plug-ins with ServiceLoader, on page ?, we'll see how to use the powerful ServiceLoader to dynamically discover implementations when creating plug-ins.
- *Custom Functional Pipeline Steps*: The functional programming capability of Java has a significant enhancement with the gatherers facility. In Chapter 11, Extending Functional Pipelines with Gatherers, on page ?, we'll take a look at the intent of gatherers and how to make use of the built-in gatherers in the JDK. In Chapter 12, Creating Custom Gatherers, on page ?, you'll learn how to create your own custom steps in a functional pipeline using the Gatherer interface.

Who's This Book For?

This book is for you if you develop applications with Java and want to stay abreast of changes in the language. This book assumes you're familiar with programming in general and with both object-oriented programming (OOP) and functional programming concepts possible in Java since version 8.

You'll benefit the most from this book if you're a programmer using the Java language on a daily basis, a team lead, a hands-on architect, or a technical manager. As a senior developer or an architect, this book will help you consider and decide features that may be the most useful for the applications you're in charge of designing.

• Click HERE to purchase this book now. discuss

In addition to learning the concepts you can directly use for your enterprise applications, you can also use this book to train your team members with the latest features of the Java language.

Java Version Used in This Book

The different features you'll learn in this book were introduced over time in different versions of Java from Java 9 onward. To be able to execute all the code in this book, you'll need at least Java 24.

Take a few minutes to download the appropriate build of the JDK for your machine and operating system. This will help you follow along with the examples in this book.

How to Read the Code Examples

When writing code in Java, we place classes in packages and executable statements and expressions in methods. To reduce clutter, we'll skip the package names and imports in the code listings. All code in this book, except where explicitly stated, has been placed in a package using the statement:

```
package vsca;
```

In case you're wondering, the package name vsca is based on the code name we use for this book's repository and isn't related to any tool, product, or company.

Any executable code not listed within a method is part of an undisplayed main() method. When going through the code listings, if you have an urge to look at the full source code, remember it's only a click away at the website for this book.

Online Resources

You can download all the source code examples from this book's page[1] at the Pragmatic Bookshelf website. You can also provide feedback there by submitting errata entries or posting your comments and questions in the forum. If you're reading the book in PDF form, you can click on the link above a code listing to view or download the specific examples.

Now let's dive into the exciting features of the recent versions of Java.

Venkat Subramaniam

April 2025

1. http://www.pragprog.com/titles/vscajava

Part I

Syntax Sugar

A few features in Java make the programmers productive but don't have a direct bytecode representation. They don't impact runtime performance in any way. They're purely Java language facilities and, unlike most other features, aren't available at the JVM level for other languages to use or interoperate with.

In this part we'll look at how we can benefit from type inference. Then we'll see how text blocks reduce so much verbosity from code.