Extracted from:

Functional Programming in Java, Second Edition

Harness the Power of Streams and Lambda Expressions

The Pragmatic Bookshelf

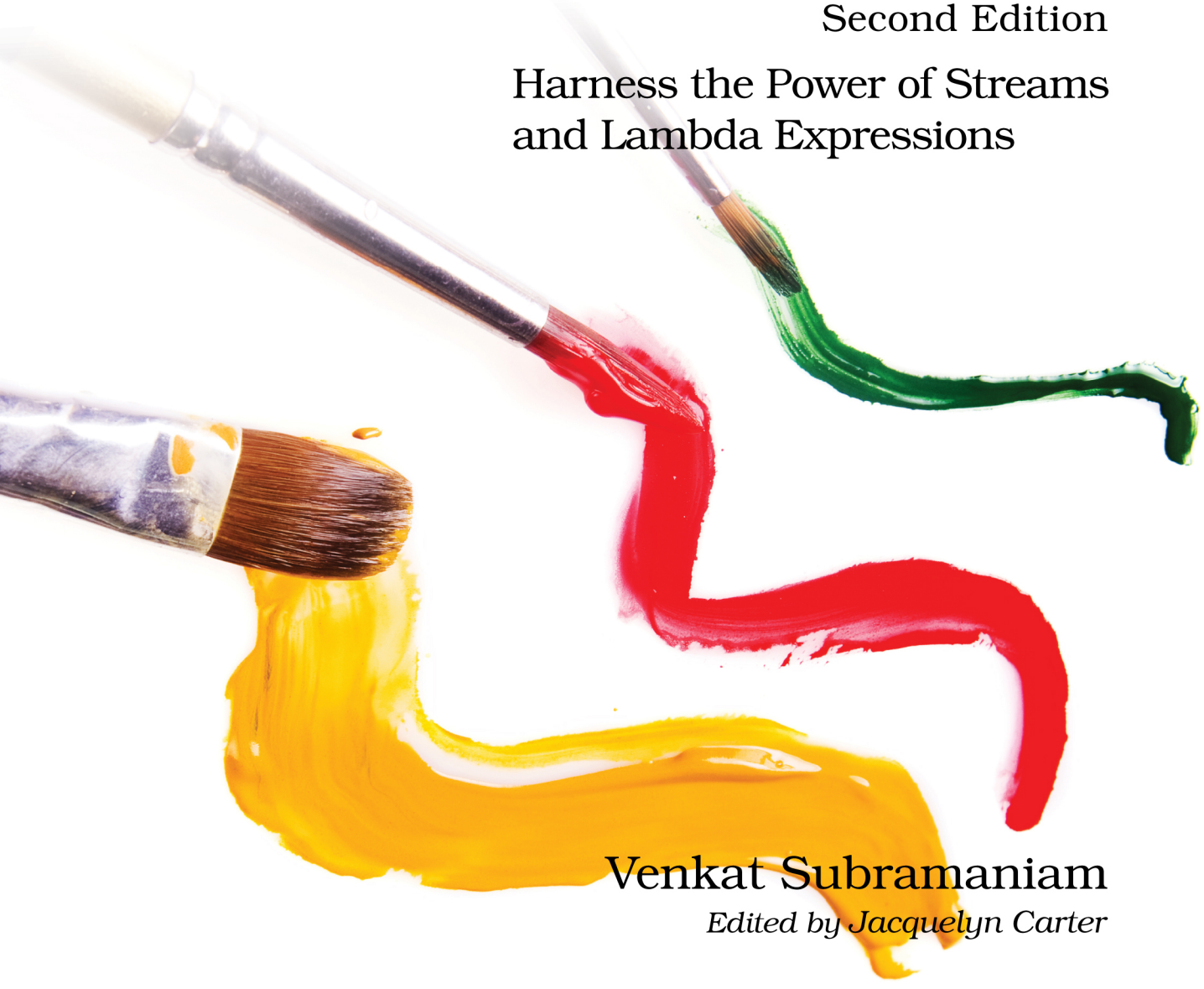
Raleigh, North Carolina

The Pragmatic Programmers

Functional Programming in Java

Second Edition

Harness the Power of Streams and Lambda Expressions



Venkat Subramaniam

Edited by Jacquelyn Carter

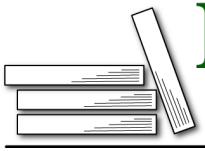
Functional Programming in Java, Second Edition

Harness the Power of Streams and Lambda Expressions

Venkat Subramaniam

The Pragmatic Bookshelf

Raleigh, North Carolina



Pragmatic Bookshelf

For our complete catalog of hands-on, practical, and Pragmatic content for software developers, please visit *https://pragprog.com*.

Sir Charles Antony Richard Hoare's quote is used by permission of the ACM.
Abelson and Sussman's quote is used under Creative Commons license.

The team that produced this book includes:

CEO: Dave Rankin
COO: Janet Furlow
Managing Editor: Tammy Coron
Development Editor: Jacquelyn Carter
Indexing: Potomac Indexing, LLC
Layout: Gilson Graphics
Founders: Andy Hunt and Dave Thomas

For sales, volume licensing, and support, please contact *support@pragprog.com*.

For international rights, please contact *rights@pragprog.com*.

ISBN-13: 978-1-68050-979-3
Encoded using the finest acid-free high-entropy binary digits.
Book version: P1.0—July 2023

To perseverance.

Preface

You're in for a treat. One of the most prominent and widely used languages in the world supports object-oriented, imperative style, and functional style programming. You can mix one of the most powerful tools—the object-oriented paradigm—with the imperative style, as we did in the past, or with the functional style, as you'll learn in this book, to reduce the complexity of code. We can do quite effectively in Java what was previously possible only on the JVM using other languages—this means more power to Java programmers.

I'm thankful to have had the privilege over the past few decades to program with multiple languages: C, C++, Java, C#, F#, Ruby, Groovy, Scala, Clojure, Kotlin, Erlang, Haskell, Elm, JavaScript.... When asked which one's my favorite, my resounding answer has been that it's not the language that excites me, but the way we program.

The science and engineering in programming drew me in, but it's the art in programming that keeps me. Coding has a lot in common with writing—there's more than one way to express our ideas. Java helps us write code using objects, and we can mix that with the functional capabilities to implement our designs and ideas.

The functional programming facilities in Java can make our code more expressive, easier to write, less error-prone, and easier to parallelize than with the imperative style. The functional way has been around for decades and widely used in languages like Lisp, Clojure, Erlang, Smalltalk, Scala, Groovy, and Ruby. It's not only a relatively new way in Java but also a better way.

Since coding is like writing, we can learn a few things from that field. In *On Writing Well [Zin01]*, William Zinsser recommends simplicity, clarity, and brevity. To create better applications, we can start by making the code simpler, clearer, and more concise. We'll explore how the functional style of programming in Java helps us do exactly that throughout this book.

Who's This Book For

This book is for programmers well versed in object-oriented programming in Java and keen to learn and apply the functional programming facilities. You'll need good programming experience in previous versions of Java, at least Java 5, to make the best use of this book.

Programmers mostly interested in JVM languages like Scala, Groovy, JRuby, Kotlin, and Clojure can benefit from the examples in this book and can relate back to the facilities offered in those languages. They can also use these examples to help fellow Java programmers on their teams.

Programmers experienced with the functional style of programming in other languages and who are now involved in Java projects can use this book, as well. They can learn how what they know translates to the specifics of the lambda expressions' usage in Java.

Programmers who are familiar with lambda expressions in Java can use this book to help coach and train their team members who are getting up to speed in this area.

What's in This Book

This book will help you get up to speed with the functional programming capabilities in Java, think in the elegant style, and benefit from the additions to the Java Development Kit (JDK) library. We'll take an example-driven approach to exploring the concepts. Rather than discuss the theory of functional programming, we'll dive into specific day-to-day tasks to apply the elegant style. This approach will quickly get these concepts under our belts so we can put them to real use on projects right away.

On the first read, take the time to go over the chapters sequentially as we build upon previously discussed concepts and examples. Each chapter closes with a quick summary to recap what was covered. Later, when working on applications, take a quick glance at any relevant example or section in the book. There's also a syntax appendix for quick reference.

Here's how the rest of the book is organized.

We discuss the functional style of programming, its benefits, and how it differs from the prevalent imperative style in Chapter 1, Hello, Lambda Expressions!, on page ?. We'll also look into how Java supports lambda expressions in this chapter.

The JDK collections have received some special functional treatment in Java, with new interfaces, classes, and methods that support functional-style operations. We'll explore these in Chapter 2, Using Collections, on page ?.

In Chapter 3, Strings, Comparators, and Filters, on page ?, we'll exploit functional style and lambda expressions to work with strings, implement the Comparator interface, and use filters for file selection.

The Collectors is one of the most versatile utility classes in the JDK with some amazing capabilities to transform data. We'll dive more deeply into that in Chapter 4, Transforming Data, on page ?.

In addition to using the functional-style facilities in the JDK, we can benefit from applying the elegant style in the design of methods and classes we create. We'll cover functional-style design techniques in Chapter 5, Designing with Lambda Expressions, on page ?.

The lambda expressions facilitate a code structure that helps delineate operations to manage object lifetimes and resource cleanup, as we'll discuss in Chapter 6, Working with Resources, on page ?.

We'll see lambda expressions shine in Chapter 7, Being Lazy, on page ?. They provide us the ability to postpone instance creation and method evaluations as well as create infinite lazy collections, and thereby improve the code's performance.

In Chapter 8, Optimizing Recursions, on page ?, we'll use lambda expressions to optimize recursions and achieve stellar performance using memoization techniques.

We'll put the techniques we cover in the book to some real use in Chapter 9, Composing Functions with Lambda Expressions, on page ?, where we'll transform objects, implement MapReduce, and safely parallelize a program with little effort.

Failures are part of system design and in Chapter 10, Error Handling, on page ? we'll look at how to deal with errors in functional programming and when working with the Streams API.

Since Java has been around for a few decades, there's a lot of imperative style code out there in the enterprises. You can use the many examples in Chapter 11, Refactoring to Functional Style, on page ? to practice thinking functionally and refactoring existing imperative style code to the functional style.

Writing good maintainable code requires more than learning the syntax and semantics. Knowing the dos and especially the don'ts can help create code

that truly is easier for the team to maintain. You'll learn about the idiomatic styles that work the best in Chapter 12, Functional Programming Idioms, on page ?.

In Chapter 13, Bringing It All Together, on page ?, we'll go over the key concepts and the practices needed to adopt functional programming techniques.

In Appendix 1, Starter Set of Functional Interfaces, on page ?, we'll take a glance at some of the most popular functional interfaces.

A quick overview of the syntax for functional interfaces, lambda expressions, and method/constructor references is in Appendix 2, Syntax Overview, on page ?.

The URLs mentioned throughout this book are gathered together for your convenience in Appendix 3, Web Resources, on page ?.

Java Version Used in This Book

To run most of the examples in this book you need at least Java 8. Some of the examples use features that are present in newer versions of Java.

Using automated scripts, the examples in this book have been tried out with the following version of Java:

```
openjdk version "18.0.1.1" 2022-04-22
OpenJDK Runtime Environment (build 18.0.1.1+2-6)
OpenJDK 64-Bit Server VM (build 18.0.1.1+2-6, mixed mode, sharing)
```

Take a few minutes to download the appropriate version of Java for your 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 belongs to a package:

```
package fpij;
```

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

Online Resources

A number of web resources referenced throughout the book are collected in Appendix 3, Web Resources, on page ?. Here are a couple that will help you get started with this book:

The OpenJDK website[1] for downloading the version of Java used in this book.

This book's page[2] on the Pragmatic Bookshelf website. From there you can download all the example source code for the book. You can also provide feedback 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 for some fun with lambda expressions and Stream…

Venkat Subramaniam
June 2023

1. https://openjdk.org/projects/jdk/18/
2. http://www.pragprog.com/titles/vsjava2e