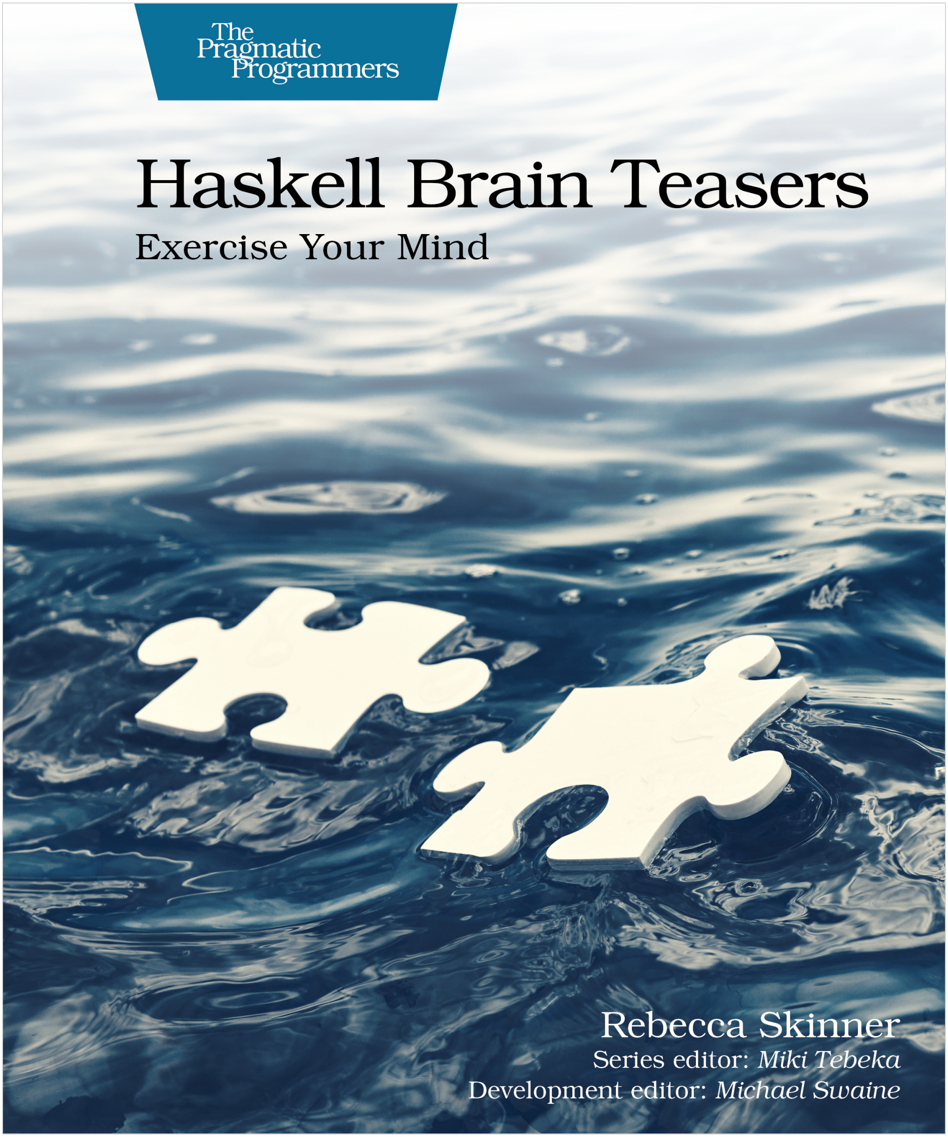

The Pragmatic Programmers

Haskell Brain Teasers

Exercise Your Mind

Rebecca Skinner

Series editor: *Miki Tebeka*

Development editor: *Michael Swaine*

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.

Introduction

Haskell's reputation is all about purity, safety, expressiveness, and power. It's a language admired by many, but loved by few. For those of us who do love it, the joy comes from its flexibility and the ease of expressing our ideas without compromise.

Haskell's flexibility is one of its greatest strengths: A predictable and composable grammar gives developers the latitude to write code with nuance and style, and language extensions allow Haskell to continually raise the bar for what an industrial-strength language can do. At the same time, Haskell's flexibility can be an obstacle to fluency. When a language pragma changes the semantic meaning of your code or you first learn that "an expression" really means yes, any expression, your reaction is more likely to be frustration than joy.

This book is here to help. As you work through short, focused puzzles you'll gain a clear, intuitive understanding of Haskell and start to see it as a cohesive whole. And don't forget to stop and smell the rose trees.

Who This Book Is For

It's for you if you are a Haskell expert, are somewhat familiar with Haskell, or are just generally interested in functional programming.

How to Read this Book

Each puzzle begins with a small self-contained Haskell program intended to run with GHC 9.12 or later using the GHC2024 language version. All required imports and language extensions are included in the puzzles and examples.

When you read the puzzle, try to predict what will happen when you compile and run the program. For example, will it:

1. Run normally and print something to the screen?
2. Run slowly or use a lot of memory?

3. Fail to compile due to a syntax or type error?
4. Hang indefinitely in an infinite loop or deadlock?
5. Crash with a runtime exception?

Example Code and Tests

The sample code for this book includes standalone files for each puzzle, full example code, unit and property tests, and a complete, reproducible nix environment you can use to build and run the code. It's available on the Pragmatic Bookshelf website.[1] There, you can download the code, participate in discussions, and post issues if needed.

Puzzle Organization and Difficulty

You can work through the puzzles in any order. They're grouped thematically, based on which part of the language they focus on. For example, Part I focuses on challenges related to lazy evaluation, while puzzles in Part IV will lean more into reasoning about types.

Puzzles in each part are organized roughly from easiest to hardest. Since difficulty varies depending on your personal experience and background, you might run across an easier puzzle that's challenging for you- or you might find a difficult puzzle quite easy. If you do find a puzzle that you can't solve now, feel free to skip ahead and come back to it later.

[1] http://pragprog.com/titles/haskellbt