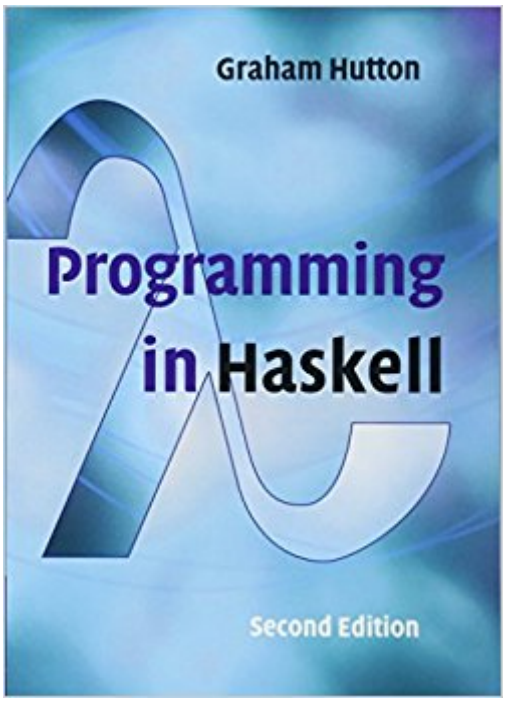
Programming In Haskell



Synopsis

Haskell is a purely functional language that allows programmers to rapidly develop clear, concise, and correct software. The language has grown in popularity in recent years, both in teaching and in industry. This book is based on the author's experience of teaching Haskell for more than twenty years. All concepts are explained from first principles and no programming experience is required, making this book accessible to a broad spectrum of readers. While Part I focuses on basic concepts, Part II introduces the reader to more advanced topics. This new edition has been extensively updated and expanded to include recent and more advanced features of Haskell, new examples and exercises, selected solutions, and freely downloadable lecture slides and example code. The presentation is clean and simple, while also being fully compliant with the latest version of the language, including recent changes concerning applicative, monadic, foldable, and traversable types.

Book Information

Paperback: 318 pages

Publisher: Cambridge University Press; 2 edition (September 12, 2016)

Language: English

ISBN-10: 1316626229

ISBN-13: 978-1316626221

Product Dimensions: 5.4 x 0.7 x 8.5 inches

Shipping Weight: 1.3 pounds (View shipping rates and policies)

Average Customer Review: 4.9 out of 5 stars 12 customer reviews

Best Sellers Rank: #34,411 in Books (See Top 100 in Books) #16 inÂ Books > Textbooks > Computer Science > Object-Oriented Software Design #50 inÂ Books > Computers & Technology > Programming > Software Design, Testing & Engineering > Object-Oriented Design #191 inÂ Books > Textbooks > Computer Science > Programming Languages

Customer Reviews

"The skills you acquire by studying this book will make you a much better programmer no matter what language you use to actually program in." Erik Meijer, Facebook, from the ForewordReview of previous edition: "The best introduction to Haskell available. There are many paths towards becoming comfortable and competent with the language but I think studying this book is the quickest path. I urge readers of this magazine to recommend Programming in Haskell to anyone who has been thinking about learning the language." Duncan Coutts, The Monad.ReaderReview of

previous edition: "Where this book excels is in the order and style of its exposition ... With its ripe selection of examples and its careful clarity of exposition, the book is a welcome addition to the introductory functional programming literature." Journal of Functional Programming

This edition has been extensively updated and expanded, with new chapters covering recent and more advanced features of Haskell, new examples and exercises, and freely downloadable lecture slides and example code. All concepts are explained from first principles and no programming experience is required, making it accessible to a broad range of readers.

This is the first Haskell book I have read and my very first encounter with the language itself was from reading this book. The author has done an amazing work by his succinct writing that captures the foundations of Haskell through to more advanced components of the language like Functors, Applicatives, and Monads, and more.The exercises are well posed problems and help extend and test your understanding of the materials covered. I’m now reading Programming Haskell from first principles because I want another text that will show me examples of building [large] projects. The skills I picked up from PiH are no doubt showing to be useful as I read PHffp.I am delighted that I am learning Haskell, thanks to beauty of the language and to Graham’s superior authorship of this text.

Concepts are clear, concise and well-explained.I’ve made my way through 10 of the 17 chapters, and I can say that as a functional programming and Haskell beginner, it’s been a great introductory resource for me.

Most fun you can have in front of a computer (doing something that involves the brain)!

What I didn’t expect was for this book to double as resource for quick reference as it is very well laid out.

Wisely concise

This is a beautiful book for people who want to learn Haskell and functional programming as an advanced problem-solving tool.First the good bits: The author’s extensive university teaching and research experience shines throughout the book. Starting from the fundamental principles of

functional programming, the author gently introduces the basic concepts and constructs of Haskell and strongly-typed functional programming. There are a lot of examples to demonstrate how the introduced concepts of Haskell and techniques of functional programming can be used to analyze and design solutions to problems of various complexity.After introducing the basic building blocks of Haskell in the first part, the author goes on to introduce more complex topics such as monadic parsing, as well as modern Haskell concepts such as Applicative, Traversable and Foldable type classes. Following these, another very important notion, "lazy evaluation" is introduced and its usage is described, why and how it fits into Haskell explained with examples.The exercises at the end of the chapter are carefully planned, and serve to force the reader's mind to understand concepts by forcing her to practice and think by herself.I have to say that the final two chapters is where the book totally shines. First the author introduces what it means to reason about programs and shows how systematic thinking can be applied to designing a solution. In the chapter that follows, that is the final chapter, the reader sees the full power of the ideas developed in the previous chapter applied by calculating compilers, that is, starting from a specification for a programming language, to reaching a correct compiler that can parse the statements in that language and evaluate them to produce the results. The way to do is by using induction and realizing that this systematic method can be applied to languages of ever increasing complexity is mind-blowing moment in itself.Now the not-so-good bits: Even though the books exemplifies how to break down problems into small pieces and how to compose small building blocks to create bigger and more complex Haskell solutions, it is definitely not enough for the "working programmer". That is, you will definitely learn a lot of important and critical Haskell principles and techniques from this book but you'll also miss a lot of other important aspects such as:- A stronger focus on type-driven program design in Haskell- Building Haskell projects and packages- Writing tests, both traditional unit tests, and extensive automated QuickCheck style tests- Profiling your programs- How to properly benchmark your programs- More detailed parsing techniques and libraries- Web-based programming- Network programming- And few more topics that will be important if you're working in a team of Haskell developers, working to produce software products and services for your customers.And you'll definitely need another book for that, the strongest contender being "Haskell Programming from First Principles" as of 2017.Having said that, I'd still consider this Second Edition of Programming in Haskell by Hutton to be perfectly suitable for a modern Haskell introduction, provided that it is backed up by a teacher in a classroom environment; someone that can fill in the missing parts. The final chapters of the book will definitely appeal to programmers and students who want to continue their journey into the more research-oriented areas such as compiler

design.Overall, I’m more than satisfied to add this book to my Haskell and functional programming shelf, and whenever I’ll need concise descriptions of fundamental as well as modern ideas, this will be among my go-to books for enlightenment.

The book is divided in two parts. The first part focuses on the basics of the language. Topics include types & classes, functions, recursion, list comprehensions, folds. The concepts are exposed clearly and concisely (chapters are rather short). Exercices at the end of each chapter will help you digest what was covered and are a must (there are solutions to selected exercises at the end of the book). If you have read Learn You a Haskell for Great Good you will be in known territory. If you haven’t read LYAHFGG, I recommend going straight for Graham Hutton’s book. Again, the clarity and conciseness will help you make strides quickly.The second part introduces more advanced topics. These include IO, monads, parsing, foldables & friends, lazy evaluation, program proof. Two of the things that helped me greatly were: 1) The extended examples using the IO monad with do notation (interactive programming, unbeatable tic-tac-toe, calculator). I feel more confident now writing code that needs to deal with IO. 2) The explanation on how lazy evaluation works vs eager evaluation. It is the best I have found so far. Ever wondered how to work out self-referencing lists ? Having a solid understanding of lazy evaluation is the way to go and Graham’s chapter provides just that.There are many things that I have always wondered how to do in Haskell and that found their answer through the chapters or through the examples. For example in imperative languages, you want sometimes to loop and deal with the elements one by one, also using the element from the previous iteration. By contrast, mapping or folding allows you to deal with the current element only. Graham shows you how to do it: zip a list with its tail!There are other topics that are introduced, and that invite the reader to continue on its path to learning Haskell. Monads, Functors, Applicatives and their laws (you will prove some of them for a few class instances in later exercises) are clearly presented. Monadic parsing then shows how to write a grammar for parsing in Haskell. I suspect it provides a nice introduction to parsec. The last two chapters introduce topics close to the author’s field of research. Reasoning about programs is about induction to prove program equivalence and in turn, correctness. Calculating compilers is about deriving a (correct) program (simple compiler) from specifications.Being 300 pages long, the book cannot cover everything Haskell. A few examples not covered I can think of: arrays/vectors (sometimes you want performance and lists aren’t always a good fit), modules, more in-depth information about the type system. However I like that Programming in Haskell is small and doesn’t pretend to be complete. These 300 pages are extremely concise and are definitely worth one’s time.This book is a gem and I cannot recommend it

highly enough.

This new edition is longer and much better. The first edition was too short. It also had this annoying habit of using math symbols rather than what you would actually type in a Haskell source file. Bizarre and annoying to check an appendix to see the lookup table.This new edition is a great intro to Haskell. It’s probably the best intro now. I prefer it over the others that I’ve read.

Analytics and Python Programming: 2 Bundle Manuscript: Beginners Guide to Learn Data Analytics, Predictive Analytics and Data Science with Python Programming