Real World Ocaml Functional Programming For The Masses

A Real World Guide to Programming
Trends in Functional Programming
Functional Programming in C++
Functional Programming in JavaScript
Haskell Programming from First Principles
DSLs in Action
With examples in F# and C#
More OCaml
Functional Programming For Dummies
OCaml from the Very Beginning
An Introduction
Algorithms, Methods & Diversions
Programming in Martin-Löf's Type Theory
Practical OCaml
Learn Type-Driven Development
Real World Haskell
Hands-On Functional Programming with TypeScript
Pearls of Functional Algorithm Design
Functional C
Real-World Functional Programming
Functional Programming for the Masses
Functional programming for the masses
Type-Driven Development with Idris
Practical Haskell
Programming Language Concepts
Verified Functional Programming in Agda
Real World OCaml
Domain Modeling Made Functional
The Formal Semantics of Programming Languages
Expert F# 4.0
The Little Prover
The Functional Approach to Programming
Functional Programming in Scala
18th International Symposium, TFP 2017, Canterbury, UK, June 19-21, 2017, Revised Selected Papers
Explore functional and reactive programming to create robust and testable TypeScript applications
Foundations of F#
Realm of Racket
Building Powerful Cross-Platform Environments in JavaScript

Benefit from type systems to build reliable and safe applications using ReasonML 3

PHOEBE LOWERY

A Real World Guide to Programming

Lulu.com

This book uses a functional programming language (F#) as a metalanguage to present all concepts and examples, and thus has an operational flavour, enabling practical experiments and exercises. It includes basic concepts such as abstract syntax, interpretation, stack machines, compilation, type checking, garbage collection, and real machine code. Also included are more advanced topics on polymorphic types, type inference using unification, co- and contravariant types, continuations, and backwards code generation with on-the-fly peephole optimization. This second edition includes two new chapters. One describes compilation and type checking of a full functional language, tying together the previous chapters. The other describes how to compile a C subset to real (x86) hardware, as a smooth extension of the previously presented compilers.The examples present several interpreters and compilers for toy languages, including compilers for a small but usable subset of C, abstract machines, a garbage collector, and ML-style polymorphic type inference. Each chapter has exercises. Programming Language Concepts covers practical construction of lexers and parsers, but not regular expressions, automata and grammars, which are well covered already. It discusses the design and technology of Java and C# to strengthen students' understanding of these widely used languages.

Trends in Functional Programming

Springer

The Formal Semantics of Programming Languages provides the basic mathematical techniques necessary for those who are beginning a study of the semantics and logics of programming languages. These techniques will allow students to invent, formalize, and justify rules with which to reason about a variety of programming languages. Although the treatment is elementary, several of the topics covered are drawn from recent research, including the vital area of concurency. The book contains many exercises ranging from simple to miniprojects.Starting with basic set theory, structural operational semantics is introduced as a way to define the meaning of programming languages along with associated proof techniques. Denotational and axiomatic semantics are illustrated on a simple language of while-programs, and fall proofs are given of the equivalence of the operational and denotational semantics and soundness and relative completeness of the axiomatic semantics. A proof of Godel's incompleteness theorem, which emphasizes the impossibility of achieving a fully complete axiomatic semantics, is included. It is supported by an appendix providing an introduction to the theory of computability based on while-programs. Following a presentation of domain theory, the semantics and methods of proof for several functional languages are treated. The simplest language is that of recursion equations with both call-by-value and call-by-name evaluation. This work is extended to lan guages with higher and recursive types, including a treatment of the eager and lazy lambda-calculi. Throughout, the

relationship between denotational and operational semantics is stressed, and the proofs of the correspondence between the operation and denotational semantics are provided. The treatment of recursive types - one of the more advanced parts of the book - relies on the use of information systems to represent domains. The book concludes with a chapter on parallel programming languages, accompanied by a discussion of methods for specifying and verifying nondeterministic and parallel programs.
Functional Programming in C++ Apress
"This work strikes a balance between the pure functional aspects ofF# and the object-oriented and imperative features that make it souseful in practice, enable .NET integration, and make large-scaledata processing possible." —Thore Graepel, PhD, Researcher, Microsoft Research Ltd. Over the next five years, F# is expected to become one of theworld's most popular functional programming languages forscientists of all disciplines working on the Windows platform. F#is free and, unlike MATLAB® and other software withnumerical/scientific origins, is a full-fledged programminglanguage. Developed in consultation with Don Syme of Microsoft ResearchLtd.—who wrote the language—F# for Scientistsexplains and demonstrates the powerful features of this importantnew programming language. The book assumes no prior experience andguides the reader from the basics of computer programming to theimplementation of state-of-the-art algorithms. F# for Scientists begins with coverage of introductorymaterial in the areas of functional programming, .NET, andscientific computing, and goes on to explore: Program structure Optimization Data structures Libraries Numerical analysis Databases Input and output Interoperability Visualization Screenshots of development using Visual Studio are used toillustrate compilation, debugging, and interactive use, whilecomplete examples of a few whole programs are included to givereaders a complete view of F#'s capabilities. Written in a clear and concise style, F# for Scientistsis well suited for researchers, scientists, and developers who wantto program under the Windows platform. It also serves as an idealsupplemental text for advanced undergraduate and graduate studentswith a background in science or engineering.
Functional Programming in JavaScript "O'Reilly Media, Inc."
Introduces fundamental techniques for reasoning mathematically about functional programs. Ideal for a first- or second-year undergraduate course.
Haskell Programming from First Principles Apress
Learn from F#'s inventor to become an expert in the latest version of this powerful programming language so you can seamlessly integrate functional, imperative, object-oriented, and query programming style flexibly and elegantly to solve any programming problem. Expert F# 4.0 will help you achieve unrivaled levels of programmer productivity and program clarity across multiple platforms including Windows, Linux, Android, OSX, and iOS as well as HTML5 and GPUs. F# 4.0 is a mature, open source, cross-platform, functional-first programming language which empowers users and organizations to tackle complex computing problems with simple, maintainable, and robust code. Expert F# 4.0 is: A comprehensive guide to the latest version of F# by the inventor of the language A treasury of

F# techniques for practical problem-solving An in-depth case book of F# applications and F# 4.0 concepts, syntax, and features Written by F#'s inventor and two major F# community members, Expert F# 4.0 is a comprehensive and in-depth guide to the language and its use. Designed to help others become experts, the book quickly yet carefully describes the paradigms supported by F# language, and then shows how to use F# elegantly for a practical web, data, parallel and analytical programming tasks. The world's experts in F# show you how to program in F# the way they do!

DSLs in Action Packt Publishing Ltd

Objective Caml (OCaml) is an open source programming language that utilizes both functional and object oriented programming. Practical OCaml teaches Objective Caml in a straightforward manner, teaching all the features of this functional programming language by example. You will learn how to utilize OCaml to create a simple database, do reporting, and create a spam filter. You will also learn how to do complex log file scanning, create your own network servers by creating a ShoutCast server, and create a web crawler. By the book's conclusion, you will be well on your way to creating your own applications with OCaml.

With examples in F# and C# Franklin Beedle & Associates

This is the first book to bring F# to the world. It is likely to have many imitators but few competitors. Written by F# evangelist, Rob Pickering, and tech reviewed by F#’s inventor, Don Syme, it is an elegant, comprehensive introduction to all aspects of the language and an incisive guide to using F# for real-world professional development. It is detailed, yet clear and concise, and suitable for readers at any level of experience. Every professional .NET programmer needs to learn about Functional Programming (FP), and there’s no better way to do it than by learning F# — and no easier way to learn F# than from this book.

More OCaml Cambridge University Press

Advanced text on how to program in the functional way; has exercises, solutions and code.

Functional Programming For Dummies Cambridge University Press

This easy-to-use, fast-moving tutorial introduces you to functional programming with Haskell. You'll learn how to use Haskell in a variety of practical ways, from short scripts to large and demanding applications. Real World Haskell takes you through the basics of functional programming at a brisk pace, and then helps you increase your understanding of Haskell in real-world issues like I/O, performance, dealing with data, concurrency, and more as you move through each chapter.

OCaml from the Very Beginning Simon and Schuster

Summary Functional Programming in Scala is a serious tutorial for programmers looking to learn FP and apply it to the everyday business of coding. The book guides readers from basic techniques to advanced topics in a logical, concise, and clear progression. In it, you'll find concrete examples and exercises that open up the world of functional programming. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Functional programming (FP) is a style of software development emphasizing functions that don't depend on program state. Functional code is easier to test

and reuse, simpler to parallelize, and less prone to bugs than other code. Scala is an emerging JVM language that offers strong support for FP. Its familiar syntax and transparent interoperability with Java make Scala a great place to start learning FP. About the Book Functional Programming in Scala is a serious tutorial for programmers looking to learn FP and apply it to their everyday work. The book guides readers from basic techniques to advanced topics in a logical, concise, and clear progression. In it, you'll find concrete examples and exercises that open up the world of functional programming. This book assumes no prior experience with functional programming. Some prior exposure to Scala or Java is helpful. What's Inside Functional programming concepts The whys and hows of FP How to write multicore programs Exercises and checks for understanding About the Authors Paul Chiusano and Rúnar Bjarnason are recognized experts in functional programming with Scala and are core contributors to the Scalaz library. Table of Contents PART 1 INTRODUCTION TO FUNCTIONAL PROGRAMMING What is functional programming? Getting started with functional programming in Scala Functional data structures Handling errors without exceptions Strictness and laziness Purely functional state PART 2 FUNCTIONAL DESIGN AND COMBINATOR LIBRARIES Purely functional parallelism Property-based testing Parser combinators PART 3 COMMON STRUCTURES IN FUNCTIONAL DESIGN Monoids Monads Applicative and traversable functors PART 4 EFFECTS AND I/O External effects and I/O Local effects and mutable state Stream processing and incremental I/O

An Introduction Apress

Haskell Programming makes Haskell as clear, painless, and practical as it can be, whether you're a beginner or an experienced hacker. Learning Haskell from the ground up is easier and works better. With our exercise-driven approach, you'll build on previous chapters such that by the time you reach the notorious Monad, it'll seem trivial.

Algorithms, Methods & Diversions

Cambridge University Press

Functional programming is a very powerful programming paradigm that can help us to write better code. This book presents essential functional and reactive programming concepts in a simplified manner using Typescript.

Programming in Martin-Löf's Type Theory

Real World OCamlFunctional programming for the masses

You want increased customer satisfaction, faster development cycles, and less wasted work. Domain-driven design (DDD) combined with functional programming is the innovative combo that will get you there. In this pragmatic, down-to-earth guide, you'll see how applying the core principles of functional programming can result in software designs that model real-world requirements both elegantly and concisely - often more so than an object-oriented approach. Practical examples in the open-source F# functional language, and examples from familiar business domains, show you how to apply these techniques to build software that is business-focused, flexible, and high quality. Domain-driven design is a well-established approach to designing software that ensures that domain experts and developers work together effectively to create high-quality software. This book is the first to combine DDD with techniques from statically typed functional programming.

This book is perfect for newcomers to DDD or functional programming - all the techniques you need will be introduced and explained. Model a complex domain accurately using the F# type system, creating compilable code that is also readable documentation---ensuring that the code and design never get out of sync. Encode business rules in the design so that you have "compile-time unit tests," and eliminate many potential bugs by making illegal states unrepresentable. Assemble a series of small, testable functions into a complete use case, and compose these individual scenarios into a large-scale design. Discover why the combination of functional programming and DDD leads naturally to service-oriented and hexagonal architectures. Finally, create a functional domain model that works with traditional databases, NoSQL, and event stores, and safely expose your domain via a website or API. Solve real problems by focusing on real-world requirements for your software. What You Need: The code in this book is designed to be run interactively on Windows, Mac and Linux.You will need a recent version of F# (4.0 or greater), and the appropriate .NET runtime for your platform.Full installation instructions for all platforms at fsharp.org.

Practical OCaml O'Reilly Media, Incorporated

Real World OCamlFunctional programming for the masses"O'Reilly Media, Inc."

Learn Type-Driven Development Addison Wesley Longman

Summary Functional Programming in C++ teaches developers the practical side of functional programming and the tools that C++ provides to develop software in the functional style. This in-depth guide is full of useful diagrams that help you understand FP concepts and begin to think functionally. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Well-written code is easier to test and reuse, simpler to parallelize, and less error prone. Mastering the functional style of programming can help you tackle the demands of modern apps and will lead to simpler expression of complex program logic, graceful error handling, and elegant concurrency. C++ supports FP with templates, lambdas, and other core language features, along with many parts of the STL. About the Book Functional Programming in C++ helps you unleash the functional side of your brain, as you gain a powerful new perspective on C++ coding. You'll discover dozens of examples, diagrams, and illustrations that break down the functional concepts you can apply in C++, including lazy evaluation, function objects and invokables, algebraic data types, and more. As you read, you'll match FP techniques with practical scenarios where they offer the most benefit. What's inside Writing safer code with no performance penalties Explicitly handling errors through the type system Extending C++ with new control structures Composing tasks with DSLs About the Reader Written for developers with two or more years of experience coding in C++. About the Author Ivan Čukić is a core developer at KDE and has been coding in C++ since 1998. He teaches modern C++ and functional programming at the Faculty of Mathematics at the University of Belgrade. Table of Contents Introduction to functional programming Getting started with functional programming Function objects Creating new functions

from the old ones Purity: Avoiding mutable state Lazy evaluation Ranges Functional data structures Algebraic data types and pattern matching Monads Template metaprogramming Functional design for concurrent systems Testing and debugging

Real World Haskell MIT Press

Agda is an advanced programming language based on Type Theory. Agda's type system is expressive enough to support full functional verification of programs, in two styles. In external verification, we write pure functional programs and then write proofs of properties about them. The proofs are separate external artifacts, typically using structural induction. In internal verification, we specify properties of programs through rich types for the programs themselves. This often necessitates including proofs inside code, to show the type checker that the specified properties hold. The power to prove properties of programs in these two styles is a profound addition to the practice of programming, giving programmers the power to guarantee the absence of bugs, and thus improve the quality of software more than previously possible. Verified Functional Programming in Agda is the first book to provide a systematic exposition of external and internal verification in Agda, suitable for undergraduate students of Computer Science. No familiarity with functional programming or computer-checked proofs is presupposed. The book begins with an introduction to functional programming through familiar examples like booleans, natural numbers, and lists, and techniques for external verification. Internal verification is considered through the examples of vectors, binary search trees, and Braun trees. More advanced material on type-level computation, explicit reasoning about termination, and normalization by evaluation is also included. The book also includes a medium-sized case study on Huffman encoding and decoding.

Hands-On Functional Programming with TypeScript Oxford University Press

Learn how to solve day-to-day problems in data processing, numerical computation, system scripting, and database-driven web applications with the OCaml multi-paradigm programming language. This hands-on book shows you how to take advantage of OCaml’s functional, imperative, and object-oriented programming styles with recipes for many real-world tasks. You’ll start with OCaml basics, including how to set up a development environment, and move toward more advanced topics such as the module system, foreign-function interface, macro language, and the ocamlbuild system. Quickly learn how to put OCaml to work for writing succinct and readable code.

Pearls of Functional Algorithm Design Simon and Schuster

Functional programming languages like F#, Erlang, and Scala are attractingattention as an efficient way to handle the new requirements for programmingmulti-processor and high-availability applications. Microsoft's new F# is a truefunctional language and C# uses functional language features for LINQ andother recent advances. Real-World Functional Programming is a unique tutorial that explores thefunctional programming model through the F# and C# languages. The clearlypresented ideas and examples teach readers how functional programming differsfrom other approaches. It explains how ideas look in

F#-a functionallanguage-as well as how they can be successfully used to solve programmingproblems in C#. Readers build on what they know about .NET and learn wherea functional approach makes the most sense and how to apply it effectively inthose cases. The reader should have a good working knowledge of C#. No prior exposure toF# or functional programming is required. Purchase of the print book comes with an offer of a free PDF, ePub, and Kindle eBook from Manning. Also available is all code from the book.

Functional C Coherent Press

In recent years, several formalisms for program construction have appeared. One such formalism is the type theory developed by Per Martin-Löf. Well suited as a theory for program construction, it makes possible the expression of both specifications and programs within the same formalism. Furthermore, the proof rules can be used to derive a correct program from a specification as well as to verify that a given program has a certain property. This book contains a thorough introduction to type theory, with information on polymorphic sets, subsets, monomorphic sets, and a full set of helpful examples.

Real-World Functional Programming No Starch Press

This fast-moving tutorial introduces you to OCaml, an industrial-strength programming language designed for expressiveness, safety, and speed. Through the book's many examples, you'll quickly learn how OCaml stands out as a tool for writing fast, succinct, and readable systems code. Real World OCaml takes you through the concepts of the language at a brisk pace, and then helps you explore the tools and techniques that make OCaml an effective and practical tool. In the book's third section, you'll delve deep into the details of the compiler toolchain and OCaml's simple and efficient runtime system. Learn the foundations of the language, such as higher-order functions, algebraic data types, and modules Explore advanced features such as functors, first-class modules, and objects Leverage Core, a comprehensive general-purpose standard library for OCaml Design effective and reusable libraries, making the most of OCaml's approach to abstraction and modularity Tackle practical programming problems from command-line parsing to asynchronous network programming Examine profiling and interactive debugging techniques with tools such as GNU gdb