

Functional Programming In Scala Runar Bjarnason

Introduction to the Art of Programming Using ScalaBeing GeekA Companion Booklet to Functional Programming in ScalaFunctional Programming, SimplifiedScala for Data ScienceFunctional and Reactive Domain ModelingReactive Systems ArchitectureScala for the ImpatientAkka in ActionModern Systems Programming with Scala NativeIntroduction to Programming and Problem-Solving Using Scala, Second EditionHaskellProgramming in ScalaPlay for ScalaCategory Theory for Programmers (Scala Edition, Paperback)Scala Design PatternsScala in ActionProfessional ScalaStream Processing with Apache FlinkScala CookbookThinking Functionally with HaskellReactive Design PatternsThe Go Programming LanguageAn Invitation to Applied Category TheoryModern Java in ActionScala PuzzlersFunctional Programming in ScalaFunctional Concurrency in .NetFunctional Programming in JavaLearning ScalaScala in DepthProgramming ScalaAcoustic Emission TestingPractical FP in Scala (hard-Cover)The Little TyperType-Driven Development with IdrisScala MicroservicesSpark: The Definitive GuideTesting in ScalaProgramming Language Concepts

Introduction to the Art of Programming Using Scala

Being Geek

Praise for the first edition: "The well-written, comprehensive book[is] aiming to become a de facto reference for the language and its features and capabilities. The pace is appropriate for beginners; programming concepts are introduced progressively through a range of examples and then used as tools for building applications in various domains, including sophisticated data structures and algorithms. Highly recommended. Students of all levels, faculty, and professionals/practitioners.?" —D. Papamichail, University of Miami in CHOICE Magazine ? Mark Lewis'? Introduction to the Art of Programming Using Scala? was the first textbook to use Scala for introductory CS courses. Fully revised and expanded, the new edition of this popular text has been divided into two books. Introduction to Programming and Problem-Solving Using Scala is designed to be used in first semester college classrooms to teach students beginning programming with Scala. The book focuses on the key topics students need to know in an introductory course, while also highlighting the features that make Scala a great programming language to learn. The book is filled with end-of-chapter projects and exercises, and the authors have also posted a number of different supplements on the book website. Video lectures for each chapter in the book are also available on YouTube. The videos show construction of code from the ground up and this type of "live coding" is invaluable for learning to program, as it

allows students into the mind of a more experienced programmer, where they can see the thought processes associated with the development of the code. About the Authors Mark Lewis is a Professor at Trinity University. He teaches a number of different courses, spanning from first semester introductory courses to advanced seminars. His research interests included simulations and modeling, programming languages, and numerical modeling of rings around planets with nearby moons.? Lisa Lacher is an Assistant Professor at the University of Houston, Clear Lake with over 25 years of professional software development experience. She teaches a number of different courses spanning from first semester introductory courses to graduate level courses. Her research interests include Computer Science Education, Agile Software Development, Human Computer Interaction and Usability Engineering, as well as Measurement and Empirical Software Engineering.

A Companion Booklet to Functional Programming in Scala

Introducing functional programming in the Haskell language, this book is written for students and programmers with little or no experience. It emphasises the process of crafting programmes, problem solving and avoiding common programming pitfalls. Covering basic functional programming, through abstraction to larger scale programming, students are lead step by step through the basics, before being introduced to more advanced topics. This edition includes new material on testing and domain-specific languages and a variety of new examples

and case studies, including simple games. Existing material has been expanded and re-ordered, so that some concepts – such as simple data types and input/output – are presented at an earlier stage.

Functional Programming, Simplified

An introduction to dependent types, demonstrating the most beautiful aspects, one step at a time. A program's type describes its behavior. Dependent types are a first-class part of a language, and are much more powerful than other kinds of types; using just one language for types and programs allows program descriptions to be as powerful as the programs they describe. The Little Typer explains dependent types, beginning with a very small language that looks very much like Scheme and extending it to cover both programming with dependent types and using dependent types for mathematical reasoning. Readers should be familiar with the basics of a Lisp-like programming language, as presented in the first four chapters of *The Little Schemer*. The first five chapters of *The Little Typer* provide the needed tools to understand dependent types; the remaining chapters use these tools to build a bridge between mathematics and programming. Readers will learn that tools they know from programming—pairs, lists, functions, and recursion—can also capture patterns of reasoning. *The Little Typer* does not attempt to teach either practical programming skills or a fully rigorous approach to types. Instead, it demonstrates the most beautiful aspects as simply as possible, one step at a time.

Scala for Data Science

Design, build, and run Microservices using Scala elegantly About This Book Build robust microservices using Play Framework and Lagom Model your data for highly interactive applications and scale using Event Sourcing & CQRS Build applications that are resilient to failures by using Message Passing for communication Deploy and manage Scala Microservices for scale by using docker containers with Kubernetes for orchestration Who This Book Is For It is assumed that the reader knows Scala or is proficient in a competent programming language such as Java, C#, Ruby, and so on, with some exposure to Scala. Some experience with writing web services would also be ideal but not mandatory. What You Will Learn Learn the essentials behind Microservices, the advantages and perils associated with them Build low latency, high throughput applications using Play and Lagom Dive deeper with being asynchronous and understand the superiority it provides Model your complex domain data for scale and simplicity with CQRS and Event Sourcing Be resilient to failures by using message passing Look at best practices of version control workflow, testing, continuous integration and deployments Understand operating system level virtualization using Linux Containers. Docker is used to explain how containers work Automate your infrastructure with kubernetes In Detail In this book we will learn what it takes to build great applications using Microservices, the pitfalls associated with such a design and the techniques to avoid them. We learn to build highly performant applications using Play

Framework. You will understand the importance of writing code that is asynchronous and nonblocking and how Play leverages this paradigm for higher throughput. The book introduces Reactive Manifesto and uses Lagom Framework to implement the suggested paradigms. Lagom teaches us to: build applications that are scalable and resilient to failures, and solves problems faced with microservices like service gateway, service discovery, communication and so on. Message Passing is used as a means to achieve resilience and CQRS with Event Sourcing helps us in modelling data for highly interactive applications. The book also shares effective development processes for large teams by using good version control workflow, continuous integration and deployment strategies. We introduce Docker containers and Kubernetes orchestrator. Finally, we look at end to end deployment of a set of scala microservices in kubernetes with load balancing, service discovery and rolling deployments. Style and approach The book will step through each of the core microservice concepts in Scala, building an overall picture of their capabilities. This book adopts a systematic approach, allowing you to build upon what you've learnt in previous chapters. By the end of this book you'll have an understanding of the complex aspects of building microservices in Scala and will be able to take that knowledge with you into further projects. ng of the complex aspects of building Microservices in Scala and will be able to take that knowledge with you onto whatever project calls for it

Functional and Reactive Domain Modeling

Write efficient, clean, and reusable code with Scala About This Book Unleash the power of Scala and apply it in the real world Increase your efficiency by leveraging the power of Creational, Structural, Behavioural, and Functional design patterns Build object oriented and functional applications quickly and effectively Who This Book Is For If you want to increase your understanding of Scala and apply it to real-life application development, then this book is for you. We've also designed the book to be used as a quick reference guide while creating applications. Previous Scala programming knowledge is expected. What You Will Learn Immerse yourself in industry-standard design patterns—structural, creational, and behavioral—to create extraordinary applications Feel the power of traits and their application in Scala Implement abstract and self types and build clean design patterns Build complex entity relationships using structural design patterns Create applications faster by applying functional design patterns In Detail Scala has become increasingly popular in many different IT sectors. The language is exceptionally feature-rich which helps developers write less code and get faster results. Design patterns make developer's lives easier by helping them write great software that is easy to maintain, runs efficiently and is valuable to the company or people concerned. You will learn about the various features of Scala and be able to apply well-known, industry-proven design patterns in your work. The book starts off by focusing on some of the most interesting features of Scala while using practical real-world examples. We will also cover the popular "Gang of Four" design patterns

and show you how to incorporate functional patterns effectively. By the end of this book, you will have enough knowledge and understanding to quickly assess problems and come up with elegant solutions. Style and approach The design patterns in the book will be explained using real-world, step-by-step examples. For each design pattern, there will be hints about when to use it and when to look for something more suitable. This book can also be used as a practical guide, showing you how to leverage design patterns effectively.

Reactive Systems Architecture

Leverage the power of Scala with different tools to build scalable, robust data science applications About This Book A complete guide for scalable data science solutions, from data ingestion to data visualization Deploy horizontally scalable data processing pipelines and take advantage of web frameworks to build engaging visualizations Build functional, type-safe routines to interact with relational and NoSQL databases with the help of tutorials and examples provided Who This Book Is For If you are a Scala developer or data scientist, or if you want to enter the field of data science, then this book will give you all the tools you need to implement data science solutions. What You Will Learn Transform and filter tabular data to extract features for machine learning Implement your own algorithms or take advantage of MLLib's extensive suite of models to build distributed machine learning pipelines Read, transform, and write data to both SQL

and NoSQL databases in a functional manner Write robust routines to query web APIs Read data from web APIs such as the GitHub or Twitter API Use Scala to interact with MongoDB, which offers high performance and helps to store large data sets with uncertain query requirements Create Scala web applications that couple with JavaScript libraries such as D3 to create compelling interactive visualizations Deploy scalable parallel applications using Apache Spark, loading data from HDFS or Hive In Detail Scala is a multi-paradigm programming language (it supports both object-oriented and functional programming) and scripting language used to build applications for the JVM. Languages such as R, Python, Java, and so on are mostly used for data science. It is particularly good at analyzing large sets of data without any significant impact on performance and thus Scala is being adopted by many developers and data scientists. Data scientists might be aware that building applications that are truly scalable is hard. Scala, with its powerful functional libraries for interacting with databases and building scalable frameworks will give you the tools to construct robust data pipelines. This book will introduce you to the libraries for ingesting, storing, manipulating, processing, and visualizing data in Scala. Packed with real-world examples and interesting data sets, this book will teach you to ingest data from flat files and web APIs and store it in a SQL or NoSQL database. It will show you how to design scalable architectures to process and modelling your data, starting from simple concurrency constructs such as parallel collections and futures, through to actor systems and Apache Spark. As well as Scala's emphasis on functional structures and immutability, you

will learn how to use the right parallel construct for the job at hand, minimizing development time without compromising scalability. Finally, you will learn how to build beautiful interactive visualizations using web frameworks. This book gives tutorials on some of the most common Scala libraries for data science, allowing you to quickly get up to speed with building data science and data engineering solutions. Style and approach A tutorial with complete examples, this book will give you the tools to start building useful data engineering and data science solutions straightaway

Scala for the Impatient

INTRODUCTION xv CHAPTER 1: LANGUAGE FEATURES 1 Static Types and Type Inference 2 Implicit Parameters, Conversions, and Their Resolution 3 Case Class, Tuples, and Case Object 5 Abstract Class, Traits, and Sealed 6 Pattern Matching 8 Statements Are Expressions 9 String Interpolation 9 Scala Collections, immutable and mutable 10 For Comprehension 12 Packages, Companion Objects, Package Objects, and Scoping 13 AnyVal, AnyRef, Any, and the Type Hierarchy 16 Summary 17 CHAPTER 2: FUNCTIONAL PROGRAMMING 19 Immutability 20 Pure Functions 22 Recursion 23 Higher-Order Functions 26 Core Collection Methods 27 Methods Returning a Collection 29 Methods Returning a Value 31 Currying and Partially Applied Functions 32 Null Handling (Option) 34 Strict versus Non-Strict Initialization 35 Summary 36 CHAPTER 3: JAVA COMPATIBILITY 37 Scala and Java Collections 37

Online Library Functional Programming In Scala Runar Bjarnason

Interfaces and Traits 40 Scala/Java Enumerations 42 Summary 43 CHAPTER 4: SIMPLE BUILD TOOL 45 Basic Usage 46 Project Structure 47 Single Project 47 Scopes 49 Custom Tasks 50 Dependencies 50 Resolvers 51 Advanced Usage 52 Advanced Dependencies 53 Testing in the Console 55 Release Management 56 Deploying to Sonatype 56 Packaging with SBT-Native-Packager 58 Creating a Docker Image 59 Common SBT Commands 60 Useful Plugins 61 Summary 62 CHAPTER 5: MAVEN 63 Getting Started with Maven and Scala 64 Introducing scala-maven-plugin 67 Adding Library Dependencies 70 Using the REPL 71 Getting Help 72 Running Tests 72 Joint Compilation with Java 74 Accelerating Compilation with Zinc 76 Summary 77 CHAPTER 6: SCALA STYLE/LINT 79 Scala with Style 79 Scaliform 81 Scapegoat 82 WartRemover 82 Scoverage 84 Summary 84 CHAPTER 7: TESTING 85 ScalaTest 86 Unit Tests 87 Integration Testing 87 Data-Driven Tests 88 Performance Testing 89 Acceptance Testing 90 Mocks 92 Load Testing 93 Summary 94 CHAPTER 8: DOCUMENTING YOUR CODE WITH SCALADOC 95 Why Document Your Code? 96 Revealing the Benefits 96 Bookending the Continuum 96 Choosing What to Document 96 Scaladoc Structure 97 Overall Layout 97 Index Pane 98 Content Pane 100 Invoking the Scaladoc Tool 106 Wiki Syntax 108 Formatting with Inline Wiki Syntax 108 Structuring with Block Elements 110 Linking 113 Locating Scaladoc 117 Tagging 117 Everyday Tagging 117 Tagging for Groups 123 Advanced Tagging 125 Invoking scaladoc: Additional Options 132 Integrating Scaladoc Creation with Your Project 133 Configuring Maven 133 Configuring SBT 134 Publishing Scaladoc 134 Tables and CSS 136 Summary 138 CHAPTER 9: TYPE

SYSTEM 139 What Is a Type System? 140 Static versus Dynamic Typing 140 What Static Type Systems Are Good For 141 What Dynamic Type Systems Are Good For 141 Scala's Unified Type System 141 Value Classes 143 Polymorphism 145 Subtype Polymorphism 145 Parametric Polymorphism 146 Ad Hoc Polymorphism 146 Bounds 149 Context Bounds 149 Upper and Lower Bounds 150 Variance 151 Other Niceties 155 Self-Type Annotations 155 Self-Recursive Types 158 Abstract Type Members 159 Dynamic Programming 161 Structural Types 161 Dynamic Trait 162 Summary 164 CHAPTER 10: ADVANCED FUNCTIONAL PROGRAMMING 165 Higher-Kinded Types 165 Functional Design Patterns 167 Functor 167 Applicative Functor 170 Monad 172 Semigroup 173 Monoid 174 Summary 176 CHAPTER 11: CONCURRENCY 179 Synchronize/Atomic Variables 181 Future Composition 184 Parallel Collections 187 Reactive Streams 192 STM 195 Actors (Akka) 198 Spark 200 Summary 202 CHAPTER 12: SCALA.JS 205 Scala.js and Its Design 205 Getting Started: Scala.js with SBT 206 Scala.js Peculiarities 210 Webjars and Dealing with the Frontend Ecosystem 211 Summary 213 INDEX 215

Akka in Action

Summary Akka in Action is a comprehensive tutorial on building message-oriented systems using Akka. The book takes a hands-on approach, where each new concept is followed by an example that shows you how it works, how to implement the code, and how to (unit) test it. Purchase of the print book includes a free eBook

in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Akka makes it relatively easy to build applications in the cloud or on devices with many cores that efficiently use the full capacity of the computing power available. It's a toolkit that provides an actor programming model, a runtime, and required support tools for building scalable applications. About the Book Akka in Action shows you how to build message-oriented systems with Akka. This comprehensive, hands-on tutorial introduces each concept with a working example. You'll start with the big picture of how Akka works, and then quickly build and deploy a fully functional REST service out of actors. You'll explore test-driven development and deploying and scaling fault-tolerant systems. After mastering the basics, you'll discover how to model immutable messages, implement domain models, and apply techniques like event sourcing and CQRS. You'll also find a tutorial on building streaming applications using akka-stream and akka-http. Finally, you'll get practical advice on how to customize and extend your Akka system. What's Inside Getting concurrency right Testing and performance tuning Clustered and cloud-based applications Covers Akka version 2.4 About the Reader This book assumes that you're comfortable with Java and Scala. No prior experience with Akka required. About the Authors A software craftsman and architect, Raymond Roestenburg is an Akka committer. Rob Bakker specializes in concurrent back-end systems and systems integration. Rob Williams has more than 20 years of product development experience. Table of Contents Introducing Akka Up and running Test-driven development with actors Fault tolerance Futures Your first distributed Akka app

Configuration, logging, and deployment Structural patterns for actors Routing messages Message channels Finite-state machines and agents System integration Streaming Clustering Actor persistence Performance tips Looking ahead

Modern Systems Programming with Scala Native

Why learn Scala? You don't need to be a data scientist or distributed computing expert to appreciate this object-oriented functional programming language. This practical book provides a comprehensive yet approachable introduction to the language, complete with syntax diagrams, examples, and exercises. You'll start with Scala's core types and syntax before diving into higher-order functions and immutable data structures. Author Jason Swartz demonstrates why Scala's concise and expressive syntax make it an ideal language for Ruby or Python developers who want to improve their craft, while its type safety and performance ensures that it's stable and fast enough for any application. Learn about the core data types, literals, values, and variables Discover how to think and write in expressions, the foundation for Scala's syntax Write higher-order functions that accept or return other functions Become familiar with immutable data structures and easily transform them with type-safe and declarative operations Create custom infix operators to simplify existing operations or even to start your own domain-specific language Build classes that compose one or more traits for full reusability, or create new functionality by mixing them in at instantiation

Introduction to Programming and Problem-Solving Using Scala, Second Edition

Summary Play for Scala shows you how to build Scala-based web applications using the Play 2 framework. This book starts by introducing Play through a comprehensive overview example. Then, you'll look at each facet of a typical Play application both by exploring simple code snippets and by adding to a larger running example. Along the way, you'll deepen your knowledge of Scala as a programming language and work with tools like Akka. About this Book Play is a Scala web framework with built-in advantages: Scala's strong type system helps deliver bug-free code, and the Akka framework helps achieve hassle-free concurrency and peak performance. Play builds on the web's stateless nature for excellent scalability, and because it is event-based and nonblocking, you'll find it to be great for near real-time applications. Play for Scala teaches you to build Scala-based web applications using Play 2. It gets you going with a comprehensive overview example. It then explores each facet of a typical Play application by walking through sample code snippets and adding features to a running example. Along the way, you'll deepen your knowledge of Scala and learn to work with tools like Akka. Written for readers familiar with Scala and web-based application architectures. No knowledge of Play is assumed. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications.

What's Inside Intro to Play 2 Play's MVC structure Mastering Scala templates and forms Persisting data and using web services Using Play's advanced features About the Authors Peter Hiltonv, Erik Bakker, and Francisco Canedo, are engineers at Lunatech, a consultancy with Scala and Play expertise. They are contributors to the Play framework. Table of Contents PART 1: GETTING STARTED Introduction to Play Your first Play application PART 2: CORE FUNCTIONALITY Deconstructing Play application architecture Defining the application's HTTP interface Storing data—the persistence layer Building a user interface with view templates Validating and processing input with the forms API PART 3: ADVANCED CONCEPTS Building a single-page JavaScript application with JSON Play and more Web services, iterates, and WebSockets

Haskell

This is the Scala edition of Category Theory for Programmers by Bartosz Milewski. This book contains code snippets in both Haskell and Scala.

Programming in Scala

Scala is a modern programming language for the Java Virtual Machine (JVM) that combines the best features of object-oriented and functional programming

languages. Using Scala, you can write programs more concisely than in Java, as well as leverage the full power of concurrency. Since Scala runs on the JVM, it can access any Java library and is interoperable with Java frameworks. Scala for the Impatient concisely shows developers what Scala can do and how to do it. In this book, Cay Horstmann, the principal author of the international best-selling Core Java™, offers a rapid, code-based introduction that's completely practical. Horstmann introduces Scala concepts and techniques in “blog-sized” chunks that you can quickly master and apply. Hands-on activities guide you through well-defined stages of competency, from basic to expert. Coverage includes Getting started quickly with Scala's interpreter, syntax, tools, and unique idioms Mastering core language features: functions, arrays, maps, tuples, packages, imports, exception handling, and more Becoming familiar with object-oriented programming in Scala: classes, inheritance, and traits Using Scala for real-world programming tasks: working with files, regular expressions, and XML Working with higher-order functions and the powerful Scala collections library Leveraging Scala's powerful pattern matching and case classes Creating concurrent programs with Scala actors Implementing domain-specific languages Understanding the Scala type system Applying advanced “power tools” such as annotations, implicits, and delimited continuations Scala is rapidly reaching a tipping point that will reshape the experience of programming. This book will help object-oriented programmers build on their existing skills, allowing them to immediately construct useful applications as they gradually master advanced programming techniques.

Play for Scala

Presents an introduction to the new programming language for the Java Platform.

Category Theory for Programmers (Scala Edition, Paperback)

Save time and trouble when using Scala to build object-oriented, functional, and concurrent applications. With more than 250 ready-to-use recipes and 700 code examples, this comprehensive cookbook covers the most common problems you'll encounter when using the Scala language, libraries, and tools. It's ideal not only for experienced Scala developers, but also for programmers learning to use this JVM language. Author Alvin Alexander (creator of DevDaily.com) provides solutions based on his experience using Scala for highly scalable, component-based applications that support concurrency and distribution. Packed with real-world scenarios, this book provides recipes for: Strings, numeric types, and control structures Classes, methods, objects, traits, and packaging Functional programming in a variety of situations Collections covering Scala's wealth of classes and methods Concurrency, using the Akka Actors library Using the Scala REPL and the Simple Build Tool (SBT) Web services on both the client and server sides Interacting with SQL and NoSQL databases Best practices in Scala development

Scala Design Patterns

If you build your Scala application through Test-Driven Development, you'll quickly see the advantages of testing before you write production code. This hands-on book shows you how to create tests with ScalaTest and the Specs2—two of the best testing frameworks available—and how to run your tests in the Simple Build Tool (SBT) designed specifically for Scala projects. By building a sample digital jukebox application, you'll discover how to isolate your tests from large subsystems and networks with mocking code, and how to use the ScalaCheck library for automated specification-based testing. If you're familiar with Scala, Ruby, or Python, this book is for you. Get an overview of Test-Driven Development Start a simple project with SBT and create tests before you write code Dive into SBT's basic commands, interactive mode, packaging, and history Use ScalaTest both in the command line and with SBT, and learn how to incorporate JUnit and TestNG Work with the Specs2 framework, including Specification styles, matchers DSLs, and Data Tables Understand mocking by using Java frameworks EasyMock and Mockito, and the Scala-only framework ScalaMock Automate testing by using ScalaCheck to generate fake data

Scala in Action

Online Library Functional Programming In Scala Runar Bjarnason

Access the power of bare-metal systems programming with Scala Native, an ahead-of-time Scala compiler. Without the baggage of legacy frameworks and virtual machines, Scala Native lets you re-imagine how your programs interact with your operating system. Compile Scala code down to native machine instructions; seamlessly invoke operating system APIs for low-level networking and IO; control pointers, arrays, and other memory management techniques for extreme performance; and enjoy instant start-up times. Skip the JVM and improve your code performance by getting close to the metal. Developers generally build systems on top of the work of those who came before, accumulating layer upon layer of abstraction. Scala Native provides a rare opportunity to remove layers. Without the JVM, Scala Native uses POSIX and ANSI C APIs to build concise, expressive programs that run unusually close to bare metal. Scala Native compiles Scala code down to native machine instructions instead of JVM bytecode. It starts up fast, without the sluggish warm-up phase that's common for just-in-time compilers. Scala Native programs can seamlessly invoke operating system APIs for low-level networking and IO. And Scala Native lets you control pointers, arrays, and other memory layout types for extreme performance. Write practical, bare-metal code with Scala Native, step by step. Understand the foundations of systems programming, including pointers, arrays, strings, and memory management. Use the UNIX socket API to write network client and server programs without the sort of frameworks higher-level languages rely on. Put all the pieces together to design and implement a modern, asynchronous microservice-style HTTP framework from

scratch. Take advantage of Scala Native's clean, modern syntax to write lean, high-performance code without the JVM. What You Need: A modern Windows, Mac OS, or Linux system capable of running Docker. All code examples in the book are designed to run on a portable Docker-based build environment that runs anywhere. If you don't have Docker yet, see the Appendix for instructions on how to get it.

Professional Scala

Functional languages help developers support concurrency by encouraging immutable data structures that can be passed between threads without having to worry about a shared state, all while avoiding side effects. Concurrency in .NET teaches readers how to build concurrent and scalable programs in .NET using the functional paradigm. This intermediate-level guide is aimed at developers, architects, and passionate computer programmers. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications.

Stream Processing with Apache Flink

Get started with Apache Flink, the open source framework that powers some of the world's largest stream processing applications. With this practical book, you'll

Online Library Functional Programming In Scala Runar Bjarnason

explore the fundamental concepts of parallel stream processing and discover how this technology differs from traditional batch data processing. Longtime Apache Flink committers Fabian Hueske and Vasia Kalavri show you how to implement scalable streaming applications with Flink's DataStream API and continuously run and maintain these applications in operational environments. Stream processing is ideal for many use cases, including low-latency ETL, streaming analytics, and real-time dashboards as well as fraud detection, anomaly detection, and alerting. You can process continuous data of any kind, including user interactions, financial transactions, and IoT data, as soon as you generate them. Learn concepts and challenges of distributed stateful stream processing Explore Flink's system architecture, including its event-time processing mode and fault-tolerance model Understand the fundamentals and building blocks of the DataStream API, including its time-based and stateful operators Read data from and write data to external systems with exactly-once consistency Deploy and configure Flink clusters Operate continuously running streaming applications

Scala Cookbook

Get up to speed on Scala, the JVM language that offers all the benefits of a modern object model, functional programming, and an advanced type system. Packed with code examples, this comprehensive book shows you how to be productive with the language and ecosystem right away, and explains why Scala is ideal for today's

highly scalable, data-centric applications that support concurrency and distribution. This second edition covers recent language features, with new chapters on pattern matching, comprehensions, and advanced functional programming. You'll also learn about Scala's command-line tools, third-party tools, libraries, and language-aware plugins for editors and IDEs. This book is ideal for beginning and advanced Scala developers alike. Program faster with Scala's succinct and flexible syntax Dive into basic and advanced functional programming (FP) techniques Build killer big-data apps, using Scala's functional combinators Use traits for mixin composition and pattern matching for data extraction Learn the sophisticated type system that combines FP and object-oriented programming concepts Explore Scala-specific concurrency tools, including Akka Understand how to develop rich domain-specific languages Learn good design techniques for building scalable and robust Scala applications

Thinking Functionally with Haskell

Types are often seen as a tool for checking errors, with the programmer writing a complete program first and using the type checker to detect errors. And while tests are used to show presence of errors, they can only find errors that you explicitly test for. In type-driven development, types become your tools for constructing programs and, used appropriately, can show the absence of errors. And you can express precise relationships between data, your assumptions are explicit and

checkable, and you can precisely state and verify properties. Type-driven development lets users write extensible code, create simple specifications very early in development, and easily create mock implementation for testing. Type-Driven Development with Idris, written by the creator of Idris, teaches programmers how to improve the performance and accuracy of programs by taking advantage of a state-of-the-art type system. This book teaches readers using Idris, a language designed from the very beginning to support type-driven development. Readers learn how to manipulate types just like any other construct (numbers, strings, lists, etc.). This book teaches how to use type-driven development to build real-world software, as well as how to handle side-effects, state and concurrency, and interoperating with existing systems. By the end of this book, readers will be able to develop robust and verified software in Idris and apply type-driven development methods to programming in other languages. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications.

Reactive Design Patterns

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.

The Go Programming Language

Modern distributed applications must deliver near-realtime performance while simultaneously managing big data and high user loads spread across environments ranging from cloud systems to mobile devices. Unlike traditional enterprise applications which focus on decoupling their internal components by defining programming interfaces, reactive applications go one step further and decouple

their components also at runtime. This makes it possible to react effectively and efficiently to failures, varying user demands, and changes in the application's execution environment. The resulting systems are highly concurrent and fault-tolerant, with minimal dependencies among individual system components. Reactive Design Patterns is a clearly-written guide for building message-driven distributed systems that are resilient, responsive, and elastic. It contains patterns for messaging, flow control, resource management, and concurrency, along with practical issues like test-friendly designs. All patterns include concrete examples using Scala and Akka—in some cases, Java, JavaScript, and Erlang. Software engineers and architects will learn patterns that address day-to-day distributed development problems in a fault-tolerant and scalable way. Project leaders and CTOs will gain a deeper understanding of the reactive design philosophy. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications.

An Invitation to Applied Category Theory

The Go Programming Language is the authoritative resource for any programmer who wants to learn Go. It shows how to write clear and idiomatic Go to solve real-world problems. The book does not assume prior knowledge of Go nor experience with any specific language, so you'll find it accessible whether you're most comfortable with JavaScript, Ruby, Python, Java, or C++. The first chapter is a

tutorial on the basic concepts of Go, introduced through programs for file I/O and text processing, simple graphics, and web clients and servers. Early chapters cover the structural elements of Go programs: syntax, control flow, data types, and the organization of a program into packages, files, and functions. The examples illustrate many packages from the standard library and show how to create new ones of your own. Later chapters explain the package mechanism in more detail, and how to build, test, and maintain projects using the go tool. The chapters on methods and interfaces introduce Go's unconventional approach to object-oriented programming, in which methods can be declared on any type and interfaces are implicitly satisfied. They explain the key principles of encapsulation, composition, and substitutability using realistic examples. Two chapters on concurrency present in-depth approaches to this increasingly important topic. The first, which covers the basic mechanisms of goroutines and channels, illustrates the style known as communicating sequential processes for which Go is renowned. The second covers more traditional aspects of concurrency with shared variables. These chapters provide a solid foundation for programmers encountering concurrency for the first time. The final two chapters explore lower-level features of Go. One covers the art of metaprogramming using reflection. The other shows how to use the unsafe package to step outside the type system for special situations, and how to use the cgo tool to create Go bindings for C libraries. The book features hundreds of interesting and practical examples of well-written Go code that cover the whole language, its most important packages, and a wide range of applications. Each

chapter has exercises to test your understanding and explore extensions and alternatives. Source code is freely available for download from <http://gopl.io/> and may be conveniently fetched, built, and installed using the `go get` command.

Modern Java in Action

"Modern applications take advantage of innovative designs, including microservices, reactive architectures, and streaming data. Modern Java features like lambdas, streams, and the long-awaited Java Module System make implementing these designs significantly easier. It's time to upgrade your skills and meet these challenges head on! Modern Java in action connects new features of the Java language with their practical applications. Using crystal-clear examples and careful attention to detail, this book respects your time. It will help you expand your existing knowledge of core Java as you master modern additions like the Streams API and the Java Module System, explore new approaches to concurrency, and learn how functional concepts can help you write code that's easier to read and maintain."--Page 4 de la couverture.

Scala Puzzlers

As a software engineer, you recognize at some point that there's much more to

your career than dealing with code. Is it time to become a manager? Tell your boss he's a jerk? Join that startup? Author Michael Lopp recalls his own make-or-break moments with Silicon Valley giants such as Apple, Netscape, and Symantec in *Being Geek* -- an insightful and entertaining book that will help you make better career decisions. With more than 40 standalone stories, Lopp walks through a complete job life cycle, starting with the job interview and ending with the realization that it might be time to find another gig. Many books teach you how to interview for a job or how to manage a project successfully, but only this book helps you handle the baffling circumstances you may encounter throughout your career. Decide what you're worth with the chapter on "The Business" Determine the nature of the miracle your CEO wants with "The Impossible" Give effective presentations with "How Not to Throw Up" Handle liars and people with devious agendas with "Managing Werewolves" Realize when you should be looking for a new gig with "The Itch"

Functional Programming in Scala

Introduces fundamental techniques for reasoning mathematically about functional programs. Ideal for a first- or second-year undergraduate course.

Functional Concurrency in . Net

Online Library Functional Programming In Scala Runar Bjarnason

Offers a tutorial to the Scala programming language, describing how to use the open source libraries for both Java and Scala, how to build DSLs and other productivity tools, and ways to debug and test using ScalaTest.

Functional Programming in Java

This full-color booklet contains chapter notes, hints, solutions to exercises, addenda, and errata for the book "Functional Programming in Scala" by Paul Chiusano and Runar Bjarnason. This material is freely available online, but is compiled here as a convenient companion to the book itself. All code is colorfully syntax-highlighted.

Learning Scala

Practical FP in Scala: A hands-on approach, is a book for intermediate to advanced Scala developers. Aimed at those who understand functional effects, referential transparency and the benefits of functional programming to some extent but who are missing some pieces to put all these concepts together to build a large application in a time-constrained manner. Throughout the chapters we will design, architect and develop a complete stateful application serving an API via HTTP, accessing a database and dealing with cached data, using the best practices and

best functional libraries available in the Cats ecosystem. You will also learn about common design patterns such as managing state, error handling and anti-patterns, all accompanied by clear examples. Furthermore, at the end of the book, we will dive into some advanced concepts such as MTL, Classy Optics and Typeclass derivation.

Scala in Depth

Helps programmers learn functional programming and apply it to the everyday business of coding. Original.

Programming Scala

Functional and Reactive Domain Modeling teaches readers how to think of the domain model in terms of pure functions and how to compose them to build larger abstractions. It begins with the basics of functional programming and gradually progresses to the advanced concepts and patterns needed to implement complex domain models. The book demonstrates how advanced FP patterns like algebraic data types, typeclass based design, and isolation of side-effects can make models compose for readability and verifiability. On the subject of reactive modeling, the book focuses on higher order concurrency patterns like actors and futures. It uses

the Akka framework as the reference implementation and demonstrates how advanced architectural patterns like event sourcing and CQRS can be put to great use in implementing scalable models. It offers techniques that are radically different from the standard RDBMS based applications that are based on mutation of records. It also shares important patterns like using asynchronous messaging for interaction based on non blocking concurrency and model persistence, which delivers the speed of in- memory processing along with suitable guarantees of reliability.

Acoustic Emission Testing

Category theory is unmatched in its ability to organize and layer abstractions and to find commonalities between structures of all sorts. No longer the exclusive preserve of pure mathematicians, it is now proving itself to be a powerful tool in science, informatics, and industry. By facilitating communication between communities and building rigorous bridges between disparate worlds, applied category theory has the potential to be a major organizing force. This book offers a self-contained tour of applied category theory. Each chapter follows a single thread motivated by a real-world application and discussed with category-theoretic tools. We see data migration as an adjoint functor, electrical circuits in terms of monoidal categories and operads, and collaborative design via enriched profunctors. All the relevant category theory, from simple to sophisticated, is introduced in an

accessible way with many examples and exercises, making this an ideal guide even for those without experience of university-level mathematics.

Practical FP in Scala (hard-Cover)

Presents an introduction to the Scala programming language which is an abbreviated version of object-orientated programming combined with the power of concurrency capable of running on the Java Virtual Machine.

The Little Typer

Summary Functional Programming in Java teaches Java developers how to incorporate the most powerful benefits of functional programming into new and existing Java code. You'll learn to think functionally about coding tasks in Java and use FP to make your applications easier to understand, optimize, maintain, and scale. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Here's a bold statement: learn functional programming and you'll be a better Java developer. Fortunately, you don't have to master every aspect of FP to get a big payoff. If you take in a few core principles, you'll see an immediate boost in the scalability, readability, and maintainability of your code. And did we mention that you'll have fewer bugs? Let's

get started! About the Book Functional Programming in Java teaches you how to incorporate the powerful benefits of functional programming into new and existing Java code. This book uses easy-to-grasp examples, exercises, and illustrations to teach core FP principles such as referential transparency, immutability, persistence, and laziness. Along the way, you'll discover which of the new functionally inspired features of Java 8 will help you most. What's Inside Writing code that's easier to read and reason about Safer concurrent and parallel programming Handling errors without exceptions Java 8 features like lambdas, method references, and functional interfaces About the Reader Written for Java developers with no previous FP experience. About the Author Pierre-Yves Saumont is a seasoned Java developer with three decades of experience designing and building enterprise software. He is an R&D engineer at Alcatel-Lucent Submarine Networks. Table of Contents What is functional programming? Using functions in Java Making Java more functional Recursion, corecursion, and memoization Data handling with lists Dealing with optional data Handling errors and exceptions Advanced list handling Working with laziness More data handling with trees Solving real problems with advanced trees Handling state mutation in a functional way Functional input/output Sharing mutable state with actors Solving common problems functionally

Type-Driven Development with Idris

With its flexibility for programming both small and large projects, Scala is an ideal language for teaching beginning programming. Yet there are no textbooks on Scala currently available for the CS1/CS2 levels. Introduction to the Art of Programming Using Scala presents many concepts from CS1 and CS2 using a modern, JVM-based language that works we

Scala Microservices

If you've had trouble trying to learn Functional Programming (FP), you're not alone. In this book, Alvin Alexander -- author of the Scala Cookbook and former teacher of Java and Object-Oriented Programming (OOP) classes -- writes about his own problems in trying to understand FP, and how he finally conquered it. What he originally learned is that experienced FP developers are driven by two goals: to use only immutable values, and write only pure functions. What he later learned is that they have these goals as the result of another larger goal: they want all of their code to look and work just like algebra. While that sounds simple, it turns out that these goals require them to use many advanced Scala features -- which they often use all at the same time. As a result, their code can look completely foreign to novice FP developers. As Mr. Alexander writes, "When you first see their code it's easy to ask, 'Why would anyone write code like this?'" Mr. Alexander answers that "Why?" question by explaining the benefits of writing pure functional code. Once you understand those benefits -- your motivation for learning FP -- he shares five

Online Library Functional Programming In Scala Runar Bjarnason

rules for programming in the book: All fields must be immutable ('val' fields). All functions must be pure functions. Null values are not allowed. Whenever you use an 'if' you must also use an 'else'. You won't create OOP classes that encapsulate data and behavior; instead you'll design data structures using Scala 'case' classes, and write pure functions that operate on those data structures. In the book you'll see how those five, simple rules naturally lead you to write pure, functional code that reads like algebra. He also shares one more Golden Rule for learning: Always ask "Why"? Lessons in the book include: How and why to write only pure functions Why pure function signatures are much more important than OOP method signatures Why recursion is a natural tool for functional programming, and how to write recursive algorithms Because the Scala 'for' expression is so important to FP, dozens of pages explain the details of how it works In the end you'll see that monads aren't that difficult because they're a natural extension of the Five Rules The book finishes with lessons on FP data modeling, and two main approaches for organizing your pure functions As Mr. Alexander writes, "In this book I take the time to explain all of the concepts that are used to write FP code in Scala. As I learned from my own experience, once you understand the Five Rules and the small concepts, you can understand Scala/FP." Please note that because of the limits on how large a printed book can be, the paperback version does not include all of the chapters that are in the Kindle eBook. The following lessons are not in the paperback version: Grandma's Cookies (a story about pure functions) The ScalaCheck lessons The Type Classes lessons The appendices Because those

lessons didn't fit in the print version, they have been made freely available online. (Alvin Alexander (alvinalexander.com) wrote the popular Scala Cookbook for O'Reilly, and also self-published two other books, How I Sold My Business: A Personal Diary, and A Survival Guide for New Consultants.)

Spark: The Definitive Guide

Distributed systems have helped application development teams deal with failures, downtime, and poor scaling, but these systems bring technical challenges of their own. With this unique cookbook, system architects will get a detailed understanding of reactive systems, along with proven recipes for dealing with different architectural issues. Each self-contained chapter covers the architecture of an entire reactive system, and--since these systems share many of the same architectural issues--each chapter also focuses on a particular area, such as delivery semantics or monitoring & tracing, with detailed solutions for problems that commonly arise. Learn the architecture and implementation tips for an entire reactive microservices-based system in each chapter Understand the challenges of long-term running and evolution of your distributed system Explore different failure modes of distributed systems and the approaches to address them Learn about proper site reliability and production readiness

Testing in Scala

Acoustic Emission (AE) techniques have been studied in civil engineering for a long time. The techniques are recently going to be more and more applied to practical applications and to be standardized in the codes. This is because the increase of aging structures and disastrous damages due to recent earthquakes urgently demand for maintenance and retrofit of civil structures in service for example. It results in the need for the development of advanced and effective inspection techniques. Thus, AE techniques draw a great attention to diagnostic applications and in material testing. The book covers all levels from the description of AE basics for AE beginners (level of a student) to sophisticated AE algorithms and applications to real large-scale structures as well as the observation of the cracking process in laboratory specimen to study fracture processes.

Programming Language Concepts

Learn how to use, deploy, and maintain Apache Spark with this comprehensive guide, written by the creators of the open-source cluster-computing framework. With an emphasis on improvements and new features in Spark 2.0, authors Bill Chambers and Matei Zaharia break down Spark topics into distinct sections, each with unique goals. You'll explore the basic operations and common functions of

Online Library Functional Programming In Scala Runar Bjarnason

Spark's structured APIs, as well as Structured Streaming, a new high-level API for building end-to-end streaming applications. Developers and system administrators will learn the fundamentals of monitoring, tuning, and debugging Spark, and explore machine learning techniques and scenarios for employing MLlib, Spark's scalable machine-learning library. Get a gentle overview of big data and Spark Learn about DataFrames, SQL, and Datasets—Spark's core APIs—through worked examples Dive into Spark's low-level APIs, RDDs, and execution of SQL and DataFrames Understand how Spark runs on a cluster Debug, monitor, and tune Spark clusters and applications Learn the power of Structured Streaming, Spark's stream-processing engine Learn how you can apply MLlib to a variety of problems, including classification or recommendation

[ROMANCE](#) [ACTION & ADVENTURE](#) [MYSTERY & THRILLER](#) [BIOGRAPHIES & HISTORY](#) [CHILDREN'S](#) [YOUNG ADULT](#) [FANTASY](#) [HISTORICAL FICTION](#) [HORROR](#) [LITERARY FICTION](#) [NON-FICTION](#) [SCIENCE FICTION](#)