Type Driven Development With Idris

The harrowing, utterly original debut novel by Uzodinma Iweala about the life of a child soldier in a war-torn African country—now a critically-acclaimed Netflix original film directed by Cary Fukunaga (True Detective) and starring Idris Elba (Mandela, The Wire). As civil war rages in an unnamed West-African nation, Agu, the school-aged protagonist of this stunning debut novel, is recruited into a unit of guerilla fighters. Haunted by his father's own death at the hands of militants, which he fled just before witnessing, Agu is vulnerable to the dangerous yet paternal nature of his new commander. While the war rages on, Agu becomes increasingly divorced from the life he had known before the conflict started—a life of school friends, church services, and time with his family, still intact. As he vividly recalls these sunnier times, his daily reality continues to spin further downward into inexplicable brutality, primal fear, and loss of selfhood. In a powerful, strikingly original voice, Uzodinma Iweala leads the reader through the random travels, betrayals, and violence that mark Agu's new community. Electrifying and engrossing, Beasts of No Nation announces the arrival of an extraordinary new writer.

Summary Rx.NET in Action teaches developers how to build event-driven applications using the Reactive Extensions (Rx) library. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Modern applications must react to streams of data such as user and system events, internal messages, and sensor input. Reactive Extensions (Rx) is a .NET library containing more than 600 operators that you can compose together to build reactive client- and server-side applications to handle events asynchronously in a way that maximizes responsiveness, resiliency, and elasticity. About the Book Rx.NET in Action teaches developers how to build event-driven applications using the Rx library. Starting with an overview of the design and architecture of Rx-based reactive applications, you'll get hands-on with in-depth code examples to discover firsthand how to exploit the rich query capabilities that Rx provides and the Rx concurrency model that allows you to control both the asynchronicity of your code and the processing of event handlers. You'll also learn about consuming event streams, using schedulers to manage time, and working with Rx operators to filter, transform, and group events. What's Inside Introduction to Rx in C# Creating and consuming streams of data and events Building complex queries on event streams Error handling and testing Rx code About the Reader Readers should understand OOP concepts and be comfortable coding in C#. About the Author Tamir Dresher is a senior software architect at CodeValue and a prominent member of Israel's Microsoft programming community. Table of Contents PART 1 - GETTING STARTED WITH REACTIVE EXTENSIONS Reactive programming Hello, Rx Functional thinking in C# PART 2 - CORE IDEAS Creating observable sequences Creating observables from .NET asynchronous types Controlling the observer-observable relationship Controlling the observable temperature Working with basic query operators Partitioning and combining observables Working with Rx concurrency and synchronization Error handling and recovery APPENDIXES Writing asynchronous code in .NET The Rx Disposables library Testing Rx queries and operators

A comprehensive playbook for applied design thinking inbusiness and management,

complete with concepts andtoolkits As many companies have lost confidence in the traditional waysof running a business, design thinking has entered the mix.Design Thinking for Strategic Innovation presents aframework for design thinking that is relevant to businessmanagement, marketing, and design strategies and also provides atoolkit to apply concepts for immediate use in everyday work. Itexplains how design thinking can bring about creative solutions tosolve complex business problems. Organized into five sections, thisbook provides an introduction to the values and applications ofdesign thinking, explains design thinking approaches for eight keychallenges that most businesses face, and offers an applicationframework for these business challenges through exercises,activities, and resources. An essential guide for any business seeking to use designthinking as a problem-solving tool as well as a business method totransform companies and cultures The framework is based on work developed by the author for anexecutive program in Design Thinking taught in Harvard GraduateSchool of Design Author Idris Mootee is a management guru and a leading experton applied design thinking Revolutionize your approach to solving your business's greatestchallenges through the power of Design Thinking for StrategicInnovation.

Gilmore Girls meets vibrant New Delhi in this thoughtful and hilarious new novel about a teen facing family expectations, relationship complications, and hidden secrets in a new country—sprinkled with Sheba Karim's signature wit and steamy romance, and perfect for readers who loved Mary H. K. Choi's Emergency Contact and Adib Khorram's Darius the Great Is Not Okay. To cure her post-senior year slump, made worse by the loss of her aunt Sonia, Noreen decides to follow her mom on a gap year trip to New Delhi, hoping India can lessen her grief and bring her voice back. In the world's most polluted city, Noreen soon meets kind, handsome Kabir, who introduces her to the wonders of this magical, complicated place. With the help of Kabir—plus Bollywood celebrities, fourteenth-century ruins, karaoke parties, and Sufi saints—Noreen discovers new meanings for home. But when a family scandal erupts, Noreen and Kabir must face complex questions in their own relationship: What does it mean to truly stand by someone—and what are the boundaries of love?

Summary Kotlin in Action guides experienced Java developers from the language basics of Kotlin all the way through building applications to run on the JVM and Android devices. Foreword by Andrey Breslav, Lead Designer of Kotlin. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Developers want to get work done - and the less hassle, the better. Coding with Kotlin means less hassle. The Kotlin programming language offers an expressive syntax, a strong intuitive type system, and great tooling support along with seamless interoperability with existing Java code, libraries, and frameworks. Kotlin can be compiled to Java bytecode, so you can use it everywhere Java is used, including Android. And with an effi cient compiler and a small standard library, Kotlin imposes virtually no runtime overhead. About the Book Kotlin in Action teaches you to use the Kotlin language for production-quality applications. Written for experienced Java developers, this example-rich book goes further than most language books, covering interesting topics like building DSLs with natural language syntax. The authors are core Kotlin developers, so you can trust that even the gnarly details are dead accurate. What's Inside Functional programming on the JVM Writing clean and

idiomatic code Combining Kotlin and Java Domain-specific languages About the Reader This book is for experienced Java developers. About the Author Dmitry Jemerov and Svetlana Isakova are core Kotlin developers at JetBrains. Table of Contents PART 1 - INTRODUCING KOTLIN Kotlin: what and why Kotlin basics Defining and calling functions Classes, objects, and interfaces Programming with lambdas The Kotlin type system PART 2 - EMBRACING KOTLIN Operator overloading and other conventions Higher-order functions: lambdas as parameters and return values Generics Annotations and reflection DSL construction

It's an exciting time to get involved with MicroPython, the re-implementation of Python 3 for microcontrollers and embedded systems. This practical guide delivers the knowledge you need to roll up your sleeves and create exceptional embedded projects with this lean and efficient programming language. If you're familiar with Python as a programmer, educator, or maker, you're ready to learn—and have fun along the way. Author Nicholas Tollervey takes you on a journey from first steps to advanced projects. You'll explore the types of devices that run MicroPython, and examine how the language uses and interacts with hardware to process input, connect to the outside world, communicate wirelessly, make sounds and music, and drive robotics projects. Work with MicroPython on four typical devices: PyBoard, the micro:bit, Adafruit's Circuit Playground Express, and ESP8266/ESP32 boards Explore a framework that helps you generate, evaluate, and evolve embedded projects that solve real problems Dive into practical MicroPython examples: visual feedback, input and sensing, GPIO, networking, sound and music, and robotics Learn how idiomatic MicroPython helps you express a lot with the minimum of resources Take the next step by getting involved with the Python community

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.

Just as Jack Kerouac captured the beat of the '50s, his daughter captured the rhythm of the generation that followed. With a graceful, often disturbing detachment and a

spellbinding gift for descriptive imagery, Jan Kerouac explores the tortured, freewheeling soul of a woman on her own road. From an adolescence of LSD, detention homes, probation, pregnancy, and a stillbirth in the Mexican tropics at age 15; to the peace movement in Haight-Ashbury and Washington state; to traveling by bus through Central America with a madman for a lover, Baby Driver moves with the force of a tropical storm.

Age range 14+ Meet Tariq Nader, leader of 'The Wolf Pack' at Punchbowl High, who has been commanded by the new principal to join a football competition with his mates in order to rehabilitate the public image of their school. When the team is formed, Tariq learns there's a major catch – half of the team is made up of white boys from Cronulla, aka enemy territory – and he must compete with their strongest player for captaincy of the team. At school Tariq thinks he has life all figured out until he falls for a new girl called Jamila, who challenges everything he thought he knew. At home, his outspoken ways have brought him into conflict with his family. Now, with complications on all fronts, he has to dig deep to control his anger, and find what it takes to be a leader. In confronting and often hilarious situations, Tariq's relationships with his extended Lebanese family and his friends are tested like never before, and he comes to learn that his choices can have serious consequences. 'Rawah Arja has written one of the best Australian novels I have ever read…not just because the writing sparkles even as it packs a punch, but because this is the kind of novel that shows us what Australia really is.' — Randa Abdel-Fattah 'A great novel, funny and authentic and full of characters I loved.' — Melina Marchetta, author of Looking for Alibrandi, Saving Francesca and On the Jellicoe Road 'Rawah Arja is an astounding new talent whose work will speak to young people everywhere. The F Team is a fresh, surprising work that is set to become a young adult classic.' — Felicity Castagna, author of No More Boats 'In her debut YA novel, Rawah Arja expertly expresses the difficulty of transitioning from adolescence to adulthood while navigating family, friendship, love and self-worth. Not just full of heart and the importance of community, Arja's novel is also honest...The F Team is an introspective and thought-provoking coming-of-age read.' — Books+Publishing Summary Reactive Design Patterns is a clearly written guide for building message-driven distributed systems that are resilient, responsive, and elastic. In this book you'll find 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. Foreword by Jonas Bonér. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Modern web applications serve potentially vast numbers of users - and they need to keep working as servers fail and new ones come online, users overwhelm limited resources, and information is distributed globally. A Reactive application adjusts to partial failures and varying loads, remaining responsive in an ever-changing distributed environment. The secret is message-driven architecture - and design patterns to organize it. About the Book Reactive Design Patterns presents the principles, patterns, and best practices of Reactive application design. You'll learn how to keep one slow component from bogging down others with the Circuit Breaker pattern, how to shepherd a many-staged transaction to completion with the Saga pattern, how to divide datasets by Sharding, and more. You'll even see how to keep your source code readable and the system testable despite many potential interactions

and points of failure. What's Inside The definitive guide to the Reactive Manifesto Patterns for flow control, delimited consistency, fault tolerance, and much more Hard-won lessons about what doesn't work Architectures that scale under tremendous load About the Reader Most examples use Scala, Java, and Akka. Readers should be familiar with distributed systems. About the Author Dr. Roland Kuhn led the Akka team at Lightbend and coauthored the Reactive Manifesto. Brian Hanafee and Jamie Allen are experienced distributed systems architects. Table of Contents PART 1 - INTRODUCTION Why Reactive? A walk-through of the Reactive Manifesto Tools of the trade PART 2 - THE PHILOSOPHY IN A NUTSHELL Message passing Location transparency Divide and conquer Principled failure handling Delimited consistency Nondeterminism by need Message flow PART 3 - PATTERNS Testing reactive applications Fault tolerance and recovery patterns Replication patterns Resource-management patterns Message flow patterns Flow control patterns State management and persistence patterns

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 ?uki? 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

Type-Driven Development with IdrisManning Publications

An introduction to writing proofs about computer programs, written in an accessible question-and-answer style, complete with step-by-step examples and a simple proof assistant. The Little Prover introduces inductive proofs as a way to determine facts about computer programs. It is written in an approachable, engaging style of question-and-answer, with the characteristic humor of The Little Schemer (fourth edition, MIT

Press). Sometimes the best way to learn something is to sit down and do it; the book takes readers through step-by-step examples showing how to write inductive proofs. The Little Prover assumes only knowledge of recursive programs and lists (as presented in the first three chapters of The Little Schemer) and uses only a few terms beyond what novice programmers already know. The book comes with a simple proof assistant to help readers work through the book and complete solutions to every example.

A project-filled introduction to coding that shows kids how to build programs by making cool games. Scratch, the colorful drag-and-drop programming language, is used by millions of first-time learners worldwide. Scratch 3 features an updated interface, new programming blocks, and the ability to run on tablets and smartphones, so you can learn how to code on the go. In Scratch 3 Programming Playground, you'll learn to code by making cool games. Get ready to destroy asteroids, shoot hoops, and slice and dice fruit! Each game includes easy-to-follow instructions with full-color images, review questions, and creative coding challenges to make the game your own. Want to add more levels or a cheat code? No problem, just write some code. You'll learn to make games like: • Maze Runner: escape the maze! • Snaaaaaake: gobble apples and avoid your own tail • Asteroid Breaker: smash space rocks • Fruit Slicer: a Fruit Ninja clone • Brick Breaker: a remake of Breakout, the brick-breaking classic • Platformer: a game inspired by Super Mario Bros Learning how to program shouldn't be dry and dreary. With Scratch 3 Programming Playground, you'll make a game of it! Covers: Scratch 3 Summary Get Programming with F#: A guide for .NET developers teaches F# through 43 example-based lessons with built-in exercises so you can learn the only way that really works: by practicing. The book upgrades your .NET skills with a touch of functional programming in F#. You'll pick up core FP principles and learn techniques for iron-clad reliability and crystal clarity. You'll discover productivity techniques for coding F# in Visual Studio, functional design, and integrating functional and OO code. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Your .NET applications need to be good for the long haul. F#'s unique blend of functional and imperative programming is perfect for writing code that performs flawlessly now and keeps running as your needs grow and change. It takes a little practice to master F#'s functional-first style, so you may as well get programming! What's Inside Learn how to write bug-free programs Turn tedious common tasks into quick and easy ones Use minimal code to work with JSON, CSV, XML, and HTML data Integrate F# with your existing C# and VB.NET applications Create web-enabled applications About the Reader Written for intermediate C# and Visual Basic .NET developers. No experience with F# is assumed. Table of Contents Unit 1 - F# AND VISUAL STUDIO Lesson 1 - The Visual Studio experience Lesson 2 - Creating your first F# program Lesson 3 - The REPL-changing how we develop Unit 2 - HELLO F# Lesson 4 - Saying a little, doing a lot Lesson 5 - Trusting the compiler Lesson 6 - Working with immutable data Lesson 7 - Expressions and statements Lesson 8 Capstone 1 Unit 3 - TYPES AND FUNCTIONS Lesson 9 - Shaping data with tuples Lesson 10 - Shaping data with records Lesson 11 - Building composable functions Lesson 12 - Organizing code without classes Lesson 13 - Achieving code reuse in F# Lesson 14 - Capstone 2 Unit 4 - COLLECTIONS IN F# Lesson 15 - Working with collections in F# Lesson 16 - Useful collection functions

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.

Dependent types are a powerful concept that allow us to write proof-carrying code. Idris is a programming language that supports dependent types. We will learn about the mathematical foundations, and then write correct software and mathematically prove properties about it.This book aims to be accessible to novices, and no prior experience beyond high school mathematics is needed. Thus, this book is written in a way to be self-contained.The first part of this book serves as an introduction to the theory behind Idris, while the second part is a practical introduction to Idris with examples.

Zelie Adebola remembers when the soil of Or sha hummed with magic. Burners ignited flames, Tiders beckoned waves, and Zelie s Reaper mother summoned forth souls. But

everything changed the night magic disappeared. Under the orders of a ruthless king, maji were killed, leaving Zelie without a mother and her people without hope. 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 Clary, who is still seeking a cure for her mother's enchantment, travals to the City of Glass, the capital of the forbidden country of the secretive Shadowhunters, where she uncovers important truths about her family's past.
Summary Type-Driven Development with Idris, written by the creator of Idris, teaches you how to improve the performance and accuracy of your programs by taking advantage of a state-of-the-art type system. This book teaches you with Idris, a language designed to support type-driven development. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Stop fighting type errors! Type-driven development is an approach to coding that embraces types as the foundation of your code - essentially as built-in documentation your compiler can use to check data relationships and other assumptions. With this approach, you can define specifications early in development and write code that's easy to maintain, test, and extend. Idris is a Haskell-like language with first-class, dependent types that's perfect for learning type-driven programming techniques you can apply in any codebase. About the Book Type-Driven Development with Idris teaches you how to improve the performance and accuracy of your code by

taking advantage of a state-of-the-art type system. In this book, you'll learn type-driven development of real-world software, as well as how to handle side effects, interaction, state, and concurrency. By the end, you'll be able to develop robust and verified software in Idris and apply type-driven development methods to other languages. What's Inside Understanding dependent types Types as first-class language constructs Types as a guide to program construction Expressing relationships between data About the Reader Written for programmers with knowledge of functional programming concepts. About the Author Edwin Brady leads the design and implementation of the Idris language. Table of Contents PART 1 - INTRODUCTION Overview Getting started with IdrisPART 2 - CORE IDRIS Interactive development with types User-defined data types Interactive programs: input and output processing Programming with first-class types Interfaces: using constrained generic types Equality: expressing relationships between data Predicates: expressing assumptions and contracts in types Views: extending pattern matching PART 3 - IDRIS AND THE REAL WORLD Streams and processes: working with infinite data Writing programs with state State machines: verifying protocols in types Dependent state machines: handling feedback and errors Type-safe concurrent programming

Summary Get Programming with Haskell leads you through short lessons, examples, and exercises designed to make Haskell your own. It has crystal-clear illustrations and guided practice. You will write and test dozens of interesting programs and dive into custom Haskell modules. You will gain a new perspective on programming plus the practical ability to use Haskell in the everyday world. (The 80 IQ points: not guaranteed.) Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Programming languages often differ only around the edges—a few keywords, libraries, or platform choices. Haskell gives you an entirely new point of view. To the software pioneer Alan Kay, a change in perspective can be worth 80 IQ points and Haskellers agree on the dramatic benefits of thinking the Haskell way—thinking functionally, with type safety, mathematical certainty, and more. In this hands-on book, that's exactly what you'll learn to do. What's Inside Thinking in Haskell Functional programming basics Programming in types Real-world applications for Haskell About the Reader Written for readers who know one or more programming languages. Table of Contents Lesson 1 Getting started with Haskell Unit 1 - FOUNDATIONS OF FUNCTIONAL PROGRAMMING Lesson 2 Functions and functional programming Lesson 3 Lambda functions and lexical scope Lesson 4 First-class functions Lesson 5 Closures and partial application Lesson 6 Lists Lesson 7 Rules for recursion and pattern matching Lesson 8 Writing recursive functions Lesson 9 Higher-order functions Lesson 10 Capstone: Functional object-oriented programming with robots! Unit 2 - INTRODUCING TYPES Lesson 11 Type basics Lesson 12 Creating your own types Lesson 13 Type classes Lesson 14 Using type classes Lesson 15 Capstone: Secret messages! Unit 3 - PROGRAMMING IN TYPES Lesson 16 Creating types with "and" and "or" Lesson 17 Design by composition—Semigroups and Monoids Lesson 18 Parameterized types Lesson 19 The Maybe type: dealing with missing values Lesson 20 Capstone: Time series Unit 4 - IO IN HASKELL Lesson 21 Hello World!—introducing IO types Lesson 22 Interacting with the command line and lazy I/O Lesson 23 Working with text and Unicode Lesson 24 Working with files Lesson 25 Working with binary data Lesson 26 Capstone:

Processing binary files and book data Unit 5 - WORKING WITH TYPE IN A CONTEXT Lesson 27 The Functor type class Lesson 28 A peek at the Applicative type class: using functions in a context Lesson 29 Lists as context: a deeper look at the Applicative type class Lesson 30 Introducing the Monad type class Lesson 31 Making Monads easier with donotation Lesson 32 The list monad and list comprehensions Lesson 33 Capstone: SQL-like queries in Haskell Unit 6 - ORGANIZING CODE AND BUILDING PROJECTS Lesson 34 Organizing Haskell code with modules Lesson 35 Building projects with stack Lesson 36 Property testing with QuickCheck Lesson 37 Capstone: Building a prime-number library Unit 7 - PRACTICAL HASKELL Lesson 38 Errors in Haskell and the Either type Lesson 39 Making HTTP requests in Haskell Lesson 40 Working with JSON data by using Aeson Lesson 41 Using databases in Haskell Lesson 42 Efficient, stateful arrays in Haskell Afterword - What's next? Appendix - Sample answers to exercise

"A fully realized fantasy world complete with dragons, treachery, and flawed characters discovering their courage. I couldn't put it down!" —C. J. Redwine, New York Times bestselling author of The Shadow Queen From the New York Times bestselling co-author of My Lady Jane comes a smoldering new fantasy trilogy perfect for fans of Victoria Aveyard and Kristin Cashore about a girl condemned for defending dragons and the inner fire that may be her only chance of escape. Mira has always been a symbol of hope for the Fallen Isles, perfect and beautiful—or at least that's how she's forced to appear. But when she uncovers a dangerous secret, Mira is betrayed by those closest to her and sentenced to the deadliest prison in the Fallen Isles. Except Mira is over being a pawn. Fighting to survive against outer threats and inner demons of mental illness, Mira must find her inner fire and the scorching truth about her own endangered magic—before her very world collapses. And that's all before she ignites. A handbook to the Coq software for writing and checking mathematical proofs, with a practical engineering focus. The technology of mechanized program verification can play a supporting role in many kinds of research projects in computer science, and related tools for formal proof-checking are seeing increasing adoption in mathematics and engineering. This book provides an introduction to the Coq software for writing and checking mathematical proofs. It takes a practical engineering focus throughout, emphasizing techniques that will help users to build, understand, and maintain large Coq developments and minimize the cost of code change over time. Two topics, rarely discussed elsewhere, are covered in detail: effective dependently typed programming (making productive use of a feature at the heart of the Coq system) and construction of domain-specific proof tactics. Almost every subject covered is also relevant to interactive computer theorem proving in general, not just program verification, demonstrated through examples of verified programs applied in many different sorts of formalizations. The book develops a unique automated proof style and applies it throughout; even experienced Coq users may benefit from reading about basic Coq concepts from this novel perspective. The book also offers a library of tactics, or programs that find proofs, designed for use with examples in the book. Readers will acquire the necessary skills to reimplement these tactics in other settings by the end of the book. All of the code appearing in the book is freely available online.

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 Development of formal proofs of correctness of programs can increase actual and perceived reliability and facilitate better understanding of program specifications and their underlying assumptions. Tools supporting such development have been available for over 40 years but have only recently seen wide practical use. Projects based on construction of machine-checked formal proofs are now reaching an unprecedented scale, comparable to large software projects, which leads to new challenges in proof development and maintenance. Despite its increasing importance, the field of proof engineering is seldom considered in its own right; related theories, techniques, and tools span many fields and venues. QED at Large covers the timeline and research literature concerning proof development for program verification, including theories, languages, and tools. It emphasizes challenges and breakthroughs at each stage in history and highlights challenges that are currently present due to the increasing scale of proof developments. This monograph is intended for use by researchers and students who are new to the field. It provides the reader with an insightful overview of the work that has led to modern-day techniques for formally verifying software. In times of increasing automation, this underpins many software systems so future trends are also highlighted.

"A refreshing and unique coming-of-age story...a beautiful and necessary meditation on finding strength in one's culture." —Entertainment Weekly, Top Pick of the Month "A YA marvel that will shock breath into your lungs. If you loved The Wrath and the Dawn and Children of Blood and Bone, Mirage will captivate you." —The Christian Science Monitor "This debut fantasy has what it takes to be the next big thing in sci-fi/fantasy." —SLJ, starred review "Immersive, captivating." —ALA Booklist, starred review In a world dominated by the brutal Vathek empire, eighteen-year-old Amani is a dreamer. She dreams of what life was like before the occupation; she dreams of writing poetry like the old-world poems she adores; she dreams of receiving a sign from Dihya that one day, she, too, will have adventure, and travel beyond her isolated home. But when adventure comes for Amani, it is not what she expects: she is kidnapped by the regime and taken in secret to the royal palace, where she discovers that she is nearly identical to the cruel half-Vathek Princess Maram. The princess is so hated by her conquered people that she requires a body double, someone to appear in public as Maram, ready to die in her place. As Amani is forced into her new role, she can't help but enjoy the palace's beauty—and her time with the princess' fiancé, Idris. But the glitter of the royal court belies a world of violence and fear. If Amani ever wishes to see her family again,

she must play the princess to perfection...because one wrong move could lead to her death.

Summary Nim is a multi-paradigm language that offers powerful customization options with the ability to compile to everything from C to JavaScript. In Nim in Action you'll learn how Nim compares to other languages in style and performance, master its structure and syntax, and discover unique features. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Nim is a multi-paradigm programming language that offers powerful customization options with the ability to compile to everything from C to JavaScript. It can be used in any project and illustrates that you don't have to sacrifice performance for expressiveness! About the Book Nim in Action is your guide to application development in Nim. You'll learn how Nim compares to other languages in style and performance, master its structure and syntax, and discover unique features. By carefully walking through a Twitter clone and other real-world examples, you'll see just how Nim can be used every day while also learning how to tackle concurrency, package finished applications, and interface with other languages. With the best practices and rich examples in this book, you'll be able to start using Nim today. What's Inside Language features and implementation Nimble package manager Asynchronous I/O Interfacing with C and JavaScript Metaprogramming About the Reader For developers comfortable with mainstream languages like Java, Python, C++ or C#. About the Author Dominik Picheta is one of the principal developers of Nim and author of the Nimble package manager. Summary PART 1 -THE BASICS OF NIM Why Nim? Getting started PART 2 - NIM IN PRACTICE 3 Writing a chat application 4 A tour through the standard library 5 Package management 6 Parallelism 7 Building a Twitter clone PART 3 - ADVANCED CONCEPTS 8 Interfacing with other languages 9 Metaprogramming

Haskell in Depth unlocks a new level of skill with this challenging language. Going beyond the basics of syntax and structure, this book opens up critical topics like advanced types, concurrency, and data processing. Summary Turn the corner from "Haskell student" to "Haskell developer." Haskell in Depth explores the important language features and programming skills you'll need to build production-quality software using Haskell. And along the way, you'll pick up some interesting insights into why Haskell looks and works the way it does. Get ready to go deep! Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the technology Software for high-precision tasks like financial transactions, defense systems, and scientific research must be absolutely, provably correct. As a purely functional programming language, Haskell enforces a mathematically rigorous approach that can lead to concise, efficient, and bug-free code. To write such code you'll need deep understanding. You can get it from this book! About the book Haskell in Depth unlocks a new level of skill with this challenging language. Going beyond the basics of syntax and structure, this book opens up critical topics like advanced types, concurrency, and data processing. You'll discover key parts of the Haskell ecosystem and master core design patterns that will transform how you write software. What's inside Building applications, web services, and networking apps Using sophisticated libraries like lens, singletons, and servant Organizing projects with Cabal and Stack Error-handling and testing Pure parallelism for multicore processors About the reader For developers familiar with Haskell basics. About the author Vitaly Bragilevsky has been teaching Haskell and functional programming since 2008. He is a member of the GHC Steering Committee. Table of Contents PART 1 CORE HASKELL 1 Functions and types 2 Type classes 3 Developing an application: Stock quotes PART 2 INTRODUCTION TO APPLICATION DESIGN 4 Haskell development with modules, packages, and projects 5 Monads as practical functionality providers 6 Structuring programs with monad transformers PART 3 QUALITY ASSURANCE 7 Error handling and logging 8 Writing tests 9 Haskell data and code at run time 10 Benchmarking and

profiling PART 4 ADVANCED HASKELL 11 Type system advances 12 Metaprogramming in Haskell 13 More about types PART 5 HASKELL TOOLKIT 14 Data-processing pipelines 15 Working with relational databases 16 Concurrency

Users can dramatically improve the design, performance, and manageability of object-oriented code without altering its interfaces or behavior. "Refactoring" shows users exactly how to spot the best opportunities for refactoring and exactly how to do it, step by step.

Set over the course of one day, Aminah Mae Safi's This Is All Your Fault is a smart and voice-driven YA novel that follows three young women determined to save their indie bookstore. Rinn Olivera is finally going to tell her longtime crush AJ that she's in love with him. Daniella Korres writes poetry for her own account, but nobody knows it's her. Imogen Azar is just trying to make it through the day. When Rinn, Daniella, and Imogen clock into work at Wild Nights Bookstore on the first day of summer, they're expecting the hours to drift by the way they always do. Instead, they have to deal with the news that the bookstore is closing. Before the day is out, there'll be shaved heads, a diva author, and a very large shipment of Air Jordans to contend with. And it will take all three of them working together if they have any chance to save Wild Nights Bookstore.

Basic, no nonsense introduction to the programming language Scheme

Class-tested and coherent, this textbook teaches classical and web information retrieval, including web search and the related areas of text classification and text clustering from basic concepts. It gives an up-to-date treatment of all aspects of the design and implementation of systems for gathering, indexing, and searching documents; methods for evaluating systems; and an introduction to the use of machine learning methods on text collections. All the important ideas are explained using examples and figures, making it perfect for introductory courses in information retrieval for advanced undergraduates and graduate students in computer science. Based on feedback from extensive classroom experience, the book has been carefully structured in order to make teaching more natural and effective. Slides and additional exercises (with solutions for lecturers) are also available through the book's supporting website to help course instructors prepare their lectures.

No ordinary collection of tales, this anthology was the result of extensive research that led Shah to conclude that there is a certain basic fund of human fictions which recur again and again throughout the world and never seem to lose their compelling attraction. This special paperback version of World Tales concentrates on the essentials, the text of the stories, and omits the illustrations which were part of a previous edition.

Any programmer working with a dynamically typed language will tell you how hard it is to scale to more lines of code and more engineers. That's why Facebook, Google, and Microsoft invented gradual static type layers for their dynamically typed JavaScript and Python code. This practical book shows you how one such type layer, TypeScript, is unique among them: it makes programming fun with its powerful static type system. If you're a programmer with intermediate JavaScript experience, author Boris Cherny will teach you how to master the TypeScript language. You'll understand how TypeScript can help you eliminate bugs in your code and enable you to scale your code across more engineers than you could before. In this book, you'll: Start with the basics: Learn about TypeScript's different types and type operators, including what they're for and how they're used Explore advanced topics: Understand TypeScript's sophisticated type system, including how to safely handle errors and build asynchronous programs Dive in hands-on: Use TypeScript with your favorite frontend and backend frameworks, migrate your existing JavaScript project to TypeScript, and run your TypeScript application in production

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.

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

Summary Functional and Reactive Domain Modeling teaches you how to think of the domain model in terms of pure functions and how to compose them to build larger abstractions. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Traditional distributed applications won't cut it in the reactive world of microservices, fast data, and sensor networks. To capture their dynamic relationships and dependencies, these systems require a different approach to domain modeling. A domain model composed of pure functions is a more natural way of representing a process in a reactive system, and it maps directly onto technologies and patterns like Akka, CQRS, and event sourcing. About the Book Functional and Reactive Domain Modeling teaches you consistent, repeatable techniques for building domain models in reactive systems. This book reviews the relevant concepts of FP and reactive architectures and then methodically introduces this new approach to domain modeling. As you read, you'll learn where and how to apply it, even if your systems aren't purely reactive or functional. An expert blend of theory and practice, this book presents strong examples you'll return to again and again as you apply these principles to your own projects. What's Inside Real-world libraries and frameworks Establish meaningful reliability guarantees Isolate domain logic from side effects Introduction to reactive design patterns About the Reader Readers should be comfortable with functional programming and traditional domain modeling. Examples use the Scala language. About the Author Software architect Debasish Ghosh was an early adopter of reactive design using Scala and Akka. He's the author of DSLs in Action, published by Manning in 2010. Table of Contents Functional domain modeling: an introduction Scala for functional domain models Designing functional domain models Functional patterns for domain models Modularization of domain models Being reactive Modeling with reactive streams Reactive persistence and event sourcing Testing your domain model Summary - core thoughts and principles

From #1 New York Times bestselling author Brandon Sanderson, Warbreaker is the story of two sisters, who happen to be princesses, the God King one of them has to marry, the lesser god who doesn't like his job, and the immortal who's still trying to undo the mistakes he made hundreds of years ago. Their world is one in which those who die in glory return as gods to live confined to a pantheon in Hallandren's capital city and where a power known as BioChromatic magic is based on an essence known as breath

that can only be collected one unit at a time from individual people. By using breath and drawing upon the color in everyday objects, all manner of miracles and mischief can be accomplished. It will take considerable quantities of each to resolve all the challenges facing Vivenna and Siri, princesses of Idris; Susebron the God King; Lightsong, reluctant god of bravery, and mysterious Vasher, the Warbreaker. Other Tor books by Brandon Sanderson The Cosmere The Stormlight Archive The Way of Kings Words of Radiance Edgedancer (Novella) Oathbringer The Mistborn trilogy Mistborn: The Final Empire The Well of Ascension The Hero of Ages Mistborn: The Wax and Wayne series Alloy of Law Shadows of Self Bands of Mourning Collection Arcanum Unbounded Other Cosmere novels Elantris Warbreaker The Alcatraz vs. the Evil Librarians series Alcatraz vs. the Evil Librarians The Scrivener's Bones The Knights of Crystallia The Shattered Lens The Dark Talent The Rithmatist series The Rithmatist Other books by Brandon Sanderson The Reckoners Steelheart Firefight Calamity At the Publisher's request, this title is being sold without Digital Rights Management Software (DRM) applied.

The New York Times bestseller and Read with Jenna TODAY SHOW Book Club pick telling the story of three generations of Palestinian-American women struggling to express their individual desires within the confines of their Arab culture in the wake of shocking intimate violence in their community. A GOODREADS CHOICE AWARDS FINALIST FOR BEST FICTION AND BEST DEBUT • BOOKBROWE'S BEST BOOK OF THE YEAR • A MARIE CLAIRE BEST WOMEN'S FICTION OF THE YEAR • A REAL SIMPLE BEST BOOK OF THE YEAR • A POPSUGAR BEST BOOK OF THE YEAR ALL WRITTEN BY FEMALES A New York Times Book Review Editors' Choice • A Washington Post 10 Books to Read in March • A Newsweek Best Book of the Summer • A USA Today Best Book of the Week • A Washington Book Review Difficult-To-Put-Down Novel • A Refinery 29 Best Books of the Month • A Buzzfeed News 4 Books We Couldn't Put Down Last Month • A New Arab Best Books by Arab Authors • An Electric Lit 20 Best Debuts of the First Half of 2019 • A The Millions Most Anticipated Books of 2019 "Garnering justified comparisons to Khaled Hosseini's A Thousand Splendid Suns... Etaf Rum's debut novel is a must-read about women mustering up the bravery to follow their inner voice." —Refinery 29 "Where I come from, we've learned to silence ourselves. We've been taught that silence will save us. Where I come from, we keep these stories to ourselves. To tell them to the outside world is unheard of—dangerous, the ultimate shame." Palestine, 1990. Seventeen-year-old Isra prefers reading books to entertaining the suitors her father has chosen for her. Over the course of a week, the naïve and dreamy girl finds herself quickly betrothed and married, and is soon living in Brooklyn. There Isra struggles to adapt to the expectations of her oppressive mother-in-law Fareeda and strange new husband Adam, a pressure that intensifies as she begins to have children—four daughters instead of the sons Fareeda tells Isra she must bear. Brooklyn, 2008. Eighteen-year-old Deya, Isra's oldest daughter, must meet with potential husbands at her grandmother Fareeda's insistence, though her only desire is to go to college. Deya can't help but wonder if her options would have been different had her parents survived the car crash that killed them when Deya was only eight. But her grandmother is firm on the matter: the only way to secure a worthy future for Deya is through marriage to the right man. But fate has a will of its own, and soon Deya will find herself on an unexpected path

that leads her to shocking truths about her family—knowledge that will force her to question everything she thought she knew about her parents, the past, and her own future.

Despite using them every day, most software engineers know little about how programming languages are designed and implemented. For many, their only experience with that corner of computer science was a terrifying "compilers" class that they suffered through in undergrad and tried to blot from their memory as soon as they had scribbled their last NFA to DFA conversion on the final exam. That fearsome reputation belies a field that is rich with useful techniques and not so difficult as some of its practitioners might have you believe. A better understanding of how programming languages are built will make you a stronger software engineer and teach you concepts and data structures you'll use the rest of your coding days. You might even have fun. This book teaches you everything you need to know to implement a full-featured, efficient scripting language. You'll learn both high-level concepts around parsing and semantics and gritty details like bytecode representation and garbage collection. Your brain will light up with new ideas, and your hands will get dirty and calloused. Starting from main(), you will build a language that features rich syntax, dynamic typing, garbage collection, lexical scope, first-class functions, closures, classes, and inheritance. All packed into a few thousand lines of clean, fast code that you thoroughly understand because you wrote each one yourself.

Copyright: 8173bec887ca616529717cb9c11c74f4