Read Online C Concurrency In Action

This is likewise one of the factors by obtaining the soft documents of this **C Concurrency In Action** by online. You might not require more times to spend to go to the book opening as skillfully as search for them. In some cases, you likewise reach not discover the notice C Concurrency In Action that you are looking for. It will enormously squander the time.

However below, similar to you visit this web page, it will be so enormously easy to acquire as with ease as download lead C Concurrency In Action

It will not acknowledge many get older as we notify before. You can complete it even though put-on something else at house and even in your workplace. appropriately easy! So, are you question? Just exercise just what we offer below as capably as evaluation **C Concurrency In Action** what you taking into consideration to read!

A1RBZU - DANIELA ROBINSON

Writing reliable and maintainable C++ software is hard. Designing such software at scale adds a new set of challenges. Creating large-scale systems requires a practical understanding of logical design – beyond the theoretical concepts addressed in most popular texts. To be successful on an enterprise scale, developers must also address physical design, a dimension of software engineering that may be unfamiliar even to expert developers. Drawing on over 30 years of hands-on experience building massive, mission-critical enterprise systems, John Lakos shows how to create and grow Software Capital. This groundbreaking volume lays the foundation for projects of all sizes and demonstrates the processes, methods, techniques, and tools needed for successful real-world, large-scale development. Up to date and with a solid engineering focus, Large-Scale C++, Volume I: Process and Architecture, demonstrates fundamental design concepts with concrete examples. Professional developers of all experience levels will gain insights that transform their approach to design and development by understanding how to Raise productivity by leveraging differences between infrastructure and application development Achieve exponential productivity gains through feedback and hierarchical reuse Embrace the component’s role as the fundamental unit of both logical and physical design Analyze how fundamental properties of compiling and linking affect component design Discover effective partitioning of logical content in appropriately sized physical aggregates Internalize the important differences among sufficient, complete, minimal, and primitive software Deliver solutions that simultaneously optimize encapsulation, stability, and performance Exploit the nine established levelization techniques to avoid cyclic physical dependencies Use lateral designs judiciously to avoid the “heaviness” of conventional layered architectures Employ appropriate architectural insulation techniques for eliminating compile-time coupling Master the multidimensional process of designing large systems using component-based methods This is the first of John Lakos’s three authoritative volumes on developing large-scale systems using C++. This book, written for fellow software practitioners, uses familiar C++ constructs to solve real-world problems while identifying (and motivating) modern C++ alternatives. Together with the forthcoming Volume II: Design and Implementation and Volume III: Verification and Testing, Large-Scale C++ offers comprehensive guidance for all aspects of large-scale C++ software development. If you are an architect or project leader, this book will empower you to solve critically important problems right now – and serve as your go-to reference for years to come. Register your book for convenient access to downloads, updates, and/or corrections as they become available. See inside book for details.

This title documents a convergence of programming techniques - generic programming, template metaprogramming, object-oriented programming and design patterns. It describes the C++ techniques used in generic programming and implements a number of industrial strength components.

Clojure is a practical, general-purpose language that offers expressivity rivaling other dynamic languages like Ruby and Python, while seamlessly taking advantage of Java libraries, services, and all of the resources of the JVM ecosystem. This book helps you learn the fundamentals of Clojure with examples relating it to the languages you know already, in the domains and topics you work with every day. See how this JVM language can help eliminate unnecessary complexity from your programming practice and open up new options for solving the most challenging problems. Clojure Programming demonstrates the language’s flexibility by showing how it can be used for common tasks like web programming and working with databases, up through more demanding applications that require safe, effective concurrency and parallelism, data analysis, and more. This in-depth look helps tie together the full Clojure development experience, from how to organize your project and an introduction to Clojure build tooling, to a tutorial on how to make the most of Clojure’s REPL during development, and how to deploy your finished application in a cloud environment. Learn how to use Clojure while leveraging your investment in the Java platform Understand the advantages of Clojure as an efficient Lisp for the JVM See how Clojure is used today in several practical domains Discover how Clojure eliminates the need for many verbose and complicated design patterns Deploy large or small web applications to the cloud with Clojure

A fast-paced, thorough introduction to modern C++ written for experienced programmers. After reading C++ Crash Course, you'll be proficient in the core language concepts, the C++ Standard Library, and the Boost Libraries. C++ is one of the most widely used languages for real-world software. In the hands of a knowledgeable programmer, C++ can produce small, efficient, and readable code that any programmer would be proud of. Designed for intermediate to advanced programmers, C++ Crash Course cuts through the weeds to get you straight to the core of C++17, the most modern revision of the ISO standard. Part 1 covers the core of the C++ language, where you'll learn about everything from types and functions, to the object life cycle and expressions. Part 2 introduces you to the C++ Standard Library and Boost Libraries, where you'll learn about all of the high-quality, fully-featured facilities available to you. You'll cover special utility classes, data structures, and algorithms, and learn how to manipulate file systems and build high-performance programs that communicate over networks. You'll learn all the major features of modern C++, including: • Fundamental types, reference types, and user-defined types • The object lifecycle including storage duration, memory management, exceptions, call stacks, and the RAII paradigm • Compile-time polymorphism with templates and run--time polymorphism with virtual classes • Advanced expressions, statements, and functions • Smart pointers, data structures, dates and times, numerics, and probability/statistics facilities • Containers, iterators, strings, and algorithms • Streams and files, concurrency, networking, and application development With well over 500 code samples and nearly 100 exercises, C++ Crash Course is sure to help you build a strong C++ foundation.

Software -- Programming Languages.

Rust in Action introduces the Rust programming language by exploring numerous systems programming concepts and techniques.You'll be learning Rust by delving into how computers work under the hood. You'll find yourself playing with persistent storage, memory, networking and even tinkering with CPU instructions. The book takes you through using Rust to extend other applications and teaches you tricks to write blindingly fast code. You'll also discover parallel and concurrent programming. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications.

With a new name and a new focus on CORBA, database drivers, and Microsoft Back Office applications, Inprise/Borland Delphi is enjoying a resurgence, with a growing user base of programmers who use Delphi for rapid development of enterprise computing applications. Not to rest on success, the latest version of Delphi, Version 5, includes further expansion and refinement of the 3-tier application framework introduced in Delphi 4 and has resulted in a prize-winning product.Delphi in a Nutshell is the first concise reference to Borland/Inprise Delphi available. It succinctly collects all the information you need in one easy-to-use, complete, and accurate volume that goes beyond the product documentation itself.Delphi in a Nutshell starts with the Delphi object model and how to use RTTI (Run Time Type Information) for efficient programming. The rest of the book is the most complete Delphi Pascal language reference available in print, detailing every language element with complete syntax, examples, and methods for use. The book concludes with a look at the compiler, discussing compiler directives in depth.

Summary Go in Action introduces the Go language, guiding you from inquisitive developer to Go guru. The book begins by introducing the unique features and concepts of Go. Then, you'll get hands-on experience writing real-world applications including websites and network servers, as well as techniques to manipulate and convert data at speeds that will make your friends jealous. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Application development can be tricky enough even when you aren't dealing with complex systems programming problems like web-scale concurrency and real-time performance. While it's possible to solve these common issues with additional tools and frameworks, Go handles them right out of the box, making for a more natural and productive coding experience. Developed at Google, Go powers nimble startups as well as big enterprises—companies that rely on high-performing services in their infrastructure. About the Book Go in Action is for any intermediate-level developer who has experience with other programming languages and wants a jump-start in learning Go or a more thorough understanding of the language and its internals. This book provides an intensive, comprehensive, and idiomatic view of Go. It focuses on the specification and implementation of the language, including topics like language syntax, Go's type system, concurrency, channels, and testing. What's Inside Language specification and implementation Go's type system Internals of Go's data structures Testing and benchmarking About the Reader This book assumes you're a working developer proficient with another language like Java, Ruby, Python, C#, or C++. About the Authors William Kennedy is a seasoned software developer and author of the blog GoingGo.Net. Brian Ketelsen and Erik St. Martin are the organizers of GopherCon and coauthors of the Go-based Skynet framework. Table of Contents Introducing Go Go quick-start Packaging and tooling Arrays, slices, and maps Go's type system Concurrency Concurrency patterns Standard library Testing and benchmarking

Since the introduction of Hoares' Communicating Sequential Processes notation, powerful new tools have transformed CSP into a practical way of describing industrial-sized problems. This book gives you the fundamental grasp of CSP concepts you'll need to take advantage of those tools.Part I provides a detailed foundation for working with CSP, using as little mathematics as possible. It introduces the ideas behind operational, denotational and algebraic models of CSP. Parts II and III go into greater detail about theory and practice. Topics include: parallel operators, hiding and renaming, piping and enslavement, buffers and communication, termination and sequencing, and semantic theory. Three detailed practical case studies are also presented.For anyone interested in modeling sequential processes.

Summary Scala in Action is a comprehensive tutorial that introduces Scala through clear explanations and numerous hands-on examples. Because Scala is a rich and deep language, it can be daunting to absorb all the new concepts at once. This book takes a "how-to" approach, explaining language concepts as you explore familiar programming challenges that you face in your day--to-day work. About the Technology Scala runs on the JVM and combines object-orientation with functional programming. It's designed to produce succinct, type-safe code, which is crucial for enterprise applications. Scala implements Actor-based concurrency through the amazing Akka framework, so you can avoid Java's messy threading while interacting seamlessly with Java. About this Book Scala in Action is a comprehensive tutorial that introduces the language through clear explanations and numerous hands-on examples. It takes a "how to" approach, explaining language concepts as you explore familiar programming tasks. You'll tackle concurrent programming in Akka, learn to work with Scala and Spring, and learn how to build DSLs and other productivity tools. You'll learn both the language and how to use it. Experience with Java is helpful but not required. Ruby and Python programmers will also find this book accessible. What's Inside A Scala tutorial How to use Java and Scala open source libraries How to use SBT Test-driven development Debugging Updated for Scala 2.10 Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Author Nilanjan Raychaudhuri is a skilled developer, speaker, and an avid polyglot programmer who works with Scala on production systems. Table of Contents PART 1 SCALA: THE BASICS Why Scala? Getting started OOP in Scala Having fun with functional data structures Functional programming PART 2 WORKING WITH SCALA Building web applications in functional style Connecting to a database Building scalable and extensible components Concurrency programming in Scala Building confidence with testing PART 3 ADVANCED STEPS Interoperability between Scala and Java Scalable and distributed applications using Akka

Concurrency can be notoriously difficult to get right, but fortunately, the Go open source programming language makes working with concurrency tractable and even easy. If you’re a developer familiar with Go, this practical book demonstrates best practices and patterns to help you incorporate concurrency into your systems. Author Katherine Cox-Buday takes you step-by-step through the process. You’ll understand how Go chooses to model concurrency, what issues arise from this model, and how you can compose primitives within this model to solve problems. Learn the skills and tooling you need to confidently write and implement concurrent systems of any size. Understand how Go addresses fundamental problems that make concurrency difficult to do correctly Learn the key differences between concurrency and parallelism Dig into the syntax of Go’s memory synchronization primitives Form patterns with these primitives to write maintainable concurrent code Compose patterns into a series of practices that enable you to write large, distributed systems that scale Learn the sophistication behind goroutines and how Go’s runtime stitches everything together

The C++11 standard allows programmers to express ideas more clearly, simply, and directly, and to write faster, more efficient code. Bjarne Stroustrup, the designer and original implementer of C++, thoroughly covers the details of this language and its use in his definitive reference, The C++ Programming Language, Fourth

Edition. In A Tour of C++ , Stroustrup excerpts the overview chapters from that complete reference, expanding and enhancing them to give an experienced programmer–in just a few hours–a clear idea of what constitutes modern C++. In this concise, self--contained guide, Stroustrup covers most major language features and the major standard-library components–not, of course, in great depth, but to a level that gives programmers a meaningful overview of the language, some key examples, and practical help in getting started. Stroustrup presents the C++ features in the context of the programming styles they support, such as object-oriented and generic programming. His tour is remarkably comprehensive. Coverage begins with the basics, then ranges widely through more advanced topics, including many that are new in C++11, such as move semantics, uniform initialization, lambda expressions, improved containers, random numbers, and concurrency. The tour ends with a discussion of the design and evolution of C++ and the extensions added for C++11. This guide does not aim to teach you how to program (see Stroustrup's Programming: Principles and Practice Using C++ for that); nor will it be the only resource you'll need for C++ mastery (see Stroustrup's The C++ Programming Language, Fourth Edition, for that). If, however, you are a C or C++ programmer wanting greater familiarity with the current C++ language, or a programmer versed in another language wishing to gain an accurate picture of the nature and benefits of modern C++, you can't find a shorter or simpler introduction than this tour provides.

Master the essentials of concurrent programming,including testingand debugging This textbook examines languages and libraries for multithreadedprogramming. Readers learn how to create threads in Java and C++,and develop essential concurrent programming and problem-solvingskills. Moreover, the textbook sets itself apart from othercomparable works by helping readers to become proficient in keytesting and debugging techniques. Among the topics covered, readersare introduced to the relevant aspects of Java, the POSIX Pthreadslibrary, and the Windows Win32 Applications ProgrammingInterface. The authors have developed and fine-tuned this book through theconcurrent programming courses they have taught for the past twentyyears. The material, which emphasizes practical tools andtechniques to solve concurrent programming problems, includesoriginal results from the authors' research. Chaptersinclude: * Introduction to concurrent programming * The critical section problem * Semaphores and locks * Monitors * Message-passing * Message-passing in distributed programs * Testing and debugging concurrent programs As an aid to both students and instructors, class libraries havebeen implemented to provide working examples of all the materialthat is covered. These libraries and the testing techniques theysupport can be used to assess student-written programs. Each chapter includes exercises that build skills in programwriting and help ensure that readers have mastered the chapter'skey concepts. The source code for all the listings in the text andfor the synchronization libraries is also provided, as well asstartup files and test cases for the exercises. This textbook is designed for upper-level undergraduates andgraduate students in computer science. With its abundance ofpractical material and inclusion of working code, coupled with anemphasis on testing and debugging, it is also a highly usefulreference for practicing programmers.

The official book on the Rust programming language, written by the Rust development team at the Mozilla Foundation, fully updated for Rust 2018. The Rust Programming Language is the official book on Rust: an open source systems programming language that helps you write faster, more reliable software. Rust offers control over low-level details (such as memory usage) in combination with high-level ergonomics, eliminating the hassle traditionally associated with low-level languages. The authors of The Rust Programming Language, members of the Rust Core Team, share their knowledge and experience to show you how to take full advantage of Rust's features--from installation to creating robust and scalable programs. You'll begin with basics like creating functions, choosing data types, and binding variables and then move on to more advanced concepts, such as: • Ownership and borrowing, lifetimes, and traits • Using Rust's memory safety guarantees to build fast, safe programs • Testing, error handling, and effective refactoring • Generics, smart pointers, multithreading, trait objects, and advanced pattern matching • Using Cargo, Rust's built-in package manager, to build, test, and document your code and manage dependencies • How best to use Rust's advanced compiler with compiler-led programming techniques You'll find plenty of code examples throughout the book, as well as three chapters dedicated to building complete projects to test your learning: a number guessing game, a Rust implementation of a command line tool, and a multithreaded server. New to this edition: An extended section on Rust macros, an expanded chapter on modules, and appendixes on Rust development tools and editions.

More than ever, learning to program concurrency is critical to creating faster, responsive applications. Speedy and affordable multicore hardware is driving the demand for high-performing applications, and you can leverage the Java platform to bring these applications to life. Concurrency on the Java platform has evolved, from the synchronization model of JDK to software transactional memory (STM) and actor-based concurrency. This book is the first to show you all these concurrency styles so you can compare and choose what works best for your applications. You'll learn the benefits of each of these models, when and how to use them, and what their limitations are. Through hands-on exercises, you'll learn how to avoid shared mutable state and how to write good, elegant, explicit synchronization-free programs so you can create easy and safe concurrent applications. The techniques you learn in this book will take you from dreading concurrency to mastering and enjoying it. Best of all, you can work with Java or a JVM language of your choice - Clojure, JRuby, Groovy, or Scala - to reap the growing power of multicore hardware. If you are a Java programmer, you'd need JDK 1.5 or later and the Akka 1.0 library. In addition, if you program in Scala, Clojure, Groovy or JRuby you'd need the latest version of your preferred language. Groovy programmers will also need GPars.

With the new C++ Standard and Technical Report 2 (TR2), multi--threading is coming to C++ in a big way. TR2 will provide higher-level synchronization facilities that allow for a much greater level of abstraction, and make programming multi-threaded applications simpler and safer. Concurrent programming is required if programmers are to take advantage of the multi-core microprocessors increasingly available from Intel and others. The new standard for C++ has extensions to the language that make concurrent programming more accessible to regular developers. As a guide and reference to the new concurrency features in the upcoming C++ Standard and TR2, this book is invaluable for existing programmers familiar with writing multi-threaded code in C++ using platform-specific APIs, or in other languages, as well as C++ programmers who have never written multithreaded code before.

If you're one of the many developers uncertain about concurrent and multithreaded development, this practical cookbook will change your mind. With more than 75 code-rich recipes, author Stephen Cleary demonstrates parallel processing and asynchronous programming techniques, using libraries and language features in .NET 4.5 and C# 5.0. Concurrency is becoming more common in responsive and scalable application development, but it's been extremely difficult to code. The detailed solutions in this cookbook show you how modern tools raise the level of abstraction, making concurrency much easier than before. Complete with ready-to-use code and discussions about how and why the solution works, you get recipes for using: async and await for asynchronous operations Parallel programming with the Task Parallel Library The TPL Dataflow library for creating dataflow pipelines Capabilities that Reactive Extensions build on top of LINQ Unit testing with concurrent code Interop scenarios for combining concurrent approaches Immutable, threadsafe, and producer/consumer collections Cancellation support in your concurrent code Asynchronous-friendly Object-Oriented Programming Thread synchronization for accessing data

Become a better programmer with performance improvement techniques such as concurrency, lock-free programming, atomic operations, parallelism, and memory management Key FeaturesLearn proven techniques from a heavyweight and recognized expert in C++ and high-performance computingUnderstand the limitations of modern CPUs and their performance impactFind out how you can avoid writing inefficient code and get the best optimizations from the compilerLearn the tradeoffs and costs of writing high-performance programsBook Description The great free lunch of "performance taking care of itself" is over. Until recently, programs got faster by themselves as CPUs were upgraded, but that doesn't happen anymore. The clock frequency of new processors has almost peaked, and while new architectures provide small improvements to existing programs, this only helps slightly. To write efficient software, you now have to know how to program by making good use of the available computing resources, and this book will teach you how to do that. The Art of Efficient Programming covers all the major aspects of writing efficient programs, such as using CPU resources and memory efficiently, avoiding unnecessary computations, measuring performance, and how to put concurrency and multithreading to good use. You'll also learn about compiler optimizations and how to use the programming language (C++) more efficiently. Finally, you'll understand how design decisions impact performance. By the end of this book, you'll not only have enough knowledge of processors and compilers to write efficient programs, but you'll also be able to understand which techniques to use and what to measure while improving performance. At its core, this book is about learning how to learn. What you will learnDiscover how to use the hardware computing resources in your programs effectivelyUnderstand the relationship between memory order and memory barriersFamiliarize yourself with the performance implications of different data structures and organizationsAssess the performance impact of concurrent memory accessed and how to minimize itDiscover when to use and when not to use lock-free programming techniquesExplore different ways to improve the effectiveness of compiler optimizationsDesign APIs for concurrent data structures and high-performance data structures to avoid inefficienciesWho this book is for This book is for experienced developers and programmers who work on performance-critical projects and want to learn new techniques to improve the performance of their code. Programmers in algorithmic trading, gaming, bioinformatics, computational genomics, or computational fluid dynamics communities will get the most out of the examples in this book, but the techniques are fairly universal. Although this book uses the C++ language, the concepts demonstrated in the book can be easily transferred or applied to other compiled languages such as C, Java, Rust, Go, and more.

Threads are a fundamental part of the Java platform. As multicore processors become the norm, using concurrency effectively becomes essential for building high-performance applications. Java SE 5 and 6 are a huge step forward for the development of concurrent applications, with improvements to the Java Virtual Machine to support high-performance, highly scalable concurrent classes and a rich set of new concurrency building blocks. In Java Concurrency in Practice , the creators of these new facilities explain not only how they work and how to use them, but also the motivation and design patterns behind them. However, developing, testing, and debugging multithreaded programs can still be very difficult; it is all too easy to create concurrent programs that appear to work, but fail when it matters most: in production, under heavy load. Java Concurrency in Practice arms readers with both the theoretical underpinnings and concrete techniques for building reliable, scalable, maintainable concurrent applications. Rather than simply offering an inventory of concurrency APIs and mechanisms, it provides design rules, patterns, and mental models that make it easier to build concurrent programs that are both correct and performant. This book covers: Basic concepts of concurrency and thread safety Techniques for building and composing thread--safe classes Using the concurrency building blocks in java.util.concurrent Performance optimization dos and don'ts Testing concurrent programs Advanced topics such as atomic variables, nonblocking algorithms, and the Java Memory Model

This book breaks down the C++ STL, teaching you how to extract its gems and apply them to your programming. About This Book Boost your productivity as a C++ developer with the latest features of C++17 Develop high-quality, fast, and portable applications with the varied features of the STL Migrate from older versions (C++11, C++14) to C++17 Who This Book Is For This book is for developers who would like to master the C++ STL and make full use of its components. Prior C++ knowledge is assumed. What You Will Learn Make your own iterator types, allocators, and thread pools. Master every standard container and every standard algorithm. Improve your code by replacing new/delete with smart pointers. Understand the difference between monomorphic algorithms, polymorphic algorithms, and generic algorithms. Learn the meaning and applications of vocabulary type, product type and sum type. In Detail Modern C++ has come a long way since 2011. The latest update, C++17, has just been ratified and several implementations are on the way. This book is your guide to the C++ standard library, including the very latest C++17 features. The book starts by exploring the C++ Standard Template Library in depth. You will learn the key differences between classical polymorphism and generic programming, the foundation of the STL. You will also learn how to use the various algorithms and containers in the STL to suit your programming needs. The next module delves into the tools of modern C++. Here you will learn about algebraic types such as std::optional, vocabulary types such as std::function, smart pointers, and synchronization primitives such as std::atomic and std::mutex. In the final module, you will learn about C++'s support for regular expressions and file I/O. By the end of the book you will be proficient in using the C++17 standard library to implement real programs, and you'll have gained a solid understanding of the library's own internals. Style and approach This book takes a concise but comprehensive approach to explaining and applying the C++ STL, one feature at a time.

Summary Revised and updated for Elixir 1.7, Elixir in Action, Second Edition teaches you how to apply Elixir to practical problems associated with scalability, fault tolerance, and high availability. Along the way, you'll develop an appreciation for, and considerable skill in, a functional and concurrent style of programming. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology When you're building mission-critical software, fault tolerance matters. The Elixir programming language delivers fast, reliable applications, whether you're building a large-scale distributed system, a set of backend services, or a simple web app. And Elixir's elegant syntax and functional programming mindset make your software easy to write, read, and maintain. About the Book Elixir in Action, Second Edition teaches you how to build production-quality distributed applications using the Elixir programming language. Author Saša Jurić introduces this powerful language using examples that highlight the benefits of Elixir's functional and concurrent programming. You'll discover how the OTP framework can radically reduce tedious low-level coding tasks. You'll also explore practical approaches to concurrency as you learn to distribute a production system over multiple machines. What's inside Updated for Elixir 1.7 Functional and concurrent programming Introduction to distributed system design Creating deployable releases About the Reader You'll need intermediate skills with client/server applications and a language like Java, C#, or Ruby. No previous experience with Elixir required. About the Author Saša Jurić is a developer with extensive experience using Elixir and Erlang in complex server-side systems. Table of Contents First steps

Building blocks Control flow Data abstractions Concurrency primitives Generic server processes Building a concurrent system Fault--tolerance basics Isolating error effects Beyond GenServer Working with components Building a distributed system Running the system

This book is an in-depth introduction to Erlang, a programming language ideal for any situation where concurrency, fault tolerance, and fast response is essential. Erlang is gaining widespread adoption with the advent of multi-core processors and their new scalable approach to concurrency. With this guide you'll learn how to write complex concurrent programs in Erlang, regardless of your programming background or experience. Written by leaders of the international Erlang community -- and based on their training material -- Erlang Programming focuses on the language's syntax and semantics, and explains pattern matching, proper lists, recursion, debugging, networking, and concurrency. This book helps you: Understand the strengths of Erlang and why its designers included specific features Learn the concepts behind concurrency and Erlang's way of handling it Write efficient Erlang programs while keeping code neat and readable Discover how Erlang fills the requirements for distributed systems Add simple graphical user interfaces with little effort Learn Erlang's tracing mechanisms for debugging concurrent and distributed systems Use the built-in Mnesia database and other table storage features Erlang Programming provides exercises at the end of each chapter and simple examples throughout the book.

Concurrent programming has become a required discipline for all programmers. Multi-core processors and the increasing demand for maximum performance and scalability in mission-critical applications have renewed interest in functional languages like Erlang that are designed to handle concurrent programming. Erlang, and the OTP platform, make it possible to deliver more robust applications that satisfy rigorous uptime and performance requirements. Erlang and OTP in Action teaches you to apply Erlang's message passing model for concurrent programming--a completely different way of tackling the problem of parallel programming from the more common multi-threaded approach. This book walks you through the practical considerations and steps of building systems in Erlang and integrating them with real-world C/C++, Java, and .NET applications. Unlike other books on the market, Erlang and OTP in Action offers a comprehensive view of how concurrency relates to SOA and web technologies. This hands-on guide is perfect for readers just learning Erlang or for those who want to apply their theoretical knowledge of this powerful language. You'll delve into the Erlang language and OTP runtime by building several progressively more interesting real-world distributed applications. Once you are competent in the fundamentals of Erlang, the book takes you on a deep dive into the process of designing complex software systems in Erlang. Purchase of the print book comes with an offer of a free PDF, ePub, and Kindle eBook from Manning. Also available is all code from the book.

A multi-user game, web site, cloud application, or networked database can have thousands of users all interacting at the same time. You need a powerful, industrial-strength tool to handle the really hard problems inherent in parallel, concurrent environments. You need Erlang. In this second edition of the bestselling Programming Erlang, you'll learn how to write parallel programs that scale effortlessly on multicore systems. Using Erlang, you'll be surprised at how easy it becomes to deal with parallel problems, and how much faster and more efficiently your programs run. That's because Erlang uses sets of parallel processes-not a single sequential process, as found in most programming languages. Joe Armstrong, creator of Erlang, introduces this powerful language in small steps, giving you a complete overview of Erlang and how to use it in common scenarios. You'll start with sequential programming, move to parallel programming and handling errors in parallel programs, and learn to work confidently with distributed programming and the standard Erlang/Open Telecom Platform (OTP) frameworks. You need no previous knowledge of functional or parallel programming. The chapters are packed with hands-on, real-world tutorial examples and insider tips and advice, and finish with exercises for both beginning and advanced users. The second edition has been extensively rewritten. New to this edition are seven chapters covering the latest Erlang features: maps, the type system and the Dialyzer, WebSockets, programming idioms, and a new stand-alone execution environment. You'll write programs that dynamically detect and correct errors, and that can be upgraded without stopping the system. There's also coverage of rebar (the de facto Erlang build system), and information on how to share and use Erlang projects on github, illustrated with examples from cowboy and bitcask. Erlang will change your view of the world, and of how you program. What You Need The Erlang/OTP system. Download it from erlang.org.

Coming to grips with C++11 and C++14 is more than a matter of familiarizing yourself with the features they introduce (e.g., auto type declarations, move semantics, lambda expressions, and concurrency support). The challenge is learning to use those features effectively—so that your software is correct, efficient, maintainable, and portable. That’s where this practical book comes in. It describes how to write truly great software using C++11 and C++14—i.e. using modern C++. Topics include: The pros and cons of braced initialization, noexcept specifications, perfect forwarding, and smart pointer make functions The relationships among std::move, std::forward, rvalue references, and universal references Techniques for writing clear, correct, effective lambda expressions How std::atomic differs from volatile, how each should be used, and how they relate to C++'s concurrency API How best practices in "old" C++ programming (i.e., C++98) require revision for software development in modern C++ Effective Modern C++ follows the proven guideline-based, example-driven format of Scott Meyers' earlier books, but covers entirely new material. "After I learned the C++ basics, I then learned how to use C++ in production code from Meyer's series of Effective C++ books. Effective Modern C++ is the most important how-to book for advice on key guidelines, styles, and idioms to use modern C++ effectively and well. Don't own it yet? Buy this one. Now". -- Herb Sutter, Chair of ISO C++ Standards Committee and C++ Software Architect at Microsoft

C++ Concurrency in Action, Second Edition is the definitive guide to writing elegant multithreaded applications in C++. Updated for C++ 17, it carefully addresses every aspect of concurrent development, from starting new threads to designing fully functional multithreaded algorithms and data structures. Concurrency master Anthony Williams presents examples and practical tasks in every chapter, including insights that will delight even the most experienced developer. -- Provided by publisher.

Write code that scales across CPU registers, multi-core, and machine clusters Key Features Explore concurrent programming in C++ Identify memory management problems Use SIMD and STL containers for performance improvement Book Description C++ is a highly portable language and can be used to write both large-scale applications and performance-critical code. It has evolved over the last few years to become a modern and expressive language. This book will guide you through optimizing the performance of your C++ apps by allowing them to run faster and consume fewer resources on the device they're running on without compromising the readability of your code base. The book begins by helping you measure and identify bottlenecks in a C++ code base. It then moves on by teaching you how to use modern C++ constructs and techniques. You'll see how this affects the way you write code. Next, you'll see the importance of data structure optimization and memory management, and how it can be used efficiently with respect to CPU caches. After that, you'll see how STL algorithm and composable Range V3 should be used to both achieve faster execution and more readable code, followed by how to use STL containers and how to write your own specialized iterators. Moving on, you’ll get hands-on experience in making use of modern C++ metaprogramming and reflection to reduce boilerplate code as well as in working with proxy objects to perform optimizations under the hood. After that, you’ll learn concurrent programming and understand lock-free data structures. The book ends with an overview of parallel algorithms using STL execution policies, Boost Compute, and OpenCL to utilize both the CPU and the GPU. What you will learn Benefits of modern C++ constructs and techniques Identify hardware bottlenecks, such as CPU cache misses, to boost performance Write specialized data structures for performance-critical code Use modern metaprogramming techniques to reduce runtime calculations Achieve efficient memory management using custom memory allocators Reduce boilerplate code using reflection techniques Reap the benefits of lock-free concurrent programming Perform under-the--hood optimizations with preserved readability using proxy objects Gain insights into subtle optimizations used by STL algorithms Utilize the Range V3 library for expressive C++ code Parallelize your code over CPU and GPU, without compromising readability Who this book is for If you're a C++ developer looking to improve the speed of your code or simply wanting to take your skills up to the next level, then this book is perfect for you.

Get to grips with modern software demands by learning the effective uses of Rust's powerful memory safety. Key Features Learn and improve the sequential performance characteristics of your software Understand the use of operating system processes in a high-scale concurrent system Learn of the various coordination methods available in the Standard library Book Description Most programming languages can really complicate things, especially with regard to unsafe memory access. The burden on you, the programmer, lies across two domains: understanding the modern machine and your language's pain-points. This book will teach you to how to manage program performance on modern machines and build fast, memory-safe, and concurrent software in Rust. It starts with the fundamentals of Rust and discusses machine architecture concepts. You will be taken through ways to measure and improve the performance of Rust code systematically and how to write collections with confidence. You will learn about the Sync and Send traits applied to threads, and coordinate thread execution with locks, atomic primitives, data-parallelism, and more. The book will show you how to efficiently embed Rust in C++ code and explore the functionalities of various crates for multithreaded applications. It explores implementations in depth. You will know how a mutex works and build several yourself. You will master radically different approaches that exist in the ecosystem for structuring and managing high-scale systems. By the end of the book, you will feel comfortable with designing safe, consistent, parallel, and high-performance applications in Rust. What you will learn Probe your programs for performance and accuracy issues Create your own threading and multi-processing environment in Rust Use coarse locks from Rust’s Standard library Solve common synchronization problems or avoid synchronization using atomic programming Build lock-free/wait-free structures in Rust and understand their implementations in the crates ecosystem Leverage Rust’s memory model and type system to build safety properties into your parallel programs Understand the new features of the Rust programming language to ease the writing of parallel programs Who this book is for This book is aimed at software engineers with a basic understanding of Rust who want to exploit the parallel and concurrent nature of modern computing environments, safely.

This book presents a large collection of exercises for learning to program in C++. A study plan for learning C++ based on a collection of video lectures and supplemental reading is also provided.

"This book should be on every C++ programmer's desk. It's clear, concise, and valuable." Rob Green, Bowling Green State University This bestseller has been updated and revised to cover all the latest changes to C++ 14 and 17! C++ Concurrency in Action, Second Edition teaches you everything you need to write robust and elegant multithreaded applications in C++17. You choose C++ when your applications need to run fast. Well-designed concurrency makes them go even faster. C++ 17 delivers strong support for the multithreaded, multiprocessor programming required for fast graphic processing, machine learning, and other performance-sensitive tasks. This exceptional book/course unpacks the features, patterns, and best practices of production-grade C++ concurrency. C++ Concurrency in Action, Second Edition is the definitive guide to writing elegant multithreaded applications in C++. Updated for C++ 17, it carefully addresses every aspect of concurrent development, from starting new threads to designing fully functional multithreaded algorithms and data structures. Concurrency master Anthony Williams presents examples and practical tasks in every chapter, including insights that will delight even the most experienced developer. Inside: Full coverage of new C++ 17 features Starting and managing threads Synchronizing concurrent operations Designing concurrent code Debugging multithreaded applications This book/course is written for intermediate C and C++ developers. No prior experience with concurrency required. Anthony Williams has been an active member of the BSI C++ Panel since 2001 and is the developer of the just::thread Pro extensions to the C++ 11 thread library. A thorough presentation of C++ concurrency capabilities. Maurizio Tomasi, University of Milan Highly recommended for programmers who want to further their knowledge of the latest C++ standard. Frédéric Flayol, 4Pro Web C++ The guide contains snippets for everyday use in your own projects and to help take your concurrency C++ skills from the Padawan to the Jedi level. Jura Shikin, IVI Technologies NARRATED BY LISA FARINA AND DIANA GARDINER.

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.

Create robust and scalable applications along with responsive UI using concurrency and the multi-threading infrastructure in .NET and C# About This Book Learn to combine your asynchronous operations with Task Parallel Library Master C#'s asynchronous infrastructure and use asynchronous APIs effectively to achieve optimal responsiveness of the application An easy-to-follow, example-based guide that helps you to build scalable applications using concurrency in C# Who This Book Is For If you are a C# developer who wants to develop modern applications in C# and wants to overcome problems by using asynchronous APIs and standard patterns, then this book is ideal for you. Reasonable development knowledge, an understanding of core elements and applications related to the .Net platform, and also the fundamentals of concurrency is assumed. What You Will Learn Apply general multithreading concepts to your application's design Leverage lock-free concurrency and learn about its pros and cons to achieve efficient synchronization between user threads Combine your asynchronous operations with Task Parallel Library Make your code easier with C#'s asynchrony support Use common concurrent collections and programming patterns Write scalable and robust server-side asynchronous code Create fast and responsible client applications Avoid common problems and troubleshoot your multi-threaded and asynchronous applications In Detail Starting with the traditional approach to concurrency, you will learn how to write multithreaded concurrent programs and compose ways that won't require locking. You will explore the concepts of parallelism granularity, and fine-grained and coarse-grained parallel tasks by choosing a concurrent program structure and parallelizing the workload optimally. You will also learn how to use task parallel library, cancellations, timeouts, and how to handle errors. You will know how to choose the appropriate data structure for a specific parallel algorithm to achieve scalability and performance. Further, you'll learn about server scalability, asynchronous I/O, and thread pools,

and write responsive traditional Windows and Windows Store applications. By the end of the book, you will be able to diagnose and resolve typical problems that could happen in multithreaded applications. Style and approach An easy-to-follow, example-based guide that will walk you through the core principles of concurrency and multithreading using C#.

Templates are among the most powerful features of C++, but they remain misunderstood and underutilized, even as the C++ language and development community have advanced. In C++ Templates, Second Edition, three pioneering C++ experts show why, when, and how to use modern templates to build software that's cleaner, faster, more efficient, and easier to maintain. Now extensively updated for the C++11, C++14, and C++17 standards, this new edition presents state-of-the-art techniques for a wider spectrum of applications. The authors provide authoritative explanations of all new language features that either improve templates or interact with them, including variadic templates, generic lambdas, class template argument deduction, compile-time if, forwarding references, and user-defined literals. They also deeply delve into fundamental language concepts (like value categories) and fully cover all standard type traits. The book starts with an insightful tutorial on basic concepts and relevant language features. The remainder of the book serves as a comprehensive reference, focusing first on language details and then on coding techniques, advanced applications, and sophisticated idioms. Throughout, examples clearly illustrate abstract concepts and demonstrate best practices for exploiting all that C++ templates can do. Understand exactly how templates behave, and avoid common pitfalls Use templates to write more efficient, flexible, and maintainable software Master today's most effective idioms and techniques Reuse source code without compromising performance or safety Benefit from utilities for generic programming in the C++ Standard Library Preview the upcoming concepts feature The companion website, tmplbook.com, contains sample code and additional updates.

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

Advanced Metaprogramming in Classic C++ aims to be both an introduction and a reference to C++ template metaprogramming (TMP); TMP is presented in the book as a set of techniques that will bring a new style in C++ and make code exceptionally clear and efficient. The book deals with language aspects, design patterns, examples and applications (seen as case studies). Special emphasis is put on small reusable techniques that will improve the quality of daily work. What makes the book exceptional is the level of understanding of the concepts involved imparted by the author. This is not just a rote overview of metaprogramming. You will truly understand difficult topics like static assertions, how to write metafunctions, overload resolution, lambda expressions, and many others. More than that, you will work through them with practical examples guided by the author's frank explanations. This book requires you to think and to learn and to understand the language so that you can program at a higher level.

Master multithreading and concurrent processing with C++ About This Book Delve into the fundamentals of multithreading and concurrency and find out how to implement them Explore atomic operations to optimize code performance Apply concurrency to both distributed computing and GPGPU processing Who This Book Is For This book is for intermediate C++ developers who wish to extend their knowledge of multithreading and concurrent processing. You should have basic experience with multithreading and be comfortable using C++ development toolchains on the command line. What You Will Learn Deep dive into the details of the how various operating systems currently implement multithreading Choose the best multithreading APIs when designing a new application Explore the use of mutexes, spin-locks, and other synchronization concepts and see how to safely pass data between threads Understand the level of API support provided by various C++ toolchains Resolve common issues in multithreaded code and recognize common pitfalls using tools such as Memcheck, CacheGrind, DRD, Helgrind, and more Discover the nature of atomic operations and understand how they can be useful in optimizing code Implement a multithreaded application in a distributed computing environment Design a C++-based GPGPU application that employs multithreading In Detail Multithreaded applications execute multiple threads in a single processor environment, allowing developers achieve concurrency. This book will teach you the finer points of multithreading and concurrency concepts and how to apply them efficiently in C++. Divided into three modules, we start with a brief introduction to the fundamentals of multithreading and concurrency concepts. We then take an in-depth look at how these concepts work at the hardware-level as well as how both operating systems and frameworks use these low-level functions. In the next module, you will learn about the native multithreading and concurrency support available in C++ since the 2011 revision, synchronization and communication between threads, debugging concurrent C++ applications, and the best programming practices in C++. In the final module, you will learn about atomic operations before moving on to apply concurrency to distributed and GPGPU-based processing. The comprehensive coverage of essential multithreading concepts means you will be able to efficiently apply multithreading concepts while coding in C++. Style and approach This book is filled with examples that will help you become a master at writing robust concurrent and parallel applications in C++.

An introduction to programming by the inventor of C++, Programming prepares students for programming in the real world. This book assumes that they aim eventually to write non-trivial programs, whether for work in software development or in some other technical field. It explains fundamental concepts and techniques in greater depth than traditional introductions. This approach gives students a solid foundation for writing useful, correct, maintainable, and efficient code. This book is an introduction to programming in general, including object-oriented programming and generic programming. It is also a solid introduction to the C++ programming language, one of the most widely used languages for real-world software. It presents modern C++ programming techniques from the start, introducing the C++ standard library to simplify programming tasks.

All the new language and library features of C++17 (for those who know the previous versions of C++). C++17 is the next evolution in modern C++ programming, which is already now supported by the latest version of gcc, clang, and Visual C++. Although it is not as big a step as C++11, it contains a large number of small and valuable language and library features, which will change the way we program in C++. As usual, not everything is self-explanatory, combining new features gives even more power, and there are hidden traps. This book presents all the new language and library features of C++17. It covers the motivation and context of each new feature with examples and background information. The focus is on how these features impact day-to-day programming, what it means to combine them, and how to benefit from this in practice.

Software -- Operating Systems.

A comprehensive guide to help aspiring and professional C++ developers elevate the performance of their apps by allowing them to run faster and consume fewer resources Key FeaturesUpdated to C++20 with completely revised code and more content on error handling, benchmarking, memory allocators, and concurrent programmingExplore the latest C++20 features including concepts, ranges, and coroutinesUtilize C++ constructs and techniques to carry out effective data structure optimization and memory managementBook Description C++ High Performance, Second Edition guides you through optimizing the performance of your C++ apps. This allows them to run faster and consume fewer resources on the device they're running on without compromising the readability of your codebase. The book begins by introducing the C++ language and some of its modern concepts in brief. Once you are familiar with the fundamentals, you will be ready to measure, identify, and eradicate bottlenecks in your C++ codebase. By following this process, you will gradually improve your style of writing code. The book then explores data structure optimization, memory management, and how it can be used efficiently concerning CPU caches. After laying the foundation, the book trains you to leverage algorithms, ranges, and containers from the standard library to achieve faster execution, write readable code, and use customized iterators. It provides hands-on examples of C++ metaprogramming, coroutines, reflection to reduce boilerplate code, proxy objects to perform optimizations under the hood, concurrent programming, and lock-free data structures. The book concludes with an overview of parallel algorithms. By the end of this book, you will have the ability to use every tool as needed to boost the efficiency of your C++ projects. What you will learnWrite specialized data structures for performance-critical codeUse modern metaprogramming techniques to reduce runtime calculationsAchieve efficient memory management using custom memory allocatorsReduce boilerplate code using reflection techniquesReap the benefits of lock-free concurrent programmingGain insights into subtle optimizations used by standard library algorithmsCompose algorithms using ranges libraryDevelop the ability to apply metaprogramming aspects such as constexpr, constraints, and conceptsImplement lazy generators and asynchronous tasks using C++20 coroutinesWho this book is for If you're a C++ developer looking to improve the efficiency of your code or just keen to upgrade your skills to the next level, this book is for you.