

Clean Code It Colloge

Have you ever felt frustrated working with someone else's code? Difficult-to-maintain source code is a big problem in software development today, leading to costly delays and defects. Be part of the solution. With this practical book, you'll learn 10 easy-to-follow guidelines for delivering C# software that's easy to maintain and adapt. These guidelines have been derived from analyzing hundreds of real-world systems. Written by consultants from the Software Improvement Group (SIG), this book provides clear and concise explanations, with advice for turning the guidelines into practice. Examples for this edition are written in C#, while our companion Java book provides clear examples in that language. Write short units of code: limit the length of methods and constructors Write simple units of code: limit the number of branch points per method Write code once, rather than risk copying buggy code Keep unit interfaces small by extracting parameters into objects Separate concerns to avoid building large classes Couple architecture components loosely Balance the number and size of top-level components in your code Keep your codebases as small as possible Automate tests for your codebase Write clean code, avoiding "code smells" that indicate deeper problems

Widely considered one of the best practical guides to programming, Steve McConnell's original CODE COMPLETE has been helping developers write better software for more than a decade. Now this classic book has been fully updated and revised with leading-edge practices—and hundreds of new code samples—illustrating the art and science of software construction. Capturing the body of knowledge available from research, academia, and everyday commercial practice, McConnell synthesizes the most effective techniques and practices that you need to know into clear, pragmatic guidance. No matter what your experience level, development environment, or project size, this book will inform and stimulate your thinking—and help you build the highest quality code. Discover the timeless techniques and strategies that help you: Design for minimum complexity and maximum creativity Reap the benefits of collaborative development Apply defensive programming techniques to reduce and flush out errors Exploit opportunities to refactor—or evolve—code, and do it safely Use construction practices that are right-weight for your project Debug problems quickly and effectively Resolve critical construction issues early and correctly Build quality into the beginning, middle, and end of your project

This comprehensive career guide helps readers take a close look at coding as a career path and gives them a long, actionable list for turning their interests into a career across a variety of fields. Whether a student is interested in games, engineering, design, or systems administration, each career path comes with a detailed list of resources and first-person accounts from professionals in the field. This guide is all a coding enthusiast needs to get started planning and building a career, all without having to worry about student loans. Sustainable Communities Design Handbook: Green Engineering, Architecture, and Technology, Second Edition, brings together the major players responsible for sustainable development at both community and metropolitan scales. The book aims to explain and demonstrate the practice, planning, design, building and managing of the engineering, architectural and economic development of cities and communities to meet sustainable development objectives. Offering a holistic approach to creating sustainable communities, the book includes a 40 percent increase in new methods and technology over the last edition, and 50 percent more case studies from around the world to illustrate how common sustainability problems are solved. As the concept and practices of a sustainable built environment have evolved over the years, it is increasingly recognized that the scope should be expanded beyond individual buildings to the community scale. Written by an international team of engineers, architects, and environmental experts this second edition includes new HVAC technologies for heating and cooling, energy effect technologies for lighting, and new construction materials which improve heating and cooling efficiencies. This new edition will also include critical updates on international codes: LEED, BREEAM, and Green Globes. Explains the most cutting-edge green technologies and methods for use in built communities Provides a common approach in using natural resources when building and designing Covers compliance with various international codes, methods and legal frameworks

Beyond the Basic Stuff with Python

Back to Basics

Recreation

A Guide to Crafting Better Code for Rubyists

Disciplines, Standards, and Ethics

Code Complete

New Directions for Institutional Research, Number 169

Looks at the principles and clean code, includes case studies showcasing the practices of writing clean code, and contains a list of heuristics and "smells" accumulated from the process of writing clean code.

Includes Part 1, Number 1 & 2: Books and Pamphlets, Including Serials and Contributions to Periodicals (January - December)

BRIDGE THE GAP BETWEEN NOVICE AND PROFESSIONAL You've completed a basic Python programming tutorial or finished Al Sweigart's bestseller, Automate the Boring Stuff with Python. What's the next step toward becoming a capable, confident software developer? Welcome to Beyond the Basic Stuff with Python. More than a mere collection of advanced syntax and masterful tips for writing clean code, you'll learn how to advance your Python programming skills by using the command line and other professional tools like code formatters, type checkers, linters, and version control. Sweigart takes you through best practices for setting up your development environment, naming variables, and improving readability, then tackles documentation, organization and performance measurement, as well as object-oriented design and the Big-O algorithm analysis commonly used in coding interviews. The skills you learn will be applicable to just about any language. You'll learn: • Coding style, and how to use Python's Black auto-formatting tool for cleaner code • Common sources of bugs, and how to detect them with static analyzers • How to structure the files in your code projects with the Cookiecutter template • Functional programming techniques like lambda and higher-order functions • How to profile the speed of your code with Python's built-in timeit and cProfile modules • The computer science behind Big-O algorithm analysis • How to make your comments and docstrings informative, and how often to write them • How to create classes in object-oriented programming, and why they're used to organize code Toward the end of the book you'll read a detailed source-code breakdown of two classic command-line games, the Tower of Hanoi (a logic puzzle) and Four-in-a-Row (a two-player tile-dropping game), and a breakdown of how their code follows the book's best practices. You'll test your skills by implementing the program yourself. Of course, no single book can make you a professional software developer. But Beyond the Basic Stuff with Python will get you further down that path and make you a better programmer, as you learn to write readable code that's easy to debug and perfectly Pythonic Requirements: Covers Python 3.6 and higher

Practical Software Architecture Solutions from the Legendary Robert C. Martin ("Uncle Bob") By applying universal rules of software architecture, you can dramatically improve developer productivity throughout the life of any software system. Now, building upon the success of his best-selling books Clean Code and The Clean Coder, legendary software craftsman Robert C. Martin ("Uncle Bob") reveals those rules and helps you apply them. Martin's Clean Architecture doesn't merely present options. Drawing on over a half-century of experience in software environments of every imaginable type, Martin tells you what choices to make and why they are critical to your success. As you've come to expect from Uncle Bob, this book is packed with direct, no-nonsense solutions for the real challenges you'll face—the ones that will make or break your projects. Learn what software architects need to achieve—and how to achieve it—Master essential software design principles for addressing function, component separation, and data management See how programming paradigms impose discipline by restricting what developers can do Understand what's critically important and what's merely a "detail" Implement optimal, high-level structures for web, database, thick-client, console, and embedded applications Define appropriate boundaries and layers, and organize components and services See why designs and architectures go wrong, and how to prevent (or fix) these failures Clean Architecture is essential reading for every current or aspiring software architect, systems analyst, system designer, and software manager—and for every programmer who must execute someone else's designs. Register your product for convenient access to downloads, updates, and/or corrections as they become available.

Encyclopedia of School Crime and Violence

Agile Processes in Software Engineering and Extreme Programming

OOP Concepts Bopster

Catalog of Copyright Entries, Third Series

Complete Book of Colleges, 2005

30 Core Guidelines for Writing Clean, Safe, and Fast Code

Refactor your legacy C# code base and improve application performance by applying best practices

Improve your coding skills and learn how to write readable code. Rather than teach basic programming, this book presumes that readers understand the fundamentals, and offers time-honed best practices for style, design, documenting, testing, refactoring, and more. Taking an informal, conversational tone, author Michael Stueben offers programming stories, anecdotes, observations, advice, tricks, examples, and challenges based on his 38 years experience writing code and teaching programming classes. Trying to teach style to beginners is notoriously difficult and often unproductive. This book offers solutions and many examples to back up his ideas. Good Habits for Great Coding distills Stueben's three decades of analyzing his own mistakes, analyzing student mistakes, searching for problems that teach lessons, and searching for simple examples to illustrate complex ideas. Having found that most learn by trying out challenging problems, and reflecting on them, each chapter includes quizzes and problems. The final chapter introduces dynamic programming to reduce complex problems to subcases, and illustrates many concepts discussed in the book. Code samples are provided in Python and designed to be understandable by readers familiar with any modern programming language. At the end of this book, you will have acquired a lifetime of good coding advice, the lessons the author wishes he had learned when he was a novice. What You'll Learn Create readable code through examples of good and bad style Write difficult algorithms by comparing your code to the author's code Derive and code difficult algorithms using dynamic programming Understand the psychology of the coding process Who This Book Is For Students or novice programmers who have taken a beginning programming course and understand coding basics. Teachers will appreciate the author's road-tested ideas that they may apply to their own teaching.

If you have a passion for programming and want to be a better programmer, then this is the right source. This handbook contains useful information about the techniques and approaches that help individuals boost not only their programming career but also their well-being. The author of this book presents sound advice, which when you follow, you can find it easy to understand coding using any types of programming languages. With this book, you can understand the structure of the database, identify programming languages used by many programmers in the world, and various factors you should consider while choosing the language. Becoming the best programmer depends on many factors apart from what you learn in your college or university. Most colleges focus mainly on the theoretical part of programming than on practical part. You need to continue doing programming every day to obtain new skills since programming evolves almost every time. This book contains nine chapters that span the range of the life of a good software developer, including dealing with code, improving performance, and learning the trade with no bias in language. Reading this book will enable you to find valuable tips about becoming the best programmer, regardless of what you are at the moment. In fact, the book is suitable for all types of programmers like a hobbyist, a seasonal developer, or a neophyte professional. Lastly, you will be able to learn about testing, debugging, coping with complexity, finding challenges, avoiding the problem, solving the problem effectively, using the right tools, and working with your team members well. The author believes that the first step to improving your programming skills is training your mind to think more logically and analytically. You can achieve this by associating with the right people, people who are willing to improve your programming skills. Read this book and see its positive impacts on your programming career.

The post-graduation outcomes of college students are being more widely used as key metrics to demonstrate institutional effectiveness to both external agencies and internal stakeholders. Institutional research offices play an integral role in these data collection efforts. However, underlying challenges exist regarding obtaining an adequate amount of survey responses and salary or earnings information. This volume focuses on the first-destination outcomes (e.g., earnings, employment, graduate/ professional school enrollment) of college graduates while recognizing that other outcomes are also relevant across institutional settings. Through the use of current research, case studies, and best practices, each chapter highlights how postgraduate outcomes information is collected and used across the higher education spectrum. In this volume readers will learn: the internal and external demands for these data, the strengths and challenges of their data, and how to best communicate these data to various constituents. This is the 169th volume of this Jossey-Bass quarterly report series. Timely and comprehensive, New Directions for Institutional Research provides planners and administrators in all types of academic institutions with guidelines in such areas as resource coordination, information analysis, program evaluation, and institutional management.

Coders Journal - Eat Sleep Write Code Repeat: 6 X 19 X 190 Page College Ruled

Postgraduate Outcomes of College Students

Clean Ruby

Cool Careers Without College for People Who Love Coding

Good Habits for Great Coding

How to Create Digital Portfolios to Apply for College and Jobs

In the journey to all things Agile, Uncle Bob has been there, done that, and has the both the t-shirt and the scars to show for it. This delightful book is part history, part personal stories, and all wisdom. If you want to understand what Agile is and how it came to be, this is the book for you. —Grady Booch “Bob’s frustration colors every sentence of Clean Agile, but it’s a justified frustration. What is in the world of Agile development is nothing compared to what could be. This book is Bob’s perspective on what to focus on to get to that ‘what could be.’ And he’s been there, so it’s worth listening.” —Kent Beck “It’s good to read Uncle Bob’s take on Agile. Whether just beginning, or a seasoned Agilista, you would do well to read this book. I agree with almost all of it. It’s just some of the parts make me realize my own shortcomings, dammit. It made me double-check our code coverage (85.09%).” —Von Kern Nearly twenty years after the Agile Manifesto was first presented, the legendary Robert C. Martin (“Uncle Bob”) reintroduces Agile values and principles for a new generation-programmers and nonprogrammers alike. Martin, author of Clean Code and other highly influential software development guides, was there at Agile’s founding. Now, in Clean Agile: Back to Basics, he strips away misunderstandings and distractions that over the years have made it harder to use Agile than was originally intended. Martin describes what Agile is in no uncertain terms: a small discipline that helps small teams manage small projects . . . with huge implications because every big project is comprised of many small projects. Drawing on his fifty years’ experience with projects of every conceivable type, he shows how Agile can help you bring true professionalism to software development. Get back to the basics—what Agile is, was, and should always be Understand the origins, and proper practice, of SCRUM Master essential business-facing Agile practices, from small releases and acceptance tests to whole-team communication Explore Agile team members’ relationships with each other, and with their product Rediscover indispensable Agile technical practices: TDD, refactoring, simple design, and pair programming Understand the central roles values and craftsmanship play in your Agile team’s success If you want Agile’s true benefits, there are no shortcuts: You need to do Agile right. Clean Agile: Back to Basics will show you how, whether you’re a developer, tester, manager, project manager, or customer. Register your book for convenient access to downloads, updates, and/or corrections as they become available. See inside book for details.

In Clean Craftsmanship, the legendary Robert C. Martin (“Uncle Bob”) has written every programmer’s definitive guide to working well. Martin brings together the disciplines, standards, and ethics you need to deliver robust, effective code quickly and productively, and be proud of all the software you write — every single day. Martin, the best-selling author of The Clean Coder, begins with a pragmatic, technical, and prescriptive guide to five foundational disciplines of software craftsmanship: test-driven development, refactoring, simple design, collaborative programming (pairing), and acceptance tests. Next, he moves up to standards — outlining the baseline expectations the world has of software developers, illuminating how those often differ from their own perspectives, and helping you repair the mismatch. Finally, he turns to the ethics of the programming profession, describing ten fundamental promises all software developers should make to their colleagues, their users, and above all, themselves. With Martin’s guidance and advice, you can consistently write code that builds trust instead of undermining it — trust among your users and throughout a society that depends on software for its very survival. Have you ever felt frustrated working with someone else’s code? Difficult-to-maintain source code is a big problem in software development today, leading to costly delays and defects. Be part of the solution. With this practical book, you’ll learn 10 easy-to-follow guidelines for delivering Java software that’s easy to maintain and adapt. These guidelines have been derived from analyzing hundreds of real-world systems. Written by consultants from the Software Improvement Group (SIG), this book provides clear and concise explanations, with advice for turning the guidelines into practice. Examples for this edition are written in Java, while our companion C# book provides workable examples in that language. Write short units of code: limit the length of methods and constructors Write simple units of code: limit the number of branch points per method Write code once, rather than risk copying buggy code Keep unit interfaces small by extracting parameters into objects Separate concerns to avoid building large classes Couple architecture components loosely Balance the number and size of top-level components in your code Keep your codebase as small as possible Automate tests for your codebase Write clean code, avoiding “code smells” that indicate deeper problems

In Clean Craftsmanship, the legendary Robert C. Martin (“Uncle Bob”) has written every programmer’s definitive guide to working well. Martin brings together the disciplines, standards, and ethics you need to deliver robust, effective code quickly and productively, and be proud of all the software you write — every single day. Martin, the best-selling author of The Clean Coder, begins with a pragmatic, technical, and prescriptive guide to five foundational disciplines of software craftsmanship: test-driven development, refactoring, simple design, collaborative programming (pairing), and acceptance tests. Next, he moves up to standards — outlining the baseline expectations the world has of software developers, illuminating how those often differ from their own perspectives, and helping you repair the mismatch. Finally, he turns to the ethics of the programming profession, describing ten fundamental promises all software developers should make to their colleagues, their users, and above all, themselves. With Martin’s guidance and advice, you can consistently write code that builds trust instead of undermining it — trust among your users and throughout a society that depends on software for its very survival. Have you ever felt frustrated working with someone else’s code? Difficult-to-maintain source code is a big problem in software development today, leading to costly delays and defects. Be part of the solution. With this practical book, you’ll learn 10 easy-to-follow guidelines for delivering Java software that’s easy to maintain and adapt. These guidelines have been derived from analyzing hundreds of real-world systems. Written by consultants from the Software Improvement Group (SIG), this book provides clear and concise explanations, with advice for turning the guidelines into practice. Examples for this edition are written in Java, while our companion C# book provides workable examples in that language. Write short units of code: limit the length of methods and constructors Write simple units of code: limit the number of branch points per method Write code once, rather than risk copying buggy code Keep unit interfaces small by extracting parameters into objects Separate concerns to avoid building large classes Couple architecture components loosely Balance the number and size of top-level components in your code Keep your codebase as small as possible Automate tests for your codebase Write clean code, avoiding “code smells” that indicate deeper problems

In Clean Craftsmanship, the legendary Robert C. Martin (“Uncle Bob”) has written every programmer’s definitive guide to working well. Martin brings together the disciplines, standards, and ethics you need to deliver robust, effective code quickly and productively, and be proud of all the software you write — every single day. Martin, the best-selling author of The Clean Coder, begins with a pragmatic, technical, and prescriptive guide to five foundational disciplines of software craftsmanship: test-driven development, refactoring, simple design, collaborative programming (pairing), and acceptance tests. Next, he moves up to standards — outlining the baseline expectations the world has of software developers, illuminating how those often differ from their own perspectives, and helping you repair the mismatch. Finally, he turns to the ethics of the programming profession, describing ten fundamental promises all software developers should make to their colleagues, their users, and above all, themselves. With Martin’s guidance and advice, you can consistently write code that builds trust instead of undermining it — trust among your users and throughout a society that depends on software for its very survival. Have you ever felt frustrated working with someone else’s code? Difficult-to-maintain source code is a big problem in software development today, leading to costly delays and defects. Be part of the solution. With this practical book, you’ll learn 10 easy-to-follow guidelines for delivering Java software that’s easy to maintain and adapt. These guidelines have been derived from analyzing hundreds of real-world systems. Written by consultants from the Software Improvement Group (SIG), this book provides clear and concise explanations, with advice for turning the guidelines into practice. Examples for this edition are written in Java, while our companion C# book provides workable examples in that language. Write short units of code: limit the length of methods and constructors Write simple units of code: limit the number of branch points per method Write code once, rather than risk copying buggy code Keep unit interfaces small by extracting parameters into objects Separate concerns to avoid building large classes Couple architecture components loosely Balance the number and size of top-level components in your code Keep your codebase as small as possible Automate tests for your codebase Write clean code, avoiding “code smells” that indicate deeper problems

In Clean Craftsmanship, the legendary Robert C. Martin (“Uncle Bob”) has written every programmer’s definitive guide to working well. Martin brings together the disciplines, standards, and ethics you need to deliver robust, effective code quickly and productively, and be proud of all the software you write — every single day. Martin, the best-selling author of The Clean Coder, begins with a pragmatic, technical, and prescriptive guide to five foundational disciplines of software craftsmanship: test-driven development, refactoring, simple design, collaborative programming (pairing), and acceptance tests. Next, he moves up to standards — outlining the baseline expectations the world has of software developers, illuminating how those often differ from their own perspectives, and helping you repair the mismatch. Finally, he turns to the ethics of the programming profession, describing ten fundamental promises all software developers should make to their colleagues, their users, and above all, themselves. With Martin’s guidance and advice, you can consistently write code that builds trust instead of undermining it — trust among your users and throughout a society that depends on software for its very survival. Have you ever felt frustrated working with someone else’s code? Difficult-to-maintain source code is a big problem in software development today, leading to costly delays and defects. Be part of the solution. With this practical book, you’ll learn 10 easy-to-follow guidelines for delivering Java software that’s easy to maintain and adapt. These guidelines have been derived from analyzing hundreds of real-world systems. Written by consultants from the Software Improvement Group (SIG), this book provides clear and concise explanations, with advice for turning the guidelines into practice. Examples for this edition are written in Java, while our companion C# book provides workable examples in that language. Write short units of code: limit the length of methods and constructors Write simple units of code: limit the number of branch points per method Write code once, rather than risk copying buggy code Keep unit interfaces small by extracting parameters into objects Separate concerns to avoid building large classes Couple architecture components loosely Balance the number and size of top-level components in your code Keep your codebase as small as possible Automate tests for your codebase Write clean code, avoiding “code smells” that indicate deeper problems

In Clean Craftsmanship, the legendary Robert C. Martin (“Uncle Bob”) has written every programmer’s definitive guide to working well. Martin brings together the disciplines, standards, and ethics you need to deliver robust, effective code quickly and productively, and be proud of all the software you write — every single day. Martin, the best-selling author of The Clean Coder, begins with a pragmatic, technical, and prescriptive guide to five foundational disciplines of software craftsmanship: test-driven development, refactoring, simple design, collaborative programming (pairing), and acceptance tests. Next, he moves up to standards — outlining the baseline expectations the world has of software developers, illuminating how those often differ from their own perspectives, and helping you repair the mismatch. Finally, he turns to the ethics of the programming profession, describing ten fundamental promises all software developers should make to their colleagues, their users, and above all, themselves. With Martin’s guidance and advice, you can consistently write code that builds trust instead of undermining it — trust among your users and throughout a society that depends on software for its very survival. Have you ever felt frustrated working with someone else’s code? Difficult-to-maintain source code is a big problem in software development today, leading to costly delays and defects. Be part of the solution. With this practical book, you’ll learn 10 easy-to-follow guidelines for delivering Java software that’s easy to maintain and adapt. These guidelines have been derived from analyzing hundreds of real-world systems. Written by consultants from the Software Improvement Group (SIG), this book provides clear and concise explanations, with advice for turning the guidelines into practice. Examples for this edition are written in Java, while our companion C# book provides workable examples in that language. Write short units of code: limit the length of methods and constructors Write simple units of code: limit the number of branch points per method Write code once, rather than risk copying buggy code Keep unit interfaces small by extracting parameters into objects Separate concerns to avoid building large classes Couple architecture components loosely Balance the number and size of top-level components in your code Keep your codebase as small as possible Automate tests for your codebase Write clean code, avoiding “code smells” that indicate deeper problems

Vois. 9-10 include proceedings of the 8th-11th annual meeting of the American Football Coaches Association and of the 3d-6th annual meeting of the National Association of the Basketball Coaches of the United States. From a former college basketball player and shoe rep Nike, this explosive insider’s account into the dark underworld of college basketball exposes the corrupt and racist systems that exploit young athletes and offers a new way forward For Merl Code, basketball was life. In college he played point guard for Clemson before turning pro. Later, with a keen eye for talent and vested interest in the basketball community, he pivoted to marketing, where he was eager to build relation network expanded and his responsibilities as a shoe rep grew, he found himself thrust into the dark underbelly of profit-driven college basketball programs, and the storied coaches who led them. He realized that the NCAA’s amateurism rules, which prohibit athletes from receiving compensation until they go pro, were exploiting young athletes, and athletes of color in particular. Colleges and universities, he observed, were using corporate sponsors to circumvent the NCAA’s bylaws and the explosive story of college basketball’s dark reality. Code has engaged with the sport at every level—from the AAU circuit, to college recruiting battles, to winning NBA stars to sign sneaker deals. He’s been responsible for recruiting athletes like Zion Williamson, Anthony Davis, and Giannis Antetokounmpo to endorse the shoes of his employers. But Code has also seen the darker side of college basketball. He recounts how he became the fall guy for a bribery scandal involving were not charged and still walk the sidewalks. Highlighting stories of real athletes and their families and what’s at stake for them, Code pulls back the curtain on the systemic problem of using players for financial gain in college athletics. Propulsive, urgent, and eye-opening, Black Market exposes the truth to offer a more just way forward for both colleges and athletes. Discover the Beauty of Modern C++ Beautiful C++ presents the C++ Core Guidelines from a developer’s point of view with an emphasis on what benefits can be obtained from following the rules and what nightmares can result from ignoring them. For true geeks, it is an easy and entertaining read. For most software developers, it offers something new and useful. —Barne Stroustrup, inventor of C++ and co-editor of the C++ Core Guidelines Writing great C++ code needn’t be difficult. The C++ Core Guidelines are an exceptionally reliable, efficient, and well-performing. But the Guidelines are so jam-packed with excellent advice that it’s hard to know where to start. Start here, with Beautiful C++. Expert C++ programmers Guy Davidson and Kate Gregory identify 30 Core Guidelines you’ll find especially valuable and offer detailed practical knowledge for improving your C++ style. For easy reference, this book is structured to align closely with the official C++ Core Guidelines web sample code, illuminate proven ways to use both new and longstanding language features more successfully, and show how to write programs that are more robust and performant by default. Avoid bikeshedding: stop wasting valuable time on trivia Don’t hurt yourself by writing code that will cause problems later Know which legacy features to avoid and the modern features to use instead Use newer features properly, to get their benefits without creating new problems Default Guidelines with any modern C++ version: C++20, C++17, C++14, or C++11 There’s something here to improve virtually every program you write, design, or maintain. For ease of experimentation, all sample code is available on Compiler Explorer at https://godbolt.org/z/cg30-cho.O. Register your book for convenient access to downloads, updates, and/or corrections as they become available. See inside book for details.

A Comprehensive Beginner’s Guide to Learn the Realms of Clean Code From A-Z

Games Colleges Play

Improving Programming Skills with Examples in Python

The Playground

Composition Book

Develop reliable, maintainable, and robust JavaScript

Images that Fade and Persons that Endure

Learn how to make better decisions and write cleaner Ruby code. This book shows you how to avoid messy code that is hard to test and which cripples productivity. Author Carleton DLee shares hard-learned lessons gained from years of experience across numerous codebases both large and small. Each chapter covers the topics you need to know to make better decisions and optimize your productivity. Many books will tell you how to do something; this book will tell you why you should do it. Start writing code you love. What You Will Learn Build better classes to help promote code reuse Improve your decision making and make better, smarter code Identify bad code and fixed it Create quality names for all of your variables, classes, and modules Write better, concise classes Improve the quality of your methods Properly use modules Clarify your Boolean logic See when and how you refactor Improve your understanding of TDD and better tests Who This Book Is For This book is written for Ruby developers. There is no need to learn a new language or translate concepts to Ruby.

Get the most out of JavaScript for building web applications through a series of patterns, techniques, and case studies for clean coding Key Features! Write maintainable JS code using internal abstraction, well-written tests, and well-documented code Understand the agents of clean coding like SOLID principles, OOP, and functional programming Explore solutions to tackle common JavaScript challenges in building UIs, managing APIs, and writing stateful Book Description Building robust apps starts with creating clean code. In this book, you’ll explore techniques for doing this by learning everything from the basics of JavaScript through to the practices of clean code. You’ll write functional, intuitive, and maintainable code while also understanding how your code affects the end user and the wider community. The book starts with popular clean-coding principles such as SOLID, and the Law of Demeter (LoD), along with highlighting the enemies of writing clean code such as cargo culting and over-management. You’ll then delve into JavaScript, understanding the more complex aspects of the language. Next, you’ll create meaningful abstractions using design patterns, such as the Class Pattern and the Builder Pattern, and the Strategy Pattern. Beautiful C++ presents the C++ Core Guidelines from a developer’s point of view with an emphasis on what benefits can be obtained from following the rules and what nightmares can result from ignoring them. For true geeks, it is an easy and entertaining read. For most software developers, it offers something new and useful. —Barne Stroustrup, inventor of C++ and co-editor of the C++ Core Guidelines Writing great C++ code needn’t be difficult. The C++ Core Guidelines are an exceptionally reliable, efficient, and well-performing. But the Guidelines are so jam-packed with excellent advice that it’s hard to know where to start. Start here, with Beautiful C++. Expert C++ programmers Guy Davidson and Kate Gregory identify 30 Core Guidelines you’ll find especially valuable and offer detailed practical knowledge for improving your C++ style. For easy reference, this book is structured to align closely with the official C++ Core Guidelines web sample code, illuminate proven ways to use both new and longstanding language features more successfully, and show how to write programs that are more robust and performant by default. Avoid bikeshedding: stop wasting valuable time on trivia Don’t hurt yourself by writing code that will cause problems later Know which legacy features to avoid and the modern features to use instead Use newer features properly, to get their benefits without creating new problems Default Guidelines with any modern C++ version: C++20, C++17, C++14, or C++11 There’s something here to improve virtually every program you write, design, or maintain. For ease of experimentation, all sample code is available on Compiler Explorer at https://godbolt.org/z/cg30-cho.O. Register your book for convenient access to downloads, updates, and/or corrections as they become available. See inside book for details.

Encapsulating profiles of every four-year college in the United States, an updated guide provides detailed information on academic programs, admissions requirements, financial aid, services, housing, athletics, contact names, and more for 1,600 four-year colleges throughout the U.S. Original. 22,000 first printing.

This book provides a thorough compilation of the types, specific incidents, relevant agencies, theories, responses, and prevention programs relevant to crime and violence in schools and on campuses.

Lasting Screen Stars

A Philosophy of Software Design

Coding Roblox Games Made Easy

Building Maintainable Software, Java Edition

1960

A Craftsman’s Guide to Software Structure and Design

Clean Craftsmanship

Write code that’s clean, concise, and to the point: code that others will read with pleasure and reuse. Comparing your code to that of expert programmers is a great way to improve your coding skills. Get hands-on advice to level up your coding style through small and understandable examples that compare flawed code to an improved solution. Discover handy tips and tricks, as well as common bugs an experienced Java programmer needs to know. Make your way from a Java novice to a master craftsman. This book is a useful companion for anyone learning to write clean Java code. The authors introduce you to the fundamentals of becoming a software craftsman, by comparing pieces of problematic code with an improved version, to help you to develop a sense for clean code. This unique before-and-after approach teaches you to create clean Java code. Learn to keep your booleans in check, dodge formatting bugs, get rid of magic numbers, and use the right style of iteration. Write informative comments when needed, but avoid them when they are not. Improve the understandability of your code for others by following conventions and naming your objects accurately. Make your programs more robust with intelligent exception handling and learn to assert that everything works as expected using JUnit5 as your testing framework. Impress your peers with an elegant functional programming style and clear-cut object-oriented class design. Writing excellent code isn’t just about implementing the functionality. It’s about the small important details that make your code more readable, maintainable, flexible, robust, and faster. Java by Comparison teaches you to spot these details and trains you to become a better programmer. What You Need: You need a Java 8 compiler, a text editor, and a fresh mind.That’s it.

Get started with building your first game on the Roblox platform Endorsements “I read/worked through the book with my kids to build a game together and I highly recommend pre-teens, teens, and tweens to pick this up as their first book to coding games” —James W. Y III, Technology Integration Specialist at Old Bridge Township Public Schools “...a must-read, must-practice essential book for anyone getting started with building games on Roblox using Lua programming...” —Frederic Markus, President, Feerik Games (Ex-Ubisoft, Nintendo, Rockstar, Disney, LucasArts, and Epic Games) “...includes everything from Roblox Studio menus, the basics of the Lua scripting language, how to tie in real-world (or any world!) physics into your experience of marketing your game as well as some great ideas for where to go next.” —Jay Sebastian, Computer Scientist and Adjunct Lecturer in AI for Games and Simulation Book Description Roblox isn’t just popular; it’s incredibly popular, featuring more than 34 million active players per day. Any experience imaginable can be created on Roblox. Coding Roblox Games Made Easy, 2nd Edition, is a go-to guide for anyone at any age looking to get started with building a game on Roblox using Lua programming. In just about 300 pages, you’ll learn the basics of Lua programming, build two end-to-end games, add customizations to finally publish and monetize them. The bonus chapter “50 Cool Things to do in Roblox” is a perfect end to your learning journey with information nuggets presented with examples to save you time when coding, animating, building avatars, using Roblox and so much more. Join Zander, 19-year-old Roblox developer and programmer on this game-development journey and bring your ideas to life. What you will learn: • How to create Roblox Studio from scratch • Lua programming and rewards to make your first game • Move from lobby to battle programs, build avatars, locate weapons to fight • Character selection, countdown timers, locate escape items, assign rewards • Master the 3 Ms: Mechanics, Monetization, Marketing (and Menus) • 50 cool things to do in Roblox Who is this book for? This book is for anyone interested in learning the fundamentals of Lua programming and Roblox Studio and needs direction to build and share games. The book requires no prior knowledge of game development. This book contains the refereed proceedings of the 16th International Conference on Agile Software Development, XP 2015, held in Helsinki, Finland, in May 2015. While agile development has already become mainstream in industry, this field is still constantly evolving and continues to spur an enormous interest both in industry and academia. The XP conference series has always played, and continues to play, an important role in connecting the academic and practitioner communities, providing a forum for both formal and informal sharing and development of ideas, experiences, and opinions. The theme of XP 2015 “Delivering Value: Moving from Cyclic to Continuous Value Delivery” reflects the modern trend towards organizations that are simultaneously very efficient and flexible in software development and delivery. The 15 full and 7 short papers accepted for XP 2015 were selected from 44 submissions. All of the submitted papers went through a rigorous peer-review process. Additionally, 11 experience reports were selected from 45 proposals, and in each case the authors were shepherded by an experienced researcher.

Getting the most out of Python to improve your codebase Key Features Save maintenance costs by learning to fix your legacy codebase Learn the principles and techniques of refactoring Apply microservices to your legacy systems by implementing practical techniques Book Description Python is currently used in many different areas such as software construction, systems administration, and data processing. In all of these areas, experienced professionals can find examples of inefficiency, problems, and other perils, as a result of bad code. After reading this book, readers will understand these problems, and more importantly, how to correct them. The book begins by describing the basic elements of writing clean code and how it plays an important role in Python programming. You will learn about writing efficient and readable code using the Python standard library and best practices for software design. You will learn to implement the SOLID principles in Python and use decorators to improve your code. The book delves more deeply into object oriented programming in Python and shows you how to use objects with descriptors and generators. It will also show you the design principles of software testing and how to resolve software problems by implementing design patterns in your code. In the final chapter we break down a monolithic application to a microservice one, starting from the code as the basis for a solid platform. By the end of the book, you will be proficient in applying industry approved coding practices to design clean, sustainable and readable Python code. What you will learn Set up tools to effectively work in a development environment Explore how the magic methods of Python can help us write better code Examine the traits of Python to create advanced object-oriented design Understand removal of duplicated code using decorators and descriptors Effectively refactor code with the help of unit tests Learn to implement the SOLID principles in Python Who this book is for This book will appeal to team leads, software architects and senior software engineers who would like to work on their legacy systems to save cost and improve efficiency. A strong understanding of programming is assumed.

Black and Gray Binary Code, College Ruled School Notebook, 200 Pages, 7. 44 X9. 69

Clean Code in Python

Develop maintainable and efficient code

Sustainable Cities and Communities Design Handbook

Athletic Journal

Clean Agile

Clean Code in C#

We all live in a digital world of information technology. In this technology-driven world, computer software and applications are everywhere around us. Have you ever wondered how different applications and software work together efficiently? This book will be a comprehensive guide to make users understand how coding practices work in a few different computer programs and software. This book provides details about programming concepts, the history of programming, the importance of programming in daily life, how programming concepts are evolving in our daily life, and the best practices of using programming languages. We also discuss the best programming languages available in the world, different components of a program, how programs are improved in their efficiency, learning programming for a bright career choice and the future of programming. The programming is involved everywhere around us, even though many people are not aware of it. People work on digital platforms all the time, and they are using different kinds of programs. They do not have a deep understanding of programming concepts. This book is a comprehensive guide to help you understand how different programming concepts work together, and how different applications are made by using effective programming strategies. This book will be a comprehensive guide to understand all these concepts. This book will depict all the concepts of the programming languages from beginning to end. It will be a comprehensive and complete guide to understand the use of the best available sources to make an application that will work effectively and efficiently on the intended platform. Writing clean code is a skill that all computer programmers will want to master.

Develop your programming skills by exploring essential topics such as code reviews, implementing TDD and BDD, and designing APIs to overcome code inefficiency, redundancy, and other problems arising from bad code Key FeaturesWrite code that cleanly integrates with other systems while maintaining well-defined software boundariesUnderstand how coding principles and standards enhance software qualityLearn how to avoid common errors while implementing concurrency or threadingBook Description Traditionally associated with developing Windows desktop applications and games, C# is now used in a wide variety of domains, such as web and cloud apps, and has become increasingly popular for mobile development. Despite its extensive coding features, professionals experience problems related to efficiency, scalability, and maintainability because of bad code. Clean Code in C# will help you identify these problems and solve them using coding best practices. The book starts with a comparison of good and bad code, helping you understand the importance of coding standards, principles, and methodologies. You'll then get to grips with code reviews and their role in improving your code while ensuring that you adhere to industry-recognized coding standards. This C# book covers unit testing, delves into test-driven development, and addresses cross-cutting concerns. You'll explore good programming practices for objects, data structures, exception handling, and other aspects of writing C# computer programs. Once you've studied API design and discovered tools for improving code quality, you'll look at examples of bad code and understand which coding practices you should avoid. By the end of this clean code book, you'll have the developed skills you need in order to apply industry-approved coding practices

to write clean, readable, extendable, and maintainable C# code. What you will learnWrite code that allows software to be modified and adapted over timeImplement the fail-pass-refactor methodology using a sample C# console applicationAddress cross-cutting concerns with the help of software design patternsWrite custom C# exceptions that provide meaningful informationIdentify poor quality C# code that needs to be refactoredSecure APIs with API keys and protect data using Azure Key VaultImprove your code's performance by using tools for profiling and refactoringWho this book is for C# developers, team leads, senior software engineers, and software architects who want to improve the efficiency of their legacy systems. A strong understanding of C# programming is required.

Explores the history of college athletics and examines the position of sports relative to academics within the university

Presents practical advice on the disciplines, techniques, tools, and practices of computer programming and how to approach software development with a sense of pride, honor, and self-respect.

Clean Architecture

An Insider's Journey into the High-Stakes World of College Basketball

Ten Guidelines For Future-Proof Code

The Robert C. Martin Clean Code Collection (Collection)

Clean Code

Refactor your legacy code base

16th International Conference, XP 2015, Helsinki, Finland, May 25-29, 2015, Proceedings

It's time to level up your programming skills! The one thing that giants like Apple, Microsoft and Facebook have in common is that they became tech powerhouses by following a simple principle; they constantly capitalize on innovative concepts. If you want to create revolutionary software as they have, then you need to follow in their footsteps. That first step starts with mastering Object-Oriented Programming concepts! Here's how this book helps: Gain clarity on OOP nuances. Learn to leverage advanced OOP concepts to effectively build high-quality software. Write more maintainable and flexible code by adapting different OOP features. Enables COLLEGE students and FRESHERS to get industry-level knowledge in no time. Makes JOB SEEKER interviews surprisingly impressive. Following a simple but detailed question & answer format, this book also contains quick notes to enhance your coding skills for industry-level applications. The key difference between being a highly skilled programmer and a poor one is your ability to use fluid clean code. Take your coding skills to the next level with OOP Concepts Booster!

The Robert C. Martin Clean Code Collection consists of two bestselling eBooks: Clean Code: A Handbook of Agile Software Craftmanship The Clean Coder: A Code of Conduct for Professional Programmers In Clean Code, legendary software expert Robert C. Martin has teamed up with his colleagues from Object Mentor to distill their best agile practice of cleaning code “on the fly” into a book that will instill within you the values of a software craftsman and make you a better programmer—but only if you work at it. You will be challenged to think about what’s right about that code and what’s wrong with it. More important, you will be challenged to reassess your professional values and your commitment to your craft. In The Clean Coder, Martin introduces the disciplines, techniques, tools, and practices of true software craftsmanship. This book is packed with practical advice—about everything from estimating and coding to refactoring and testing. It covers much more than technique: It is about attitude. Martin shows how to approach software development with honor, self-respect, and pride; work well and work clean; communicate and estimate faithfully; face difficult decisions with clarity and honesty; and understand that deep knowledge comes with a responsibility to act. Readers of this collection will come away understanding How to tell the difference between good and bad code How to write good code and how to transform bad code into good code How to create good names, good functions, good objects, and good classes How to format code for maximum readability How to implement complete error handling without obscuring code logic How to unit test and practice test-driven development What it means to behave as a true software craftsman How to deal with conflict, tight schedules, and unreasonable managers How to get into the flow of coding and get past writer’s block How to handle unrelenting pressure and avoid burnout How to combine enduring attitudes with new development paradigms How to manage your time and avoid blind alleys, marshes, bogs, and swamps How to foster environments where programmers and teams can thrive When to say “No”--and how to say it When to say “Yes”--and what yes really means

With this title, readers will learn to create digital portfolios for future-ready success! Electronic portfolios enable college- and career-bound students to curate and demonstrate their skills and academic achievements in a way that can be continually updated and expanded upon using hyperlinks. Readers will learn how to make a digital portfolio that puts their best foot forward, from what content to include to pro tips that will take it to the next level. Whether applying to college or for a new job, or showcasing a body of work, readers will learn how to make a digital portfolio sure to set them apart.

150 Page College Ruled for Coders This book can be used to track of your bank roll, log your hands, or something to write in while playing online poker. Looking for the perfect gift for a coder or programmer in your life that loves to keep a daily journal or try to stay organized? He/she will love the clean pages of this lined diary with college ruled pages that can be used for reflecting on his/her day, making to-do lists, or doodling the day away. The notebook has journal lines and measures 6 x 9 inches which is perfect for keeping a diary, taking notes in class, making notes about your days, writing out your gratitude, or logging a book journal. Features: 150 pages 6 x 9 page size College ruled lined pages Cream/Ivory colored paper Soft cover / paperback Matte finish cover This is a great unique gift idea under \$10

Java by Comparison

The Law and Higher Education: a Casebook: Students, professors. v. 2. Administration, academic program, torts

A Handbook of Agile Software Craftmanship

Clean Code in JavaScript

A Code of Conduct for Professional Programmers

Scandal and Reform in Intercollegiate Athletics

Green Engineering, Architecture, and Technology

A simplistic design for the programmer who likes the clean, grayscale color scheme. Features: Thick, glossy paperback cover Sturdy glued binding, white paper, college ruled, 200 pages 7.44 x 9.69 inches

The Clean Coder

The ultimate guide to creating games with Roblox Studio and Luau programming