

## The Productive Programmer Neal Ford

97 Things Every Project Manager Should Know  
Programming Concurrency on the JVM  
The Software Architect Elevator  
The ThoughtWorks Anthology, Volume 2  
The Productive Programmer  
Solid Code  
The Thoughtworks Anthology  
Adaptive Code via C#  
Pragmatic Thinking and Learning  
Functional Thinking  
Cloud Application Architectures  
The Pragmatic Programmer  
Microservices Patterns  
Beautiful Architecture  
Dreaming in Code  
tmux 2  
Fundamentals of Software Architecture  
97 Things Every Software Architect Should Know  
Incremental Software Architecture  
Algorithms to Live By  
Cloud Security and Privacy  
Art of Java Web Development  
Smalltalk Best Practice Patterns  
Software Design X-Rays  
Beautiful Code  
Masterminds of Programming  
No Fluff, Just Stuff Anthology  
Seven Languages in Seven Weeks  
Joel on Software  
Domain-Specific Languages  
97 Things Every Programmer Should Know  
Presentation Patterns  
Enterprise Service Bus  
Building Evolutionary Architectures  
Java Concurrency in Practice  
Principles of Software Engineering  
Management  
Qlik Sense Cookbook  
The Productive Programmer  
Making Things Happen  
Rails for Java Developers

### 97 Things Every Project Manager Should Know

How do the experts solve difficult problems in software development? In this

## Read Book The Productive Programmer Neal Ford

unique and insightful book, leading computer scientists offer case studies that reveal how they found unusual, carefully designed solutions to high-profile projects. You will be able to look over the shoulder of major coding and design experts to see problems through their eyes. This is not simply another design patterns book, or another software engineering treatise on the right and wrong way to do things. The authors think aloud as they work through their project's architecture, the tradeoffs made in its construction, and when it was important to break rules. This book contains 33 chapters contributed by Brian Kernighan, Karl Fogel, Jon Bentley, Tim Bray, Elliotte Rusty Harold, Michael Feathers, Alberto Savoia, Charles Petzold, Douglas Crockford, Henry S. Warren, Jr., Ashish Gulhati, Lincoln Stein, Jim Kent, Jack Dongarra and Piotr Luszczek, Adam Kolawa, Greg Kroah-Hartman, Diomidis Spinellis, Andrew Kuchling, Travis E. Oliphant, Ronald Mak, Rogerio Atem de Carvalho and Rafael Monnerat, Bryan Cantrill, Jeff Dean and Sanjay Ghemawat, Simon Peyton Jones, Kent Dybvig, William Otte and Douglas C. Schmidt, Andrew Patzer, Andreas Zeller, Yukihiro Matsumoto, Arun Mehta, TV Raman, Laura Wingerd and Christopher Seiwald, and Brian Hayes. Beautiful Code is an opportunity for master coders to tell their story. All author royalties will be donated to Amnesty International.

### **Programming Concurrency on the JVM**

"This book will depict rare insights into actual failed-to-perform software systems,

## Read Book The Productive Programmer Neal Ford

followed by comprehensive root-cause analyses identifying the reasons for their unsuccessful execution in production. Remedies will be provided that offer strategies to tackle the chief issues. Last, architecture and design best practices will conclude the discussion. The book will assist users to: Mitigate risks of software development projects Increase return on investments (ROI) Provide effective tools to assess technological achievability and viability Introduce software design best practices for enterprise architecture efforts Identify actual software construction value proposition Foster software assets reuse and consolidation Accelerate time-to-market "--

### **The Software Architect Elevator**

In this truly unique technical book, today's leading software architects present valuable principles on key development issues that go way beyond technology. More than four dozen architects -- including Neal Ford, Michael Nygard, and Bill de hOra -- offer advice for communicating with stakeholders, eliminating complexity, empowering developers, and many more practical lessons they've learned from years of experience. Among the 97 principles in this book, you'll find useful advice such as: Don't Put Your Resume Ahead of the Requirements (Nitin Borwankar) Chances Are, Your Biggest Problem Isn't Technical (Mark Ramm) Communication Is King; Clarity and Leadership, Its Humble Servants (Mark Richards) Simplicity Before Generality, Use Before Reuse (Kevlin Henney) For the End User, the Interface Is the

## Read Book The Productive Programmer Neal Ford

System (Vinayak Hegde) It's Never Too Early to Think About Performance (Rebecca Parsons) To be successful as a software architect, you need to master both business and technology. This book tells you what top software architects think is important and how they approach a project. If you want to enhance your career, 97 Things Every Software Architect Should Know is essential reading.

## **The ThoughtWorks Anthology, Volume 2**

Are you working on a codebase where cost overruns, death marches, and heroic fights with legacy code monsters are the norm? Battle these adversaries with novel ways to identify and prioritize technical debt, based on behavioral data from how developers work with code. And that's just for starters. Because good code involves social design, as well as technical design, you can find surprising dependencies between people and code to resolve coordination bottlenecks among teams. Best of all, the techniques build on behavioral data that you already have: your version-control system. Join the fight for better code! Use statistics and data science to uncover both problematic code and the behavioral patterns of the developers who build your software. This combination gives you insights you can't get from the code alone. Use these insights to prioritize refactoring needs, measure their effect, find implicit dependencies between different modules, and automatically create knowledge maps of your system based on actual code contributions. In a radical, much-needed change from common practice, guide

## Read Book The Productive Programmer Neal Ford

organizational decisions with objective data by measuring how well your development teams align with the software architecture. Discover a comprehensive set of practical analysis techniques based on version-control data, where each point is illustrated with a case study from a real-world codebase. Because the techniques are language neutral, you can apply them to your own code no matter what programming language you use. Guide organizational decisions with objective data by measuring how well your development teams align with the software architecture. Apply research findings from social psychology to software development, ensuring you get the tools you need to coach your organization towards better code. If you're an experienced programmer, software architect, or technical manager, you'll get a new perspective that will change how you work with code. What You Need: You don't have to install anything to follow along in the book. The case studies in the book use well-known open source projects hosted on GitHub. You'll use CodeScene, a free software analysis tool for open source projects, for the case studies. We also discuss alternative tooling options where they exist.

### **The Productive Programmer**

Printed in full color. Software development happens in your head. Not in an editor, IDE, or design tool. You're well educated on how to work with software and hardware, but what about wetware--our own brains? Learning new skills and new

## Read Book The Productive Programmer Neal Ford

technology is critical to your career, and it's all in your head. In this book by Andy Hunt, you'll learn how our brains are wired, and how to take advantage of your brain's architecture. You'll learn new tricks and tips to learn more, faster, and retain more of what you learn. You need a pragmatic approach to thinking and learning. You need to Refactor Your Wetware. Programmers have to learn constantly; not just the stereotypical new technologies, but also the problem domain of the application, the whims of the user community, the quirks of your teammates, the shifting sands of the industry, and the evolving characteristics of the project itself as it is built. We'll journey together through bits of cognitive and neuroscience, learning and behavioral theory. You'll see some surprising aspects of how our brains work, and how you can take advantage of the system to improve your own learning and thinking skills. In this book you'll learn how to: Use the Dreyfus Model of Skill Acquisition to become more expert Leverage the architecture of the brain to strengthen different thinking modes Avoid common "known bugs" in your mind Learn more deliberately and more effectively Manage knowledge more efficiently

### **Solid Code**

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

## Read Book The Productive Programmer Neal Ford

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

## **The Thoughtworks Anthology**

As the digital economy changes the rules of the game for enterprises, the role of software and IT architects is also transforming. Rather than focus on technical

## Read Book The Productive Programmer Neal Ford

decisions alone, architects and senior technologists need to combine organizational and technical knowledge to effect change in their company's structure and processes. To accomplish that, they need to connect the IT engine room to the penthouse, where the business strategy is defined. In this guide, author Gregor Hohpe shares real-world advice and hard-learned lessons from actual IT transformations. His anecdotes help architects, senior developers, and other IT professionals prepare for a more complex but rewarding role in the enterprise. This book is ideal for: Software architects and senior developers looking to shape the company's technology direction or assist in an organizational transformation Enterprise architects and senior technologists searching for practical advice on how to navigate technical and organizational topics CTOs and senior technical architects who are devising an IT strategy that impacts the way the organization works IT managers who want to learn what's worked and what hasn't in large-scale transformation

### **Adaptive Code via C#**

Masterminds of Programming features exclusive interviews with the creators of several historic and highly influential programming languages. In this unique collection, you'll learn about the processes that led to specific design decisions, including the goals they had in mind, the trade-offs they had to make, and how their experiences have left an impact on programming today. Masterminds of

## Read Book The Productive Programmer Neal Ford

Programming includes individual interviews with: Adin D. Falkoff: APL Thomas E. Kurtz: BASIC Charles H. Moore: FORTH Robin Milner: ML Donald D. Chamberlin: SQL Alfred Aho, Peter Weinberger, and Brian Kernighan: AWK Charles Geschke and John Warnock: PostScript Bjarne Stroustrup: C++ Bertrand Meyer: Eiffel Brad Cox and Tom Love: Objective-C Larry Wall: Perl Simon Peyton Jones, Paul Hudak, Philip Wadler, and John Hughes: Haskell Guido van Rossum: Python Luiz Henrique de Figueiredo and Roberto Ierusalimschy: Lua James Gosling: Java Grady Booch, Ivar Jacobson, and James Rumbaugh: UML Anders Hejlsberg: Delphi inventor and lead developer of C# If you're interested in the people whose vision and hard work helped shape the computer industry, you'll find Masterminds of Programming fascinating.

### **Pragmatic Thinking and Learning**

Get best-in-class engineering practices to help you write more-robust, bug-free code. Two Microsoft .NET development experts share real-world examples and proven methods for optimizing the software development life cycle—from avoiding costly programming pitfalls to making your development team more efficient. Managed code developers at all levels will find design, prototyping, implementation, debugging, and testing tips to boost the quality of their code—today. Optimize each stage of the development process—from design to testing—and produce higher-quality applications. Use metaprogramming to reduce

## Read Book The Productive Programmer Neal Ford

code complexity, while increasing flexibility and maintainability Treat performance as a feature—and manage it throughout the development life cycle Apply best practices for application scalability Employ preventative security measures to ward off malicious attacks Practice defensive programming to catch bugs before run time Incorporate automated builds, code analysis, and testing into the daily engineering process Implement better source-control management and check-in procedures Establish a quality-driven, milestone-based project rhythm—and improve your results!

### **Functional Thinking**

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

## Read Book The Productive Programmer Neal Ford

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 GPar.

### **Cloud Application Architectures**

Salary surveys worldwide regularly place software architect in the top 10 best jobs, yet no real guide exists to help developers become architects. Until now. This book provides the first comprehensive overview of software architecture's many aspects. Aspiring and existing architects alike will examine architectural characteristics, architectural patterns, component determination, diagramming and presenting architecture, evolutionary architecture, and many other topics. Mark Richards and Neal Ford—hands-on practitioners who have taught software architecture classes professionally for years—focus on architecture principles that apply across all technology stacks. You'll explore software architecture in a modern light, taking into account all the innovations of the past decade. This book examines:

- Architecture patterns: The technical basis for many architectural decisions
- Components: Identification, coupling, cohesion, partitioning, and granularity
- Soft skills: Effective team management, meetings, negotiation,

## Read Book The Productive Programmer Neal Ford

presentations, and more Modernity: Engineering practices and operational approaches that have changed radically in the past few years Architecture as an engineering discipline: Repeatable results, metrics, and concrete valuations that add rigor to software architecture

### **The Pragmatic Programmer**

Anyone who develops software for a living needs a proven way to produce it better, faster, and cheaper. The Productive Programmer offers critical timesaving and productivity tools that you can adopt right away, no matter what platform you use. Master developer Neal Ford not only offers advice on the mechanics of productivity-how to work smarter, spurn interruptions, get the most out your computer, and avoid repetition-he also details valuable practices that will help you elude common traps, improve your code, and become more valuable to your team. You'll learn to: Write the test before you write the code Manage the lifecycle of your objects fastidiously Build only what you need now, not what you might need later Apply ancient philosophies to software development Question authority, rather than blindly adhere to standards Make hard things easier and impossible things possible through meta-programming Be sure all code within a method is at the same level of abstraction Pick the right editor and assemble the best tools for the job This isn't theory, but the fruits of Ford's real-world experience as an Application Architect at the global IT consultancy ThoughtWorks. Whether you're a

## Read Book The Productive Programmer Neal Ford

beginner or a pro with years of experience, you'll improve your work and your career with the simple and straightforward principles in The Productive Programmer.

### **Microservices Patterns**

ThoughtWorks is a well-known global consulting firm; ThoughtWorkers are leaders in areas of design, architecture, SOA, testing, and agile methodologies. This collection of essays brings together contributions from well-known ThoughtWorkers such as Martin Fowler, along with other authors you may not know yet. While ThoughtWorks is perhaps best known for their work in the Agile community, this anthology confronts issues throughout the software development life cycle. From technology issues that transcend methodology, to issues of realizing business value from applications, you'll find it here.

### **Beautiful Architecture**

What are the ingredients of robust, elegant, flexible, and maintainable software architecture? Beautiful Architecture answers this question through a collection of intriguing essays from more than a dozen of today's leading software designers and architects. In each essay, contributors present a notable software architecture,

## Read Book The Productive Programmer Neal Ford

and analyze what makes it innovative and ideal for its purpose. Some of the engineers in this book reveal how they developed a specific project, including decisions they faced and tradeoffs they made. Others take a step back to investigate how certain architectural aspects have influenced computing as a whole. With this book, you'll discover: How Facebook's architecture is the basis for a data-centric application ecosystem The effect of Xen's well-designed architecture on the way operating systems evolve How community processes within the KDE project help software architectures evolve from rough sketches to beautiful systems How creeping featurism has helped GNU Emacs gain unanticipated functionality The magic behind the Jikes RVM self-optimizable, self-hosting runtime Design choices and building blocks that made Tandem the choice platform in high-availability environments for over two decades Differences and similarities between object-oriented and functional architectural views How architectures can affect the software's evolution and the developers' engagement Go behind the scenes to learn what it takes to design elegant software architecture, and how it can shape the way you approach your own projects, with Beautiful Architecture.

### **Dreaming in Code**

Offers a collection of essays on philosophies and strategies for defining, leading, and managing projects. This book explains to technical and non-technical readers alike what it takes to get through a large software or web development project. It

does not cite specific methods, but focuses on philosophy and strategy.

### **tmux 2**

When you hit a rough spot in software development, it's nice to know that someone has been there before. The domain experts at ThoughtWorks share what they've learned in this anthology, bringing together the best field-tested insights in IT and software development. You'll benefit from their experience in areas from testing to information visualization, from object oriented to functional programming, from incremental development to driving innovation in delivery. You'll find yourself referring to this collection of solved problems whenever you need an expert's insight. This new collection of essays from the experts at ThoughtWorks offers practical insight and advice on a range of challenges faced daily by software developers and IT professionals. It covers a broad spectrum of software development topics, from tuning agile methodologies to hard-core language geekery. This anthology captures the wide-ranging intellect and diversity of ThoughtWorks, reflected through practical and timely topics. In it, you'll find from-the-trenches advice on topics such as continuous integration, testing, and improving the software delivery process. See how people use functional programming techniques in object-oriented languages, modern Java web applications, and deal with current problems in JavaScript development. Scan an overview of the most interesting programming languages today and the current

## Read Book The Productive Programmer Neal Ford

state of information visualization. And it's all field-tested insight, because it comes from the practical perspective of ThoughtWorks experts. Each essay focuses on extending your skills and enlarging your toolkit. And each is drawn from practical experience gained in the field. You'll benefit from this book if you are involved in developing, deploying, or testing software, either as a manager or developer.

### **Fundamentals of Software Architecture**

Presents a collection of articles covering such topics as web services, DSLs, Shale, legacy code, and CSS.

### **97 Things Every Software Architect Should Know**

A guide to the skills required for state-of-the-art web development, this book covers a variety of web development frameworks. The uses of the standard web API to create applications with increasingly sophisticated architectures are highlighted, and a discussion of the development of industry-accepted best practices for architecture is included. The history and evolution toward this architecture and the reasons it is superior to previous efforts are described, and an overview of the most popular web application frameworks, their architecture, and use is provided. The same application is built in six different frameworks, allowing

## Read Book The Productive Programmer Neal Ford

developers to conduct an informed comparison. An evaluation of the pros and cons of each framework is provided to assist developers in making decisions or evaluating frameworks on their own. Best practices covered include sophisticated user interface techniques, intelligent caching and resource management, performance tuning, debugging, testing, and web services.

### **Incremental Software Architecture**

A noted journalist chronicles three years in the lives of a team of maverick software developers, led by Lotus 1-2-3 creator Mitch Kapor, intent on creating a revolutionary personal information manager to challenge Microsoft Outlook. Reprint. 30,000 first printing.

### **Algorithms to Live By**

If you're familiar with functional programming basics and want to gain a much deeper understanding, this in-depth guide takes you beyond syntax and demonstrates how you need to think in a new way. Software architect Neal Ford shows intermediate to advanced developers how functional coding allows you to step back a level of abstraction so you can see your programming problem with greater clarity. Each chapter shows you various examples of functional thinking,

## Read Book The Productive Programmer Neal Ford

using numerous code examples from Java 8 and other JVM languages that include functional capabilities. This book may bend your mind, but you'll come away with a much better grasp of functional programming concepts. Understand why many imperative languages are adding functional capabilities Compare functional and imperative solutions to common problems Examine ways to cede control of routine chores to the runtime Learn how memoization and laziness eliminate hand-crafted solutions Explore functional approaches to design patterns and code reuse View real-world examples of functional thinking with Java 8, and in functional architectures and web frameworks Learn the pros and cons of living in a paradigmatically richer world If you're new to functional programming, check out Josh Backfield's book *Becoming Functional*.

## **Cloud Security and Privacy**

Create dynamic dashboards to perform interactive analytics for business intelligence operations Key Features Explore newly added features in Qlik Sense Discover best practices to work with data using Qlik Sense Learn to implement advanced functions for better data insight Book Description Qlik Sense allows you to explore simple and complex data to reveal hidden insight and data relationships that help you make quality decisions for overall productivity. An expert Qlik Sense user can use its features for business intelligence in an enterprise environment effectively. Qlik Sense Cookbook is an excellent guide for all aspiring Qlik Sense

## Read Book The Productive Programmer Neal Ford

developers and will empower you to create featured desktop applications to obtain daily insights at work. This book takes you through the basics and advanced functions of Qlik Sense February 2018 release. You'll start with a quick refresher on obtaining data from data files and databases, and move on to some more refined features including visualization, and scripting, as well as managing apps and user interfaces. You will then understand how to work with advanced functions like set analysis and set expressions. As you make your way through this book, you will uncover newly added features in Qlik Sense such as new visualizations, label expressions and colors for dimension and measures. By the end of this book, you will have explored various visualization extensions to create your own interactive dashboard with the required tips and tricks. This will help you overcome challenging situations while developing your applications in Qlik Sense. What you will learn Source, preview, and distribute your data through interactive dashboards Explore and work with the latest visualization functions Learn how to write and use script subroutines Make your UI advanced and intuitive with custom objects and indicators Use visualization extensions for your Qlik Sense dashboard Work with Aggr and learn to use it within set analysis Who this book is for Qlik Sense Cookbook is for data and BI analysts who want to become well versed with Qlik Sense to apply business intelligence in data. If you are a beginner in data analytics and want to adopt an independent recipe-based approach to learn the required concepts and services in detail, this book is ideal! Individuals with prior knowledge of its sister product, QlikView, will also benefit from this book. Familiarity with the

basics of business intelligence is a prerequisite.

### **Art of Java Web Development**

The software development ecosystem is constantly changing, providing a constant stream of new tools, frameworks, techniques, and paradigms. Over the past few years, incremental developments in core engineering practices for software development have created the foundations for rethinking how architecture changes over time, along with ways to protect important architectural characteristics as it evolves. This practical guide ties those parts together with a new way to think about architecture and time.

### **Smalltalk Best Practice Patterns**

Tap into the wisdom of experts to learn what every programmer should know, no matter what language you use. With the 97 short and extremely useful tips for programmers in this book, you'll expand your skills by adopting new approaches to old problems, learning appropriate best practices, and honing your craft through sound advice. With contributions from some of the most experienced and respected practitioners in the industry--including Michael Feathers, Pete Goodliffe, Diomidis Spinellis, Cay Horstmann, Verity Stob, and many more--this book contains

## Read Book The Productive Programmer Neal Ford

practical knowledge and principles that you can apply to all kinds of projects. A few of the 97 things you should know: "Code in the Language of the Domain" by Dan North "Write Tests for People" by Gerard Meszaros "Convenience Is Not an -ility" by Gregor Hohpe "Know Your IDE" by Heinz Kabutz "A Message to the Future" by Linda Rising "The Boy Scout Rule" by Robert C. Martin (Uncle Bob) "Beware the Share" by Udi Dahan

### **Software Design X-Rays**

What others in the trenches say about The Pragmatic Programmer “The cool thing about this book is that it’s great for keeping the programming process fresh. The book helps you to continue to grow and clearly comes from people who have been there.” —Kent Beck, author of Extreme Programming Explained: Embrace Change “I found this book to be a great mix of solid advice and wonderful analogies!” —Martin Fowler, author of Refactoring and UML Distilled “I would buy a copy, read it twice, then tell all my colleagues to run out and grab a copy. This is a book I would never loan because I would worry about it being lost.” —Kevin Ruland, Management Science, MSG-Logistics “The wisdom and practical experience of the authors is obvious. The topics presented are relevant and useful. By far its greatest strength for me has been the outstanding analogies—tracer bullets, broken windows, and the fabulous helicopter-based explanation of the need for orthogonality, especially in a crisis situation. I have little doubt that this book will

## Read Book The Productive Programmer Neal Ford

eventually become an excellent source of useful information for journeymen programmers and expert mentors alike.” —John Lakos, author of Large-Scale C++ Software Design “This is the sort of book I will buy a dozen copies of when it comes out so I can give it to my clients.” —Eric Vought, Software Engineer “Most modern books on software development fail to cover the basics of what makes a great software developer, instead spending their time on syntax or technology where in reality the greatest leverage possible for any software team is in having talented developers who really know their craft well. An excellent book.” —Pete McBreen, Independent Consultant “Since reading this book, I have implemented many of the practical suggestions and tips it contains. Across the board, they have saved my company time and money while helping me get my job done quicker! This should be a desktop reference for everyone who works with code for a living.” —Jared Richardson, Senior Software Developer, iRenaissance, Inc. “I would like to see this issued to every new employee at my company.” —Chris Cleeland, Senior Software Engineer, Object Computing, Inc. “If I’m putting together a project, it’s the authors of this book that I want. . . . And failing that I’d settle for people who’ve read their book.” —Ward Cunningham Straight from the programming trenches, The Pragmatic Programmer cuts through the increasing specialization and technicalities of modern software development to examine the core process--taking a requirement and producing working, maintainable code that delights its users. It covers topics ranging from personal responsibility and career development to architectural techniques for keeping your code flexible and easy to

## Read Book The Productive Programmer Neal Ford

adapt and reuse. Read this book, and you'll learn how to Fight software rot; Avoid the trap of duplicating knowledge; Write flexible, dynamic, and adaptable code; Avoid programming by coincidence; Bullet-proof your code with contracts, assertions, and exceptions; Capture real requirements; Test ruthlessly and effectively; Delight your users; Build teams of pragmatic programmers; and Make your developments more precise with automation. Written as a series of self-contained sections and filled with entertaining anecdotes, thoughtful examples, and interesting analogies, The Pragmatic Programmer illustrates the best practices and major pitfalls of many different aspects of software development. Whether you're a new coder, an experienced programmer, or a manager responsible for software projects, use these lessons daily, and you'll quickly see improvements in personal productivity, accuracy, and job satisfaction. You'll learn skills and develop habits and attitudes that form the foundation for long-term success in your career. You'll become a Pragmatic Programmer.

### **Beautiful Code**

Large IT organizations increasingly face the challenge of integrating various web services, applications, and other technologies into a single network. The solution to finding a meaningful large-scale architecture that is capable of spanning a global enterprise appears to have been met in ESB, or Enterprise Service Bus. Rather than conform to the hub-and-spoke architecture of traditional enterprise

## Read Book The Productive Programmer Neal Ford

application integration products, ESB provides a highly distributed approach to integration, with unique capabilities that allow individual departments or business units to build out their integration projects in incremental, digestible chunks, maintaining their own local control and autonomy, while still being able to connect together each integration project into a larger, more global integration fabric, or grid. Enterprise Service Bus offers a thorough introduction and overview for systems architects, system integrators, technical project leads, and CTO/CIO level managers who need to understand, assess, and evaluate this new approach. Written by Dave Chappell, one of the best known and authoritative voices in the field of enterprise middleware and standards-based integration, the book drills down into the technical details of the major components of ESB, showing how it can utilize an event-driven SOA to bring a variety of enterprise applications and services built on J2EE, .NET, C/C++, and other legacy environments into the reach of the everyday IT professional. With Enterprise Service Bus, readers become well versed in the problems faced by IT organizations today, gaining an understanding of how current technology deficiencies impact business issues. Through the study of real-world use cases and integration patterns drawn from several industries using ESB--including Telcos, financial services, retail, B2B exchanges, energy, manufacturing, and more--the book clearly and coherently outlines the benefits of moving toward this integration strategy. The book also compares ESB to other integration architectures, contrasting their inherent strengths and limitations. If you are charged with understanding, assessing, or implementing an integration

## Read Book The Productive Programmer Neal Ford

architecture, Enterprise Service Bus will provide the straightforward information you need to draw your conclusions about this important disruptive technology.

### **Masterminds of Programming**

This classic book is the definitive real-world style guide for better Smalltalk programming. This author presents a set of patterns that organize all the informal experience successful Smalltalk programmers have learned the hard way. When programmers understand these patterns, they can write much more effective code. The concept of Smalltalk patterns is introduced, and the book explains why they work. Next, the book introduces proven patterns for working with methods, messages, state, collections, classes and formatting. Finally, the book walks through a development example utilizing patterns. For programmers, project managers, teachers and students -- both new and experienced. This book presents a set of patterns that organize all the informal experience of successful Smalltalk programmers. This book will help you understand these patterns, and empower you to write more effective code.

### **No Fluff, Just Stuff Anthology**

Anyone who develops software for a living needs a proven way to produce it

## Read Book The Productive Programmer Neal Ford

better, faster, and cheaper. The Productive Programmer offers critical timesaving and productivity tools that you can adopt right away, no matter what platform you use. Master developer Neal Ford not only offers advice on the mechanics of productivity-how to work smarter, spurn interruptions, get the most out your computer, and avoid repetition-he also details valuable practices that will help you elude common traps, improve your code, and become more valuable to your team. You'll learn to: Write the test before you write the code Manage the lifecycle of your objects fastidiously Build only what you need now, not what you might need later Apply ancient philosophies to software development Question authority, rather than blindly adhere to standards Make hard things easier and impossible things possible through meta-programming Be sure all code within a method is at the same level of abstraction Pick the right editor and assemble the best tools for the job This isn't theory, but the fruits of Ford's real-world experience as an Application Architect at the global IT consultancy ThoughtWorks. Whether you're a beginner or a pro with years of experience, you'll improve your work and your career with the simple and straightforward principles in The Productive Programmer.

### **Seven Languages in Seven Weeks**

When carefully selected and used, Domain-Specific Languages (DSLs) may simplify complex code, promote effective communication with customers, improve

## Read Book The Productive Programmer Neal Ford

productivity, and unclog development bottlenecks. In *Domain-Specific Languages*, noted software development expert Martin Fowler first provides the information software professionals need to decide if and when to utilize DSLs. Then, where DSLs prove suitable, Fowler presents effective techniques for building them, and guides software engineers in choosing the right approaches for their applications. This book's techniques may be utilized with most modern object-oriented languages; the author provides numerous examples in Java and C#, as well as selected examples in Ruby. Wherever possible, chapters are organized to be self-standing, and most reference topics are presented in a familiar patterns format. Armed with this wide-ranging book, developers will have the knowledge they need to make important decisions about DSLs—and, where appropriate, gain the significant technical and business benefits they offer. The topics covered include: How DSLs compare to frameworks and libraries, and when those alternatives are sufficient Using parsers and parser generators, and parsing external DSLs Understanding, comparing, and choosing DSL language constructs Determining whether to use code generation, and comparing code generation strategies Previewing new language workbench tools for creating DSLs

### **Joel on Software**

Shows you, using detailed comparisons and commentary, how to translate your hard-earned Java knowledge and skills into the world of Ruby and Rails.

### **Domain-Specific Languages**

If the projects you manage don't go as smoothly as you'd like, *97 Things Every Project Manager Should Know* offers knowledge that's priceless, gained through years of trial and error. This illuminating book contains 97 short and extremely practical tips -- whether you're dealing with software or non-IT projects -- from some of the world's most experienced project managers and software developers. You'll learn how these professionals have dealt with everything from managing teams to handling project stakeholders to runaway meetings and more. While this book highlights software projects, its wise axioms contain project management principles applicable to projects of all types in any industry. You can read the book end to end or browse to find topics that are of particular relevance to you. *97 Things Every Project Manager Should Know* is both a useful reference and a source of inspiration. Among the 97 practical tips: "Clever Code Is Hard to Maintain and Maintenance Is Everything" -- David Wood, Partner, Zepheira "Every Project Manager Is a Contract Administrator" -- Fabio Teixeira de Melo, Planning Manager, Construtora Norberto Odebrecht "Can Earned Value and Velocity Coexist on Reports?" -- Barbee Davis, President, Davis Consulting "How Do You Define 'Finished'?" -- Brian Sam-Bodden, author, software architect "The Best People to Create the Estimates Are the Ones Who Do the Work" -- Joe Zenevitch, Senior Project Manager, ThoughtWorks "How to Spot a Good IT Developer" -- James Graham, independent management consultant "One Deliverable, One Person" --

Alan Greenblatt, CEO, Sciova

### **97 Things Every Programmer Should Know**

Your mouse is slowing you down. The time you spend context switching between your editor and your consoles eats away at your productivity. Take control of your environment with tmux, a terminal multiplexer that you can tailor to your workflow. With this updated second edition for tmux 2.3, you'll customize, script, and leverage tmux's unique abilities to craft a productive terminal environment that lets you keep your fingers on your keyboard's home row. You have a database console, web server, test runner, and text editor running at the same time, but switching between them and trying to find what you need takes up valuable time and breaks your concentration. By using tmux 2.3, you can improve your productivity and regain your focus. This book will show you how. This second edition includes many features requested by readers, including how to integrate plugins into your workflow, how to integrate tmux with Vim for seamless navigation - oh, and how to use tmux on Windows 10. Use tmux to manage multiple terminal sessions in a single window using only your keyboard. Manage and run programs side by side in panes, and create the perfect development environment with custom scripts so that when you're ready to work, your programs are waiting for you. Manipulate text with tmux's copy and paste buffers, so you can move text around freely between applications. Discover how easy it is to use tmux to

collaborate remotely with others, and explore more advanced usage as you manage multiple tmux sessions, add custom scripts into the tmux status line, and integrate tmux with your system. Whether you're an application developer or a system administrator, you'll find many useful tricks and techniques to help you take control of your terminal.

### **Presentation Patterns**

Presentation Patterns is the first book on presentations that categorizes and organizes the building blocks (or patterns) that you'll need to communicate effectively using presentation tools like Keynote and PowerPoint. Patterns are like the lower-level steps found inside recipes; they are the techniques you must master to be considered a master chef or master presenter. You can use the patterns in this book to construct your own recipes for different contexts, such as business meetings, technical demonstrations, scientific expositions, and keynotes, just to name a few. Although there are no such things as antirecipes, this book shows you lots of antipatterns—things you should avoid doing in presentations. Modern presentation tools often encourage ineffective presentation techniques, but this book shows you how to avoid them. Each pattern is introduced with a memorable name, a definition, and a brief explanation of motivation. Readers learn where the pattern applies, the consequences of applying it, and how to apply it. The authors also identify critical antipatterns: clichés, fallacies, and design

## Read Book The Productive Programmer Neal Ford

mistakes that cause presentations to disappoint. These problems are easy to avoid—once you know how. Presentation Patterns will help you Plan what you'll say, who you'll say it to, how long you'll talk, and where you'll present Perfectly calibrate your presentation to your audience Use the storyteller's "narrative arc" to full advantage Strengthen your credibility—and avoid mistakes that hurt it Hone your message before you ever touch presentation software Incorporate visuals that support your message instead of hindering it Create highly effective "infodecks" that work when you're not able to deliver a talk in person Construct slides that really communicate and avoid "Ant Fonts," "Floodmarks," "Alienating Artifacts," and other errors Master 13 powerful techniques for delivering your presentation with power, authority, and clarity Whether you use this book as a handy reference or read it from start to finish, it will be a revelation: an entirely new language for systematically planning, creating, and delivering more powerful presentations. You'll quickly find it indispensable—no matter what you're presenting, who your audiences are, or what message you're driving home.

### **Enterprise Service Bus**

\* Covers three years of the best essays. \* Essays range from technical to humorous, but are always tangible. \* Beautifully written and extremely timely. \* Google lists 183,000 links for "Joel on Software". \* Spolsky is one of the most popular programmers around today, with legions of followers.

### **Building Evolutionary Architectures**

Agile coding with design patterns and SOLID principles As every developer knows, requirements are subject to change. But when you build adaptability into your code, you can respond to change more easily and avoid disruptive rework. Focusing on Agile programming, this book describes the best practices, principles, and patterns that enable you to create flexible, adaptive code--and deliver better business value. Expert guidance to bridge the gap between theory and practice Get grounded in Scrum: artifacts, roles, metrics, phases Organize and manage architectural dependencies Review best practices for patterns and anti-patterns Master SOLID principles: single-responsibility, open/closed, Liskov substitution Manage the versatility of interfaces for adaptive code Perform unit testing and refactoring in tandem See how delegation and abstraction impact code adaptability Learn best ways to implement dependency interjection Apply what you learn to a pragmatic, agile coding project Get code samples at: <http://github.com/garymclean/AdaptiveCode>

### **Java Concurrency in Practice**

If you're involved in planning IT infrastructure as a network or system architect, system administrator, or developer, this book will help you adapt your skills to

## Read Book The Productive Programmer Neal Ford

work with these highly scalable, highly redundant infrastructure services. While analysts hotly debate the advantages and risks of cloud computing, IT staff and programmers are left to determine whether and how to put their applications into these virtualized services. Cloud Application Architectures provides answers -- and critical guidance -- on issues of cost, availability, performance, scaling, privacy, and security. With Cloud Application Architectures, you will: Understand the differences between traditional deployment and cloud computing Determine whether moving existing applications to the cloud makes technical and business sense Analyze and compare the long-term costs of cloud services, traditional hosting, and owning dedicated servers Learn how to build a transactional web application for the cloud or migrate one to it Understand how the cloud helps you better prepare for disaster recovery Change your perspective on application scaling To provide realistic examples of the book's principles in action, the author delves into some of the choices and operations available on Amazon Web Services, and includes high-level summaries of several of the other services available on the market today. Cloud Application Architectures provides best practices that apply to every available cloud service. Learn how to make the transition to the cloud and prepare your web applications to succeed.

## **Principles of Software Engineering Management**

This practical guide is designed to assist professionals with the problems involved

## Read Book The Productive Programmer Neal Ford

in developing complex software systems, presenting a set of guidelines and tools to manage the technical and organisational aspects of software engineering projects

### **Qlik Sense Cookbook**

You may regard cloud computing as an ideal way for your company to control IT costs, but do you know how private and secure this service really is? Not many people do. With *Cloud Security and Privacy*, you'll learn what's at stake when you trust your data to the cloud, and what you can do to keep your virtual infrastructure and web applications secure. Ideal for IT staffers, information security and privacy practitioners, business managers, service providers, and investors alike, this book offers you sound advice from three well-known authorities in the tech security world. You'll learn detailed information on cloud computing security that-until now-has been sorely lacking. Review the current state of data security and storage in the cloud, including confidentiality, integrity, and availability Learn about the identity and access management (IAM) practice for authentication, authorization, and auditing of the users accessing cloud services Discover which security management frameworks and standards are relevant for the cloud Understand the privacy aspects you need to consider in the cloud, including how they compare with traditional computing models Learn the importance of audit and compliance functions within the cloud, and the various

standards and frameworks to consider Examine security delivered as a service-a different facet of cloud security

### **The Productive Programmer**

A fascinating exploration of how insights from computer algorithms can be applied to our everyday lives, helping to solve common decision-making problems and illuminate the workings of the human mind All our lives are constrained by limited space and time, limits that give rise to a particular set of problems. What should we do, or leave undone, in a day or a lifetime? How much messiness should we accept? What balance of new activities and familiar favorites is the most fulfilling? These may seem like uniquely human quandaries, but they are not: computers, too, face the same constraints, so computer scientists have been grappling with their version of such issues for decades. And the solutions they've found have much to teach us. In a dazzlingly interdisciplinary work, acclaimed author Brian Christian and cognitive scientist Tom Griffiths show how the algorithms used by computers can also untangle very human questions. They explain how to have better hunches and when to leave things to chance, how to deal with overwhelming choices and how best to connect with others. From finding a spouse to finding a parking spot, from organizing one's inbox to understanding the workings of memory, Algorithms to Live By transforms the wisdom of computer science into strategies for human living.

### **Making Things Happen**

Summary Microservices Patterns teaches enterprise developers and architects how to build applications with the microservice architecture. Rather than simply advocating for the use the microservice architecture, this clearly-written guide takes a balanced, pragmatic approach, exploring both the benefits and drawbacks. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Successfully developing microservices-based applications requires mastering a new set of architectural insights and practices. In this unique book, microservice architecture pioneer and Java Champion Chris Richardson collects, catalogues, and explains 44 patterns that solve problems such as service decomposition, transaction management, querying, and inter-service communication. About the Book Microservices Patterns teaches you how to develop and deploy production-quality microservices-based applications. This invaluable set of design patterns builds on decades of distributed system experience, adding new patterns for writing services and composing them into systems that scale and perform reliably under real-world conditions. More than just a patterns catalog, this practical guide offers experience-driven advice to help you design, implement, test, and deploy your microservices-based application. What's inside How (and why!) to use the microservice architecture Service decomposition strategies Transaction management and querying patterns Effective testing strategies Deployment patterns including containers and serverless

## Read Book The Productive Programmer Neal Ford

About the Reader Written for enterprise developers familiar with standard enterprise application architecture. Examples are in Java. About the Author Chris Richardson is a Java Champion, a JavaOne rock star, author of Manning's POJOs in Action, and creator of the original CloudFoundry.com. Table of Contents Escaping monolithic hell Decomposition strategies Interprocess communication in a microservice architecture Managing transactions with sagas Designing business logic in a microservice architecture Developing business logic with event sourcing Implementing queries in a microservice architecture External API patterns Testing microservices: part 1 Testing microservices: part 2 Developing production-ready services Deploying microservices Refactoring to microservices

### **Rails for Java Developers**

"Seven Languages in Seven Weeks" presents a meaningful exploration of seven languages within a single book. Rather than serve as a complete reference or installation guide, the book hits what's essential and unique about each language.

## Read Book The Productive Programmer Neal Ford

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