Elegant Objects Volume 1

"Naked Objects is the embodiment of the Agile movement: lean, elegant, user-focused, and with testing built right in. Reduce a problem to its bare essentials, code it up with no extra fluff, then ship it out. Naked Objects brings programming back to its real purpose: expressing and solving business problems." Dave Thomas, co-author, The Agile Manifesto and The Pragmatic Programmer "I believe that this could be a landmark book. Naked Objects may well herald the next major evolution in the way systems are presented to end users, and how they're developed. Naked Objects adds near-instant prototyping to the business modeller's toolbox." Oliver Sims, co-author, Business Component Factory "A well-written description of a radical new approach to OO programming." James W Cooper, IBM T J Watson Research Center "Naked Objects is a bold approach. If you want to push the envelope and let end-users access their business objects without cluttered interfaces, read this book." Rebecca Wirfs-Brock, co-author, Object Design An object should completely model the behaviour of that which it represents. This principle of 'behaviourally complete' objects is the driving force behind this book. Naked Objects is a Java-based open source framework that exposes behaviourally complete business objects such as Customer, Product and Order, directly to the user - without the need for scripts, controllers or even dialog boxes in between. The resulting systems are empowering for the user and immensely agile. With Naked Objects the user presentation is generated automatically from the business object definitions, so you need never write another line of code for a user interface again! This book, written for business object modellers and Java developers, includes: an introduction to designing systems from naked objects a tutorial on programming with the Naked Objects framework a lightweight methodology case studies on business applications

“A debut story collection of the rarest kind ... you wish that every single entry could be an entire novel." —Entertainment Weekly Fresh, intimate stories of women’s lives from an extraordinary new literary voice, laying bare the unexpected beauty and irony in contemporary life A college freshman, traveling home, strikesup an odd, ephemeral friendship with the couple next to her on the plane. A mother prepares for her son’s wedding, her own life unraveling as his comes together. A long-lost stepbrother’s visit to New York prompts a family’s reckoning with its old taboos. A wife considers the secrets her marriage once contained. An office worker, exhausted by the ambitions of the men around her, emerges into a gridlocked city one afternoon to make a decision. In these eleven powerful stories, thrilling desire and melancholic yearning animate women’s lives, from the brink of adulthood to the labyrinthine path between twenty and thirty, to middle age, when certain possibilities quietly elapse. Tender, lucid, and piercingly funny, Objects of Desire is a collection pulsing with subtle drama, rich with unforgettable scenes, and alive with moments of recognition each more startling than the last—a spellbinding debut that announces a major talent.

Object-oriented analysis and design (OOAD) has over the years, become a vast field, encompassing such diverse topics as design process and principles, documentation tools, refactoring, and design and architectural patterns. For most students the learning experience is incomplete without implementation. This new textbook provides a comprehensive introduction to OOAD. The salient points of its coverage are: • A sound footing on object-oriented concepts such as classes, objects, interfaces, inheritance, polymorphism, dynamic linking, etc. • A good introduction to the stage of requirements analysis. • Use of UML to document user requirements and design. • An extensive treatment of the design process. • Coverage of implementation issues. • Appropriate use of design and architectural patterns. • Introduction to the art and craft of refactoring. • Pointers to resources that further the reader’s knowledge. All the main case-studies used for this book have been implemented by the authors using Java. The text is liberally peppered with snippets of code, which are short and fairly self-explanatory and easy to read. Familiarity with a Java-like syntax and a broad understanding of the structure of Java would be helpful in using the book to its full potential.

The classic manifesto of the liberated woman, this book explores every facet of a woman's life.

Learning JavaScript Design Patterns

Elegant Objects

JavaScript: The Good Parts

Code Ahead

Python 3 Object-oriented Programming

TL;DR Object-oriented programming is a mess. However, if you want to find a good job, you have to use it. This book will help you learn it, from scratch. Also, it will point you to all necessary topics in software engineering that you need to learn before you call yourse
senior software developer.

TL;DR Compound variable names, validators, private static literals, configurable objects, inheritance, annotations, MVC, dependency injection containers, reflection, ORM and even algorithms are our enemies.

One of Smithsonian's Favorite Books of 2018 One of Forbes's 2018 Best Books About Astronomy, Physics and Mathematics One of Kirkus's Best Books of 2018 The intellectual adventure story of the "double-slit" experiment, showing how a sunbeam split into two pa
challenged our understanding of light and then the nature of reality itself--and continues to almost 200 years later. Many of science's greatest minds have grappled with the simple yet elusive "double-slit" experiment. Thomas Young devised it in the early 1800s to sho
light behaves like a wave, and in doing so opposed Isaac Newton. Nearly a century later, Albert Einstein showed that light comes in quanta, or particles, and the experiment became key to a fierce debate between Einstein and Niels Bohr over the nature of reality. Ric
Feynman held that the double slit embodies the central mystery of the quantum world. Decade after decade, hypothesis after hypothesis, scientists have returned to this ingenious experiment to help them answer deeper and deeper questions about the fabric of the u
How can a single particle behave both like a particle and a wave? Does a particle exist before we look at it, or does the very act of looking create reality? Are there hidden aspects to reality missing from the orthodox view of quantum physics? Is there a place where t
world ends and the familiar classical world of our daily lives begins, and if so, can we find it? And if there's no such place, then does the universe split into two each time a particle goes through the double slit? With his extraordinarily gifted eloquence, Anil Ananthasw
travels around the world and through history, down to the smallest scales of physical reality we have yet fathomed. Through Two Doors at Once is the most fantastic voyage you can take.

Most programming languages contain good and bad parts, but JavaScript has more than its share of the bad, having been developed and released in a hurry before it could be refined. This authoritative book scrapes away these bad features to reveal a subset of Jav
that's more reliable, readable, and maintainable than the language as a whole—a subset you can use to create truly extensible and efficient code. Considered the JavaScript expert by many people in the development community, author Douglas Crockford identifies
abundance of good ideas that make JavaScript an outstanding object-oriented programming language-ideas such as functions, loose typing, dynamic objects, and an expressive object literal notation. Unfortunately, these good ideas are mixed in with bad and downr
ideas, like a programming model based on global variables. When Java applets failed, JavaScript became the language of the Web by default, making its popularity almost completely independent of its qualities as a programming language. In JavaScript: The Good
Crockford finally digs through the steaming pile of good intentions and blunders to give you a detailed look at all the genuinely elegant parts of JavaScript, including: Syntax Objects Functions Inheritance Arrays Regular expressions Methods Style Beautiful features T
beauty? As you move ahead with the subset of JavaScript that this book presents, you'll also sidestep the need to unlearn all the bad parts. Of course, if you want to find out more about the bad parts and how to use them badly, simply consult any other JavaScript bo
JavaScript: The Good Parts, you'll discover a beautiful, elegant, lightweight and highly expressive language that lets you create effective code, whether you're managing object libraries or just trying to get Ajax to run fast. If you develop sites or applications for the We
book is an absolute must.

APPLYING UML & PATTERNS 3RD EDITION

The Syntax of Objects

How to Use Objects

Fewer, Better Things

Practical Object-oriented Design in Ruby

Through Two Doors at Once

Software -- Software Engineering.

TL;DR There are 23 practical recommendations for object-oriented programmers. Most of them are completely against everything you've read in other books. For example, static methods, NULL references, getters, setters, and mutable classes are called evil.

This book takes a dramatically original approach to the history of humanity, using objects which previous civilisations have left behind them, often accidentally, as prisms through which we can explore past worlds and the lives of the men and women who lived in them. The book's range is enormous. It begins with one of the earliest surviving objects made by human hands, a chopping tool from the Olduvai gorge in Africa, and ends with an object from the 21st century which represents the world we live in today. Neil MacGregor's aim is not simply to describe these remarkable things, but to show us their significance - how a stone pillar tells us about a great Indian emperor preaching tolerance to his people, how Spanish pieces of eight tell us about the beginning of a global currency or how an early Victorian tea-set tells us about the impact of empire. Each chapter immerses the reader in a past civilisation accompanied by an exceptionally well-informed guide. Seen through this lens, history is a kaleidoscope - shifting, interconnected, constantly surprising, and shaping our world today in ways that most of us have never imagined. An intellectual and visual feast, it is one of the most engrossing and unusual history books published in years.

Larman covers how to investigate requirements, create solutions and then translate designs into code, showing developers how to make practical use of the most significant recent developments. A summary of UML notation is included

Guidance for the Aspiring Software Craftsman

The Joy of JavaScript

New Intersections of the Material Text : Essays in Honor of David Scott Kastan

Le Deuxième Sexe

Object-Oriented Implementation of Numerical Methods

The Elegant Experiment That Captures the Enigma of Our Quantum Reality

”Demystifies object-oriented programming, and lays out how to use it to design truly secure and performant applications.” —Charles Soetan, Plum.io Key Features Dozens of techniques for writing object-oriented code that’s easy to read, reuse, and maintain Write code that other programmers will instantly understand Design rules for constructing objects, changing and exposing state, and more Examples written in an instantly familiar pseudocode that’s easy to apply to Java, Python, C#, and any object-oriented language Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About The Book Well-written object-oriented code is easy to read, modify, and debug. Elevate your coding style by mastering the universal best practices for object design presented in this book. These clearly presented rules, which apply to any OO language, maximize the clarity and durability of your codebase and increase productivity for you and your team. In Object Design Style Guide, veteran developer Matthias Noback lays out design rules for constructing objects, defining methods, and much more. All examples use instantly familiar pseudocode, so you can follow along in the language you prefer. You’ll go case by case through important scenarios and challenges for object design and then walk through a simple web application that demonstrates how different types of objects can work together effectively. What You Will Learn Universal design rules for a wide range of objects Best practices for testing objects A catalog of common object types Changing and exposing state Test your object design skills with exercises This Book Is Written For For readers familiar with an object-oriented language and basic application architecture. About the Author Matthias Noback is a professional web developer with nearly two decades of experience. He runs his own web development, training, and consultancy company called “Noback’s Office.” Table of Contents: 1 ¦ Programming with objects: A primer 2 ¦ Creating services 3 ¦ Creating other objects 4 ¦ Manipulating objects 5 ¦ Using objects 6 ¦ Retrieving information 7 ¦ Performing tasks 8 ¦ Dividing responsibilities 9 ¦ Changing the behavior of services 10 ¦ A field guide to objects 11 ¦ Epilogue

Elegant ObjectsCreatespace Independent Publishing Platform

Digital objects, in their simplest form, are data. They are also a new kind of industrial object that pervades every aspect of our life today—as online videos, images, text files, e-mails, blog posts, Facebook events.Yet, despite their ubiquity, the nature of digital objects remains unclear. On the Existence of Digital Objects conducts a philosophical examination of digital objects and their organizing schema by creating a dialogue between Martin Heidegger and Gilbert Simondon, which Yuk Hui contextualizes within the history of computing. How can digital objects be understood according to individualization and individuation? Hui pursues this question through the history of ontology and the study of markup languages and Web ontologies; he investigates the existential structure of digital objects within their systems and milieux. With this relational approach toward digital objects and technical systems, the book addresses alienation, described by Simondon as the consequence of mistakenly viewing technics in opposition to culture. Interdisciplinary in philosophical and technical insights, with close readings of Husserl, Heidegger, and Simondon as well as the history of computing and the Web, Hui’s work develops an original, productive way of thinking about the data and metadata that increasingly define our world.

Unleash the power of Python 3 objects About This Book Stop writing scripts and start architecting programs Learn the latest Python syntax and libraries A practical, hands-on tutorial that teaches you all about abstract design patterns and how to implement them in Python 3 Who This Book Is For If you're new to object-oriented programming techniques, or if you have basic Python skills and wish to learn in depth how and when to correctly apply object-oriented programming in Python to design software, this is the book for you. What You Will Learn Implement objects in Python by creating classes and defining methods Separate related objects into a taxonomy of classes and describe the properties and behaviors of those objects via the class interface Extend class functionality using inheritance Understand when to use object-oriented features, and more importantly when not to use them Discover what design patterns are and why they are different in Python Uncover the simplicity of unit testing and why it's so important in Python Grasp common concurrency techniques and pitfalls in Python 3 Exploit object-oriented programming in key Python technologies such as Kivy and Django. Object-oriented programming concurrently with asyncio In Detail Python 3 is more versatile and easier to use than ever. It runs on all major platforms in a huge array of use cases. Coding in Python minimizes development time and increases productivity in comparison to other languages. Clean, maintainable code is easy to both read and write using Python's clear, concise syntax. Object-oriented programming is a popular design paradigm in which data and behaviors are encapsulated in such a way that they can be manipulated together. Many modern programming languages utilize the powerful concepts behind object-oriented programming and Python is no exception. Starting with a detailed analysis of object-oriented analysis and design, you will use the Python programming language to clearly grasp key concepts from the object-oriented paradigm. This book fully explains classes, data encapsulation, inheritance, polymorphism, abstraction, and exceptions with an emphasis on when you can use each principle to develop well-designed software. You'll get an in-depth analysis of many common object-oriented design patterns that are more suitable to Python's unique style. This book will not just teach Python syntax, but will also build your confidence in how to program. You will also learn how to create maintainable applications by studying higher level design patterns. Following this, you'll learn the complexities of string and file manipulation, and how Python distinguishes between binary and textual data. Not one, but two very powerful automated testing systems will be introduced in the book. After you discover the joy of unit testing and just how easy it can be, you'll study higher level libraries such as database connectors and GUI toolkits and learn how they uniquely apply object-oriented principles. You'll learn how these principles will allow you to make greater use of key members of the Python eco-system such as Django and Kivy. This new edition includes all the topics that made Python 3 Object-oriented Programming an instant Packt classic. It's also packed with updated content to reflect recent changes in the core Python library and covers modern third-party packages that were not available on the Python 3 platform when the book was first published. Style and approach Throughout the book you will learn key object-oriented programming techniques demonstrated by comprehensive case studies in the context of a larger project.

Naked Objects

The Elegance of the Hedgehog

An Introduction with Java & Smalltalk

A Poem in Twelve Books

Uncovering My Family's Past, One Chair, Pistol, and Pickle Fork at a Time

A History of the World in 100 Objects

“Ace of Shades has it all …an utter delight.”—Claire Legrand, New York Times bestselling author of Furyborn From the New York Times bestselling coauthor of All of Us Villains. Welcome to the City of Sin, where casino families reign, gangs infest the streets…and secrets hide in every shadow. New Reynes, t
so-called City of Sin, is no place for a properly raised young lady. But when her mother goes missing, Enne Salta must leave her finishing school—and her reputation—behind to follow her mother’s trail in the city where no one survives uncorrupted. Frightened and alone, Enne’s only clue leads her to Levi
Glaiyser—a street lord and a con man in desperate need of the compensation Enne offers. Their search sends this unlikely duo through glamorous casinos, illicit cabarets, and into the clutches of a ruthless Mafia donna. But as Levi’s enemies close in on them, a deadly secret from Enne’s past comes to light a
she must surrender herself to the City of Sin — to a vicious game of execution… Where the players never win. Praise for Ace of Shades: “A rich, satisfying, complicated story. One of the best fantasy series I’ve read in years.”—Christine Lynn Herman, author of The Devouring Gray "Thieves, rogues, and shady
characters have always fascinated me, and so I enjoyed my dive into the morally ambiguous world of New Reynes." -New York Times bestselling author Cinda Williams Chima The Shadow Game Series: Ace of Shades King of Fools Queen of Volts

Recounts how the author and her sisters inherited furniture and other artifacts collected over the course of centuries by ancestors including several who served in the military, describing the stories behind various pieces of interest and what they revealed about past family members.

"Judaism and Christianity as condensed illustrations of how people across time struggle with the materiality of life and death. Speaking across many fields, including classics, history, anthropology, literary, gender, and queer studies, the book journeys through the ancient Mediterranean world by way of the myriad physical artifacts that punctuate the transnational history of early Christianity. By bringing a psychoanalytically inflected approach to bear upon her materialist studies of religious history, Kotrosits makes a contribution not only to our understanding of Judaism and early Christianity, but also our sense of how different disciplines construe historical knowledge, and how we as people and thinkers understand our own relation to our material and affective past"--

If you’re just learning how to program, Julia is an excellent JIT-compiled, dynamically typed language with a clean syntax. This hands-on guide uses Julia 1.0 to walk you through programming one step at a time, beginning with basic programming concepts before moving on to more advanced capabilities, such
as creating new types and multiple dispatch. Designed from the beginning for high performance, Julia is a general-purpose language ideal for not only numerical analysis and computational science but also web programming and scripting. Through exercises in each chapter, you’ll try out programming concepts as you learn them. Think Julia is perfect for students at the high school or college level as well as self-learners and professionals who need to learn programming basics. Start with the basics, including language syntax and semantics Get a clear definition of each programming concept Learn about values, variables, statements, functions, and data structures in a logical progression Discover how to work with files and databases Understand types, methods, and multiple dispatch Use debugging techniques to fix syntax, runtime, and semantic errors Explore interface design and data structures through cas
studies

Ace of Shades

How to Think Like a Computer Scientist

Paradise Lost

On the Existence of Digital Objects

Pragmatic Thinking and Learning

Ordinary Objects

Whether for building interactive browser-based applications or creating server-side applications in Node, JavaScript is the most widely used language for web programming. With new features, language improvements, paradigms, and potential use cases appearing regularly, there's never been a more exciting time to be a JavaScript developer. In The Joy of JavaScript, author and JavaScript expert Luis Atencio teaches you key design concepts that lead to clean, lean, modular, and easy-to-maintain code. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications.

The design and analysis of efficient data structures has long been recognized as a key component of the Computer Science curriculum. Goodrich, Tomassia and Goldwasser's approach to this classic topic is based on the object-oriented paradigm as the framework of choice for the design of data structures. For each ADT presented in the text, the authors provide an associated Java interface. Concrete data structures realizing the ADTs are provided as Java classes implementing the interfaces. The Java code implementing fundamental data structures in this book is organized in a single Java package, net.datastructures. This package forms a coherent library of data structures and algorithms in Java specifically designed for educational purposes in a way that is complimentary with the Java Collections Framework.

"There are few books that show how to build programs of any kind. One common theme is compiler building, and there are shelves full of them. There are few others. It's an area, or a void, that needs filling. this book does a great job of showing how to build numerical analysis programs." -David N. Smith, IBM T J Watson Research Center Numerical methods naturally lend themselves to an object-oriented approach. Mathematics builds high- level ideas on top of previously described, simpler ones. Once a property is demonstrated for a given concept, it can be applied to any new concept sharing the same premise as the original one, similar to the ideas of reuse and inheritance in object-oriented (OO) methodology. Few books on numerical methods teach developers much about designing and building good code. Good computing routines are problem-specific. Insight and understanding are what is needed, rather than just recipes and black box routines. Developers need the ability to construct new programs for different applications. Object-Oriented Implementation of Numerical Methods reveals a complete OO design methodology in a clear and systematic way. Each method is presented in a consistent format,

beginning with a short explanation and following with a description of the general OO architecture for the algorithm. Next, the code implementations are discussed and presented along with real-world examples that the author, an experienced software engineer, has used in a variety of commercial applications. Features: Reveals the design methodology behind the code, including design patterns where appropriate, rather than just presenting canned solutions. Implements all methods side by side in both Java and Smalltalk. This contrast can significantly enhance your understanding of the nature of OO programming languages. Provides a step-by-step pathway to new object-oriented techniques for programmers familiar with using procedural languages such as C or Fortran for numerical methods. Includes a chapter on data mining, a key application of numerical methods.
Essays that chronicle some of life's biggest dramas: marriage, divorce, and the quest for the perfect fashion accessory. On the surface, Kate Carroll de Gutes' debut collection of essays considers her sexuality, gender presentation, and the end of her marriage. But, as editor Judith Kitchen says, "peel it back, begin to take it apart, semantically and linguistically and personally, and it all comes clear." Kate Carroll de Gutes invites readers to become collaborators in essays about issues we all face: growing up, identity, love, loss, and sometimes, the quest for the perfect fashion accessory. With wit matched by self-compassion and empathy, the essays offer a lesson on the inevitable journey back to the places we all began. "On every page, de Gutes reminds us that we all traverse life's roads with one eye fixed on the receding and mirrored past." - Stephanie Kallos, best-selling author of Broken for You.
The Book in History, the Book as History
Refactor Your Wetware
Material Culture, Experience, and the Real in the History of Early Christianity
A Modern Introduction to Programming
Eloquent JavaScript
Elements of Reusable Object-Oriented Software
While most developers today use object-oriented languages, the full power of objects is available only to those with a deep understanding of the object paradigm. How to Use Objects will help you gain that understanding, so you can write code that works exceptionally well in the real world. Author Holger Gast focuses on the concepts that have repeatedly proven most valuable and shows how to render those concepts in concrete code. Rather than settling for minimal examples, he explores crucial intricacies, clarifies easily misunderstood ideas, and helps you avoid subtle errors that could have disastrous consequences. Gast addresses the technical aspects of working with languages, libraries, and frameworks, as well as the strategic decisions associated with patterns, contracts, design, and system architecture. He explains the roles of individual objects in a complete application, how they react to events and fulfill service requests, and how to transform excellent designs into excellent code. Using practical examples based on Eclipse, he also shows how tools can help you work more efficiently, save you time, and sometimes even write high-quality code for you. Gast writes for developers who have at least basic experience: those who've finished an introductory programming course, a university computer science curriculum, or a first or second job assignment. Coverage includes • Understanding what a professionally designed object really looks like • Writing code that reflects your true intentions—and testing to make sure it does • Applying language idioms and connotations to write more readable and maintainable code • Using design-by-contract to write code that consistently does what it's supposed to do • Coding and architecting effective event-driven software • Separating model and view, and avoiding common mistakes • Mastering strategies and patterns for efficient, flexible design • Ensuring predictable object collaboration via responsibility-driven design Register your product at informit.com/register for convenient access to downloads, updates, and corrections as they become available.
In OBJECT THINKING, esteemed object technologist David West contends that the mindset makes the programmer--not the tools and techniques. Delving into the history, philosophy, and even politics of object-oriented programming, West reveals how the best programmers rely on analysis and conceptualization--on thinking--rather than formal process and methods. Both provocative and pragmatic, this book gives form to what's primarily been an oral tradition among the field's revolutionary thinkers--and it illustrates specific object-behavior practices that you can adopt for true object design and superior results. Gain an in-depth understanding of: Prerequisites and principles of object thinking. Object knowledge implicit in eXtreme Programming (XP) and Agile software development. Object conceptualization and modeling. Metaphors, vocabulary, and design for object development. Learn viable techniques for: Decomposing complex domains in terms of objects. Identifying object relationships, interactions, and constraints. Relating object behavior to internal structure and implementation design. Incorporating object thinking into XP and Agile practice.
By what code do objects connect with us, embrace us, refute us, and in the end, inform us? From this question, the author reaches into her personal life to search for those universal moments where the teacups and trinkets of our lives linger.
Are you doing all you can to further your career as a software developer? With today's rapidly changing and ever-expanding technologies, being successful requires more than technical expertise. To grow professionally, you also need soft skills and effective learning techniques. Honing those skills is what this book is all about. Authors Dave Hoover and Adewale Oshineye have cataloged dozens of behavior patterns to help you perfect essential aspects of your craft. Compiled from years of research, many interviews, and feedback from O'Reilly's online forum, these patterns address difficult situations that programmers, administrators, and DBAs face every day. And it's not just about financial success. Apprenticeship Patterns also approaches software development as a means to personal fulfillment. Discover how this book can help you make the best of both your life and your career. Solutions to some common obstacles that this book explores in-depth include: Burned out at work? "Nurture Your Passion" by finding a pet project to rediscover the joy of problem solving. Feeling overwhelmed by new information? Re-explore familiar territory by building something you've built before, then use "Retreat into Competence" to move forward again. Stuck in your learning? Seek a team of experienced and talented developers with whom you can "Be the Worst" for a while. "Brilliant stuff! Reading this book was like being in a time machine that pulled me back to those key learning moments in my career as a professional software developer and, instead of having to learn best practices the hard way, I had a guru sitting on my shoulder guiding me every step towards master craftsmanship. I'll certainly be recommending this book to clients. I wish I had this book 14 years ago!"-Russ Miles, CEO, OpenCredo
The Hidden Wisdom of Objects
The Good Parts
Objects of Desire
Object-Oriented Analysis and Design
Apprenticeship Patterns
An Agile Primer
Printed in full color. Software development happens in your head. Not in an editor, IDE, or designtool. You're well educated on how to work with software and hardware, but what about wetware--our own brains? Learning new skills and new 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 tipsto 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
Forecasting is required in many situations. Stocking an inventory may require forecasts of demand months in advance. Telecommunication routing requires traffic forecasts a few minutes ahead. Whatever the circumstances or time horizons involved, forecasting is an important aid in effective and efficient planning. This textbook provides a comprehensive introduction to forecasting methods and presents enough information about each method for readers to use them sensibly.
From the former director of the Museum of Arts and Design in New York, a timely and passionate case for the role of the well-designed object in the digital age. Curator and scholar Glenn Adamson opens Fewer, Better Things by contrasting his beloved childhood teddy bear to the smartphones and digital tablets children have today. He laments that many children and adults are losing touch with the material objects that have nurtured human development for thousands of years. The objects are still here, but we seem to care less and know less about them. In his presentations to groups, he often asks an audience member what he or she knows about the chair the person is sitting in. Few people know much more than whether it's made of wood, plastic, or metal. If we know little about how things are made, it's hard to remain connected to the world around us. Fewer, Better Things explores the history of craft in its many forms, explaining how raw materials, tools, design, and technique come together to produce beauty and utility in handmade or manufactured items. Whether describing the implements used in a traditional Japanese tea ceremony, the use of woodworking tools, or the use of new fabrication technologies, Adamson writes expertly and lovingly about the aesthetics of objects, and the care and attention that goes into producing them. Reading this wise and elegant book is a truly transformative experience.
Arguments that ordinary inanimate objects such as tables and chairs, sticks and stones, simply do not exist have become increasingly common and increasingly prominent. Some are based on demands for parsimony or for a non-arbitrary answer to the special composition question; others arise from prohibitions against causal redundancy, ontological vagueness, or co-location; and others still come from worries that a common sense ontology would be a rival to a scientific one. Until now, little has been done to address these arguments in a unified and systematic way. Ordinary Objects is designed to fill this gap, demonstrating that the mistakes behind all of these superficially diverse eliminativist arguments may be traced to a common source. It aims to develop an ontology of ordinary objects subject to no such problems, providing perhaps the first sustained defense of a common sense ontology in two generations. The work done along the way addresses a number of major issues in philosophy of language and metaphysics, contributing to debates about analyticity, identity conditions, co-location and the grounding problem, vagueness, overdetermination, parsimony, and ontological commitment. In the end, the most important result of addressing these eliminativist arguments is not merely avoiding their conclusions; examining their failings also gives us reason to suspect that many apparent disputes in ontology are pseudo-debates. For it brings into question widely-held assumptions about which uses of metaphysical principles are appropriate, which metaphysical demands are answerable, and how we should go about addressing such fundamental questions as "What exists?". As a result, the work of Ordinary Objects promises to provide not only the route to a reflective understanding of our unreflective common-sense view, but also a better understanding of the proper methods and limits of metaphysics. "Ordinary Objects is well worth reading because it sheds new light on how to preserve the credibility of familiar things."--Marianne Djuth, The Review of Metaphysics "In Ordinary Objects , Amie Thomasson mounts a spirited and vigorous defense of the reality of ordinary objects."--Terry Horgan, Times Literary Supplement "Ordinary Objects is a fine book.... [Thomasson] writes insightfully and persuasively, and she has a realistic view of what metaphysical arguments can and cannot demonstrate... she approaches metaphysical theorizing more systematically than many other recent writers, drawing attention to the ways in which questionable assumptions in one area of philosophy are undergirding seemingly powerful arguments in another. Everyone working in metaphysics should make time for this volume."--R. W. Fischer, Metaphilosophy "In Ordinary Objects , Thomasson pursues an integrated conception of ontology and metaontology. In ontology, she defends the existence of shoes, ships, and other ordinary objects. In metaontology, she defends a deflationary view of ontological inquiry, designed to suck the air out of arguments against ordinary objects. The result is an elegant and insightful defense of a common sense worldview."--Jonathan Schaffer, Philosophical Books "Amie Thomasson has written a lovely book which is certain to irritate many professional metaphysicians. But it is not just irritating: it is challenging...This book would be good supplementary text for upper-level metaphysics classes or seminars in which the sorts of arguments to which Thomasson replies are also read."--Alan Sidelle, The Philosophical Quarterly
Design Patterns
Data Structures and Algorithms in Java
A JavaScript and jQuery Developer's Guide
The Object-Oriented Thought Process
Objects in Mirror Are Closer Than They Appear
Stories
JavaScript is at the heart of almost every modern Web application, whether it's Google Apps, Twitter, or the newest browser-based game. Though it's simple for beginners to pick up and play with, JavaScript is not a toy—it's a flexible and complex language that can be used to build full-scale applications. Eloquent JavaScript dives into this flourishing language and teaches you to write code that's beautiful and effective. By immersing you in example code and encouraging experimentation right from the start, the author quickly gives you the tools you need to build your own programs. As you follow along with examples like an artificial life simulation and a version of the classic game Sokoban, you'll learn to: -Understand the essential elements of programming: syntax, control, and data -Use object-oriented and functional programming techniques to organize and clarify your programs -Script the browser and make basic Web applications -Work with tools like regular expressions and XMLHttpRequest objects And since programming is an art that's best learned by doing, all example code is available online in an interactive sandbox for you to experiment with. With Eloquent JavaScript as your guide, you can tweak, expand, and modify the author's code, or throw it away and build your own creations from scratch. Before you know it, you'll be fluent in the language of the Web.
The #1 New York Times Bestselling Series An Amazon Best YA Book of 2020 Glitter Magazine's #1 Pick for Best YA of 2020 Optioned for Film by Universal My whole world changed when I stepped inside the academy. Nothing is right about this place or the other students in it. Here I am, a mere mortal among gods...or monsters. I still can't decide which of these warring factions I belong to, if I belong at all. I only know the one thing that unites them is their hatred of me. Then there's Jaxon Vega. A vampire with deadly secrets who hasn't felt anything for a hundred years. But there's something about him that calls to me, something broken in him that somehow fits with what's broken in me. Which could spell death for us all. Because Jaxon walled himself off for a reason. And now someone wants to wake a sleeping monster, and I'm wondering if I was brought here intentionally—as the bait. ***INCLUDES 3 BONUS SCENES FROM THE HERO'S POV*** Don't miss a single book in the series that spawned a phenomenon! The Crave series is best enjoyed in order: Crave Crush Covet Court Charm Cherish
With Learning JavaScript Design Patterns, you'll learn how to write beautiful, structured, and maintainable JavaScript by applying classical and modern design patterns to the language. If you want to keep your code efficient, more manageable, and up-to-date with the latest best practices, this book is for you. Explore many popular design patterns, including Modules, Observers, Facades, and Mediators. Learn how modern architectural patterns—such as MVC, MVP, and MVVM—are useful from the perspective of a modern web application developer. This book also walks experienced JavaScript developers through modern module formats, how to namespace code effectively, and other essential topics. Learn the structure of design patterns and how they are written Understand different pattern categories, including creational, structural, and behavioral Walk through more than 20 classical and modern design patterns in JavaScript Use several options for writing modular code—including the Module pattern, Asyncronous Module Definition (AMD), and CommonJS Discover design patterns implemented in the jQuery library Learn popular design patterns for writing maintainable jQuery plug-ins "This book should be in every JavaScript developer's hands. It's the go-to book on JavaScript patterns that will be read and referenced many times in the future."—Andrée Hansson, Lead Front-End Developer, presis!
The essays in this collection reach beyond book history to address fundamental questions about historicism with a broad range of issues such as gender and sexuality, religion, political theory, economic history, adaptation and appropriation, and quantitative analysis and digital humanities.
Junior Objects
Code and Concepts
Objects of Our Affection
Object Design Style Guide
Forecasting: principles and practice
Object Thinking
TL;DR It's a semi-autobiographical fiction book about a software architect who is involved in programming, debugging, releasing, testing, organizing, team work, and management issues.
The Object-Oriented Thought Process Third Edition Matt Weisfeld An introduction to object-oriented concepts for developers looking to master modern application practices. Object-oriented programming (OOP) is the foundation of modern programming languages, including C++, Java, C#, and Visual Basic .NET. By designing with objects rather than treating the code and data as separate entities, OOP allows objects to fully utilize other objects' services as well as inherit their functionality. OOP promotes code portability and reuse, but requires a shift in thinking to be fully understood. Before jumping into the world of object-oriented programming languages, you must first master The Object-Oriented Thought Process. Written by a developer for developers who want to make the leap to object-oriented technologies as well as managers who simply want to understand what they are managing, The Object-Oriented Thought Process provides a solution-oriented approach to object-oriented programming. Readers will learn to understand object-oriented design with inheritance or composition, object aggregation and association, and the difference between interfaces and implementations. Readers will also become more efficient and better thinkers in terms of object-oriented development. This revised edition focuses on interoperability across various technologies, primarily using XML as the communication mechanism. A more detailed focus is placed on how business objects operate over networks, including client/server architectures and web services. "Programmers who aim to create high quality software–as all programmers should–must learn the varied subtleties of the familiar yet not so familiar beasts called objects and classes. Doing so entails careful study of books such as Matt Weisfeld's The Object-Oriented Thought Process." –Bill McCarty, author of Java Distributed Objects, and Object-Oriented Design in Java Matt Weisfeld is an associate professor in business and technology at Cuyahoga Community College in Cleveland, Ohio. He has more than 20 years of experience as a professional software developer, project manager, and corporate trainer using C++, Smalltalk, .NET, and Java. He holds a BS in systems analysis, an MS in computer science, and an MBA in project management. Weisfeld has published many articles in major computer trade magazines and professional journals.
The phenomenal New York Times bestseller that "explores the upstairs-downstairs goings-on of a posh Parisian apartment building" (Publishers Weekly). In an elegant hôtel particulier in Paris, Renée, the concierge, is all but invisible—short, plump, middle-aged, with bunions on her feet and an addiction to television soaps. Her only genuine attachment is to her cat, Leo. In short, she's everything society expects from a concierge at a bourgeois building in an upscale neighborhood. But Renée has a secret: She furtively, ferociously devours art, philosophy, music, and Japanese culture. With biting humor, she scrutinizes the lives of the tenants—her inferiors in every way except that of material wealth. Paloma is a twelve-year-old who lives on the fifth floor. Talented and precocious, she's come to terms with life's seeming futility and decided to end her own on her thirteenth birthday. Until then, she will continue hiding her extraordinary intelligence behind a mask of mediocrity, acting the part of an average pre-teen high on pop culture, a good but not outstanding student, an obedient if obstinate daughter. Paloma and Renée hide their true talents and finest qualities from a world they believe cannot or will not appreciate them. But after a wealthy Japanese man named Ozu arrives in the building, they will begin to recognize each other as kindred souls, in a novel that exalts the quiet victories of the inconspicuous among us, and "teaches philosophical lessons by shrewdly exposing rich secret lives hidden beneath conventional exteriors" (Kirkus Reviews). "The narrators' kinetic minds and engaging voices (in Alison Anderson's fluent translation) propel us ahead." —The New York Times Book Review "Barbery's sly wit . . . bestows lightness on the most ponderous cogitations." —The New Yorker
The Complete Guide to Writing More Maintainable, Manageable, Pleasing, and Powerful Ruby Applications Ruby's widely admired ease of use has a downside: Too many Ruby and Rails applications have been created without concern for their long-term maintenance or evolution. The Web is awash in Ruby code that is now virtually impossible to change or extend. This text helps you solve that problem by using powerful real-world object-oriented design techniques, which it thoroughly explains using simple and practical Ruby examples. Sandi Metz has distilled a lifetime of conversations and presentations about object-oriented design into a set of Ruby-focused practices for crafting manageable, extensible, and pleasing code. She shows you how to build new applications that can survive success and repair existing applications that have become impossible to change. Each technique is illustrated with extended examples, all downloadable from the companion Web site,poodr.info. The first title to focus squarely on object-oriented Ruby application design, Practical Object-Oriented Design in Ruby will guide you to superior outcomes, whatever your previous Ruby experience. Novice Ruby programmers will find specific rules to live by; intermediate Ruby programmers will find valuable principles they can flexibly interpret and apply; and advanced Ruby programmers will find a common language they can use to lead development and guide their colleagues. This guide will help you Understand how object-oriented programming can help you craft Ruby code that is easier to maintain and upgrade Decide what belongs in a single Ruby class Avoid entangling objects that should be kept separate Define flexible interfaces among objects Reduce programming overhead costs with duck typing Successfully apply inheritance Build objects via composition Design cost-effective tests Solve common problems associated with poorly designed Ruby code
Think Julia
Crave
The Lives of Objects