Design Patterns - Erich Gamma 2015

Design Patterns - Erich Gamma 1994-10-31 Capturing a wealth of experience about the design of object-oriented software, four top-notch designers present a catalog of simple and succinct solutions to commonly occurring design problems. Previously undocumented, these 23 patterns allow designers to create more flexible, elegant, and ultimately reusable designs without having to rediscover the design solutions themselves. The authors begin by describing what patterns are and how they can help you design object-oriented software. They then go on to systematically name, explain, evaluate, and catalog recurring designs in object-oriented systems. With Design Patterns as your guide, you will learn how these important patterns fit into the software development process, and how your problems can be solved by applying your own design problems most effectively. Each pattern describes the circumstances in which it is applicable, when it can be applied in view of other design constraints, and the consequences and trade-offs of using the pattern within a larger design. All patterns are compiled from real systems and are based on real-world examples. Each pattern also includes code that demonstrates how it may be implemented in object-oriented programming languages like C++ or Smalltalk.

Apex Design Patterns - Jitendra Zaa 2016-04-27 Harness the power of Apex design patterns to build robust and scalable code architectures on the Force.com platform About This Book Apex Creational, Structural and behavioural patterns in Apex to fix governor limit issues. Have a grasp of the anti patterns to be taken care in Apex which could have adverse effect on the application. The authors, Jitendra Zaa is a salesforce MVP and Anshul Verma has 12+ years of experience in the area of application development. Who This Book Is For If you are a competent developer with working knowledge of Apex, and now want to dive deep into the world of Apex design patterns to optimize the application performance, then this book is for you. Prior knowledge of Salesforce and Force.com platform is recommended. What You Will Learn Apply OOPs principal in Apex to design a robust and efficient solution to address various facets to a business problem Get to grips with the benefits and applicability of using different design patterns in Apex Solve problems while instantiating, structuring and giving dynamic behavior to Apex classes Understand the implementation of creational, structural, behavioral, concurrency and anti-patterns in your application Follow the Apex best practices to resolve governor limit issues Get clued up about the Inheritance, abstract classes, polymorphism in Apex to deal with the object mechanism Master various design patterns and determine the best out of them Explore the anti patterns that could not be applied to Apex and their appropriate solutions In Detail Apex is an on-demand programming language providing a complete set of features for building business applications - including data models and objects to manage data. Apex being a proprietary programming language from Salesforce to be worked with multi tenant environment is a lot different than traditional OOPs languages like Java and C++. It acts as a workflow engine for managing collaboration of the data between users, a user interface model to handle forms and other interactions, and a SOAP API for programmatic access and integration. Apex Design Patterns gives you an insight to several problematic situations that can arise while developing on Force.com platform and the usage of Design patterns to solve them. Packed with real life examples, it gives you a walkthrough from learning design patterns that Apex can offer us, to implementing the appropriate ones in your own application. Furthermore, we learn about the creational patterns that deal with object creation mechanism and structural patterns that helps to identify the relationship between entities. Also, the behavioural and concurrency patterns are put forward explaining the communication between objects and multi-threaded programming paradigm respectively. We later on, deal with the issues regarding structuring of classes, instantiating or how to give a dynamic behaviour at a runtime, with the help of anti-patterns. We learn the basic OOPs principal in polymorphic and modular way to enhance its capability. Also, best practices of writing Apex code are explained to differentiate between the implementation of appropriate patterns. This book will also explain some unique patterns that could be applied to get around governor limits. By the end of this book, you will be a maestro in developing your applications on Force.com for Salesforce Style and approach This book is a step-by-step guide, complete with well-tested programs and real world situations to solve your common occurring problems in Apex design by using the anti-patterns. It gets cracking from exploring every appropriate solution to comparing the best one as per OOPs principal.

Learning JavaScript Design Patterns-Addy Osmani 2012-07-08 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, Factories, 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.

Learning JavaScript Design Patterns-Addy Osmani 2012-07-08 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, Factories, 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, Asynchronous Module Definition (AMD), and CommonJS 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."—Andrië Hansson, Lead Front-End Developer, presis!

Design Patterns in Modern C++-Dmitri Nesteruk 2018-04-18 Apply modern C++17 to the implementations of classic design patterns. As well as covering traditional design patterns, this book fleshes out new patterns and approaches that will be useful to C++ developers. The author presents concepts as a fun investigation of how problems can be solved in different ways, along the way using varying degrees of technical sophistication and explaining different sorts of trade-offs. Design Patterns in Modern C++ also provides a technology demo for modern C++, showcasing how some of its latest features (e.g., coroutines) make difficult problems a lot easier to solve. The examples in this book are all suitable for putting into production, with only a few simplifications made in order to aid readability. What You Will Learn Apply design patterns to modern C++ programming Use creational patterns of builder, factories, prototype and singleton Implement structural patterns such as adapter, bridge, decorator, facade and more Work with the behavioral patterns such as chain of responsibility, command, iterator, mediator and more Apply functional design patterns such as Monad and more Who This Book Is For Those with at least some prior programming experience, especially in C++.

Design Patterns Explained-Alan Shalloway 2004-10-12 "One of the great things about the book is the way the authors explain concepts very simply using analogies rather than programming examples-this has been very inspiring for a product I'm working on, an audio-only introduction to OOP and software development." - Bruce Eckel "...I would expect that readers with a basic understanding of object-oriented programming and design would find this book useful, before approaching design patterns completely. Design Patterns Explained complements the existing design patterns texts and may perform a very useful role, fitting between introductory
Use will feed in and refine individual patterns and produce an evolving system of patterns. Visit our Web Page http://www.wiley.com/comppublishers

Design Patterns in .NET-Dmitri Nesteruk 2019-05-11 Implement design patterns in .NET using the latest versions of the C# and F# languages. This book provides a comprehensive overview of the field of design patterns as they are used in today's developer toolbox. Using the C# programming language, Design Patterns in .NET explores classic design patterns implemented on .NET and discusses the applicability and relevance of specific language features for the purpose of implementing patterns. You will learn by example, reviewing scenarios where patterns are applicable. MVP and patterns expert Dmitri Nesteruk demonstrates possible implementations of patterns, discusses alternatives and pattern inter-relationships, and illustrates the way that a dedicated refactoring tool (ReSharper) can be used to implement design patterns with ease. What You'll Learn Know the latest pattern implementations available in C# and F# Refer to researched and proven variations of patterns Study complex design patterns implemented on .NET, including many that cover advanced scenarios Use the latest implementations of C# and Visual Studio/ReSharper Who This Book Is For Developers who have some experience in the C# language and want to expand their comprehension of the art of programming by leveraging design approaches to solving modern problems

Enterprise Application Architecture with .NET Core-Ganesan Senthilvel 2017-04-25 Architect and design highly scalable, robust, clean and highly performant applications in .NET Core About This Book Incorporate architectural soft-skills such as DevOps and Agile methodologies to enhance program-level objectives Gain knowledge of architectural approaches on the likes of SOA architecture and microservices to provide traceability and rationale for architectural decisions Explore a variety of practical use cases and code examples to implement the tools and techniques described in the book Who This Book Is For This book is for experienced .NET developers who are aspiring to become architects of enterprise-grade applications, as well as software architects who would like to leverage .NET to create effective blueprints of applications. What You Will Learn Grasp the important aspects and best practices of application lifecycle management Leverage the popular ALM tools, application insights, and their usage to monitor performance, testability, and optimization tools in an enterprise Explore various authentication models such as social media-based authentication, OAuth and OpenID Connect, learn authorization techniques Explore Azure with various solution approaches for Microservices and Serverless architecture along with Docker containers Gain knowledge about the recent market trends and practices and how they can be achieved with .NET Core and Microsoft tools and technologies In Detail If you want to design and develop enterprise applications using .NET Core as the development framework and learn about industry-wide best practices and guidelines, then this book is for you. The book starts with a brief introduction to enterprise architecture, which will help you to understand what enterprise architecture is and what the key components are. It will then teach you about the types of patterns and the principles of software development, and explain the various aspects of distributed computing to keep your applications effective and scalable. These chapters act as a catalyst to start the practical implementation, and design and develop applications using different architectural approaches, such as layered architecture, service oriented architecture, microservices and cloud-specific solutions. Gradually, you will learn about the different approaches and models of the Security framework and explore various authentication models and authorization techniques, such as social media-based authentication and safe storage using app secrets. By the end of the book, you will get to know the concepts and usage of the emerging fields, such as DevOps, BigData, architectural practices, and Artificial Intelligence. Style and approach Filled with examples and use cases, this guide takes a no-nonsense approach to show you the best tools and techniques required to become a successful software architect.

Head First Design Patterns-Eric Freeman 2004-10-25 Using research in neurobiology, cognitive science and learning theory, this text loads patterns into your brain in a way that lets you put them to work immediately, makes you better at solving software design problems, and improves your ability to speak the language of patterns with others on your team.

Design Patterns For Dummies-Steve Holzner 2006-07-28 There’s a pattern here, and here’s how to use it! Find out how the 23 leading design patterns can save you time and trouble Ever feel as if you’ve solved this
programming problem before? You – or someone -- probably did, and that’s why there’s a design pattern to help
this time around. This book shows you how (and when) to use the famous patterns developed by the “Gang of Four,”
plus some new ones, all designed to make your programming life easier. Discover how to: * Simplify the
programming process with design patterns * Make the most of the Decorator, Factory, and Adapter patterns * Identify
which pattern applies * Reduce the amount of code needed for a task * Create your own patterns

Hohb on Patterns-Allen Hohb 2004-09-27 * Allen Hohb is a highly regarded instructor for the University of
California, Berkeley. Extension. He has taught since 1982 on various topics, including Object-Oriented Analysis
and Design, Java, C++, C. Hohb will use this book in his Berkeley Extension classes. * Hohb is a regular
presenter at the Software Development conferences and is Contributing Editor for the online magazine
JavaWorld, for whom he writes the Java Toolbox. He also wrote the OO Design Process column for IBM
DeveloperWorks. * This book is not time-sensitive. It is an extremely well-thought out approach to learning design
patterns, with Java as the example platform, but the concepts presented are not limited to just Java programmers.
This is a complement to the Addison-Wesley seminal "Design Patterns" book by the "Gang of Four".

Mastering Python Design Patterns-Kamon Ayeva 2018-08-31 Python is an object-oriented, scripting language
that is used in wide range of categories. In software engineering, a design pattern is a recommended solution to
a software design problem. Although not new, design patterns remain one of the hottest topics in software
engineering and they come as a ready reference for software developers to ...

Game Programming Patterns-Robert Nyström 2014-11-03 The biggest challenge facing many game
programmers is completing their game. Most game projects fizzle out, overwhelmed by the complexity of their
own code. Game Programming Patterns tackles that exact problem. Based on years of experience in shippered AAA
titles, this book collects proven patterns to untangle and optimize your game, organized as independent recipes so
you can pick just the patterns you need. You will learn how to write a robust game loop, how to organize your
entities using components, and take advantage of the CPU’s cache to improve your performance. You’ll dive deep
into how scripting engines encode behavior, how quadtrees and other spatial partitions optimize your engine, and
how other classic design patterns can be used in games.

Professional Java EE Design Patterns-Murat Yener 2014-12-17 Master Java EE design pattern implementation
to improve your design skills and your application’s architecture Professional Java EE Design Patterns is the
perfect companion for anyone who wants to work more effectively with Java EE, and the only resource that covers
both the theory and application of design patterns in solving real-world problems. The authors guide readers
through both the fundamental and advanced features of Java EE 7, presenting patterns throughout,
and demonstrating how they are used in day-to-day problem solving. As the most popular programming language
in community-driven enterprise software, Java EE provides an API and runtime environment that is a superset of
Java SE. Written for the junior and experienced Java EE developer, Java EE Design Patterns offers a comprehensive
effectiveness, the book covers areas including: Implementation and problem-solving with design patterns
Connection between existing Java SE design patterns and new Java EE concepts Harnessing the power of Java EE in
design patterns Individually-based focus that fully explores each pattern Understand the limitation imposed by Java EE
and how to solve them using design patterns Colorful war-stories showing how patterns were used in the field to solve real-life problems Unlike most Java EE books that simply offer descriptions or recipes, this book drives home the implementation of the pattern by exploring the solutions to ensure that the reader learns how the patterns should be used and to be aware of their pitfalls. For the programmer looking for a
comprehensive guide that is factually useful in the everyday workflow, Professional Java EE Design Patterns is the
definitive resource on the market.

Easy Learning Design Patterns of Java Patterns-Jaehoon Hong 2019-04-25 Experience about the design of object-
oriented software, the design patterns allow designers to create more flexible, elegant, and ultimately reusable
designs without having to re-invent the design solutions themselves. Each pattern describes the circumstances
in which it is applicable, when it can be applied in view of other design constraints, and the consequences and trade-
offs of using the pattern within a larger design. All patterns are compiled from real systems and are based on real
-world examples. Each pattern also includes code that demonstrates how it may be implemented in object-oriented
Case32. Proxy Pattern Case

The Clean Coder-Robert C. Martin 2011 Presents practical advice on the disciplines, techniques, tools, and
practices of computer programming and how to approach software development with a sense of pride, honor, and
self-respect.

Pattern Languages of Program Design Design Languages 5: Dragos-Andion Manelescu 2006 The long awaited fifth volume in a
collection of key practices for pattern languages and design.

J2EE Design Patterns-William Crawford 2003-09-24 Architects of buildings and architects of software have
more in common than most people think. Both professions require attention to detail, and both practitioners will
see their work collapse around them if they make too many mistakes. It’s impossible to imagine a world in which
buildings are still common but software is still chaotic. But whereas architects are trained for years to solve
problems without blueprints, or in this case, design patterns.A software design pattern can be identified as "a recurring
solution to a recurring problem." Using design patterns for software development makes sense in the same way
that architectural design patterns make sense—if it works well in one place, why not use it in another? But
developers have had enough of books that simply catalog design patterns without extending into new areas, and
books that are so theoretical that you can’t actually do anything better after reading them than you could before
you started. Crawford and Kaplan’s J2EE Design Patterns offers extensive coverage of the five problem areas enterprise developers
face: Maintenance (Extensibility) Performance (System Scalability) Data Modeling (Business Object Modeling)
Transactions (process Modeling) Messaging (Interoperability) And with its careful balance between theory and
practice, J2EE Design Patterns will give developers new to the Java enterprise development arena a solid
understanding of how to approach a wide variety of architectural and procedural problems, and will give
experienced J2EE pros an opportunity to extend and improve on their existing experience.

Hands-On Design Patterns with C++-Fedor C. Pikus 2019-01-30 A comprehensive guide with extensive
coverage on concepts such as OOP, functional programming, generic programming, and STL along with the latest
features of C++14, the book explores design patterns in C++. Design Patterns and Components of C++ in order to master application
design Learn tricks, techniques, and best practices to solve common design and architectural challenges
Understand the limitation imposed by C++ and how to solve them using design patterns Book Description C++ is
a general-purpose programming language designed with the goals of efficiency, performance, and flexibility in
mind. Design patterns are commonly accepted solutions to well-recognized design problems. In essence, they are
a library of reusable components, only for software architecture, and not for a concrete implementation. The focus
of this book is on the design patterns that naturally lend themselves to the needs of a C++ programmer, and on
the patterns that uniquely benefit from the features of C++, in particular, the generic programming. Armed with the
knowledge of these patterns, you will spend less time searching for a solution to a common problem and be
familiar with the solutions developed from experience, as well as their advantages and drawbacks. The other use
Clean Code—Robert C. Martin 2009 Looks at the principles and clean code, includes case studies showcasing the practices of writing clean code, and contains a list of heuristics and "smells" accumulated from the process of writing clean code.

Learning Dynamics NAV Patterns—Mark Brummel 2015-09-24 Create solutions that are easy to maintain, quick to upgrade, and follow proven concepts and designs About This Book Design software that is maintainable outside the ecosystem of their creators Ensure quality by following patterns that have been proved to work Over two dozen practical Architectural and Design patterns Who This Book Is For Learning Dynamics NAV Patterns is intended for developers, architects, (technical) consultants, and application managers. You may have very little or no knowledge about NAV patterns, but you should be acquainted with programming. What You Will Learn Apply object-oriented practices to C/AL programming Structure your application to avoid merge conflicts Befactor legacy code and avoid anti-patterns Design decision trees to decide when to use which patterns Clone codes and their application in Dynamics NAV Make your application extensible by creating predefined hooks and faucets In Detail Microsoft Dynamics NAV is a complete ERP system, which also contains a robust set of development tools to support customization and enhancement. These include an object designer for each of the seven application objects, object designers for application programming orient, a user interface designer, a debugger, and a programming testing language support. Learning Dynamics NAV Patterns will guide you through the NAV way of solving problems. This book will first introduce you to patterns and the software architecture of the NAV and then help you to build an example application. Then, it walks you through the details of architectural patterns, design patterns, and implementation patterns. This book will also talk about anti-patterns and handling legacy code. Finally, it teaches you to build solutions using patterns. Proven patterns and best practices will help you create better solutions that are easy to maintain in larger teams across several locations. It will guide you through combining abstract patterns using easy-to-understand examples and will help you decide which patterns to use in which scenarios. Style and approach This book explains the concepts of patterns, code structuring, and object-oriented concepts in a way that is easy to understand for Dynamics NAV specialists through practical examples.

Design Patterns CD—Erich Gamma 1998 The 23 patterns contained in the book, Design Patterns: Elements of Reusable Object-Oriented Software have become an essential resource for anyone developing reusable software designs. Now these design patterns, along with the entire text of the book, are being made available on CD. This electronic version will enable programmers to install the patterns directly onto a computer or network and create an architecture building reusable components. Produced in HTML format, the CD is heavily cross-referenced with numerous links to the online text.

Design Patterns—Erich Gamma 1990 This is the eBook version of the printed book. If the print book includes a CD, this eBook version includes the CD.

Kubernetes Patterns—Bilgın İryam 2019-04-09 The way developers design, build, and run software has changed significantly with the evolution of microservices and containers. These modern architectures use new primitives that require a different set of practices than most developers, tech leads, and architects are accustomed to. With this focused guide, Bilgın İryam and Roland Huß from Red Hat provide common reusable elements, patterns, and practices for designing and implementing cloud-native applications on Kubernetes. Each pattern includes a high-level overview of the problem, a proposed solution with Kubernetes specifics. Many patterns are also backed by concrete code examples. This book is ideal for developers already familiar with basic Kubernetes concepts who want to learn common cloud native patterns. You’ll learn about the following pattern categories: Foundational patterns cover the core principles and practices for building container-based cloud-native applications. Behavioral patterns explore finer-grained concepts for managing various types of container and platform interactions. Structural patterns help you organize containers within a pod, the atom of the Kubernetes platform. Configuration patterns provide insight into how application configurations can be handled in Kubernetes. Advanced patterns covers more advanced topics such as extending the platform with operators.

The Pragmatic Programmer—David Thomas 2009-09-15 “One of the most significant books in my life.” —Obie Fernandez, Author, The Rails Way “Twenty years ago, the first edition of The Pragmatic Programmer completely changed the trajectory of my career. This new edition could do the same for yours.” —Mike Cohn, Author of Succeeding with Agile, Agile Estimating and Planning, and User Stories Applied “...filled with practical advice, both technical and professional, that will serve you and your projects well for years to come.” —Andrea Goulet, CEO, Corpayse, Founder, LegacyCode.Rocks “…lightning does strike twice, and this book is proof.” —Vicky Brassere, Director of Open Source Strategy, Juniper Networks The Pragmatic Programmer is one of those rare tech books you’ll read, re-read, and read again over the years. Whether you’re new to the field or an expert fresh into a new role, you’ll come away every time with new understandings. Andy Hunt wrote the first edition of this influential book in 1999 to help their clients create better software and rediscover the joy of coding. These lessons have helped a generation of programmers examine the very essence of software development, independent of any particular language, framework, or methodology, and the Pragmatic philosophy has spawned hundreds of books, screencasts, and audio books, as well as thousands of careers and success stories. Now, twenty years later, this new edition re-examines what it means to be a modern programmer. Topics range from personal responsibility and career development to architectural techniques for keeping your code flexible and easy to adapt and reuse. Read this book, and you’ll learn how to: Fight software rot Learn continuously Avoid the trap of duplicating knowledge Write flexible, dynamic, and adaptable code Harness the power of basic tools Avoid programming by coincidence Learn real requirements Solve the underlying problems of concurrent code Guard against security vulnerabilities Build teams of Pragmatic Programmers Take responsibility for your work and Career Test ruthlessly and effectively, including property-based testing Implement the Pragmatic Starter Kit Delight your users Written as a series of self-contained sections and filled with classic and fresh anecdotes, thoughtful examples, and interesting analogies, The Pragmatic Programmer illustrates the best approaches 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 development habits and attitudes that form the foundation for long-term success in your career. You’ll become a Pragmatic Programmer. Register your book for convenient access to downloads, updates, and/or corrections as they become available. See inside book for details.
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 requires careful mapping of existing code to terms like Matt Weisfeld’s ‘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.

Software Architect’s Handbook—Joseph Ingeno 2018-08-30 A comprehensive guide to exploring software architecture concepts and implementing best practices Key Features Enhance your skills to grow your career as a software architect Design efficient software architectures using patterns and best practices Learn how software architecture relates to an organization as well as software development methodology Book Description The Software Architect’s Handbook is a comprehensive guide to help developers, architects, and senior programmers advance their career in the software architecture domain. This book takes you through all the important concepts, right from design principles to different considerations at various stages of your career in software architecture. The book begins by covering the fundamentals, benefits, and purpose of software architecture. You will discover how software architecture relates to an organization and its constituents, as well as how to evaluate software architectures that evolve as the organization evolves. Once you have covered the basics, you will explore design patterns, best practices, and paradigms for efficient software development. The book discusses which factors you need to consider for performance and security enhancements. You will learn to write documentation for your architectures and make appropriate decisions when considering DevOps. In addition to this, you will explore how to design legacy applications before understanding enhancements. You will learn to write documentation for your architectures and make appropriate decisions when considering DevOps. In addition to this, you will explore how to design legacy applications before understanding enhancements. You will learn to write documentation for your architectures and make appropriate decisions when considering DevOps.

Adaptive Code—Gary McLean Hall 2017-04-18 Write code that can adapt to changes. By applying this book’s principles, you can create software that can evolve more quickly and easily, with fewer errors, because it doesn’t need to be changed every time. Revised, updated, and expanded, Adaptive Code, Second Edition adds indispensable practical insights on Kanban, dependency injection, and creating reusable abstractions. Drawing on over a decade of Agile consulting and development experience, McLean Hall has updated his best-seller with new insights about refactoring, pure dependency injection, and more. Master powerful new ways to: • Write code that enables and complements Scrum, Kanban, or any other Agile framework • Develop code that can survive major changes in requirements • Plan for adaptability by using dependencies, layers, interfaces, and design patterns • Perform unit testing and refactoring in tandem, gaining more value from both • Use the “golden master” technique to make your code more resilient • Build SOLID code with simple SOLID guidelines, frameworks, and patterns • Create smaller interfaces to support more-diverse client and architectural needs • Leverage dependency injection best practices to improve code adaptability • Apply dependency injection with the Stairway pattern, and avoid related anti-patterns About You This book is for programmers of all skill levels seeking more-practical insight into design patterns, SOLID principles, unit testing, refactoring, and related topics. Most readers will have programmed in C#, Java, C++, or similar object-oriented languages, and will be familiar with core procedural programming techniques.

Design Patterns in Ruby (Adobe Reader)—Russ Olsen 2007-12-10 Praise for Design Patterns in Ruby “Design Patterns in Ruby documents smart ways to resolve many problems that Ruby developers commonly encounter. Russ Olsen has done a great job of selecting classic patterns and augmenting these with newer patterns that have special relevance for Ruby. He clearly explains each idea, making a wealth of experience available to Ruby developers for their own daily work.” —Steve Metzker, Managing Consultant with Dominion Digital, Inc. “This book provides a great demonstration of the key ‘Gang of Four’ design patterns without resorting to overly technical explanations. Written in a precise, yet almost informal style, this book covers enough ground that even those without prior exposure to design patterns will soon feel confident applying them using Ruby. Olsen has done a great job to make a book about a classically dry subject into such an engaging and even occasionally humorous read.” —Peter Cotton, Ruby Ninja, Relevance Most design pattern books are based on C++ and Java. But Ruby is different—and the language’s unique qualities make design patterns easier to implement and use. In this book, Russ Olsen demonstrates how to combine Ruby’s power and elegance with patterns, and write more sophisticated, effective software with far fewer lines of code. After reviewing the history, concepts, and goals of design patterns, Olsen offers a quick tour of the Ruby language—enough to allow any experienced software developer to immediately utilize patterns with Ruby. The book especially calls attention to Ruby features that simplify the use of patterns, including dynamic typing, code closures, and “mixins” for easier code reuse. Fourteen of the classic “Gang of Four” patterns are considered from the Ruby point of view, explaining what problems each pattern solves, discussing whether traditional implementations make sense in the Ruby environment, and introducing Ruby-specific improvements. You’ll discover opportunities to implement patterns in just one or two lines of code, instead of the endlessly repeated boilerplate that conventional languages often require. Design Patterns in Ruby also signals patterns that have arisen organically instead of being borrowed from the Ruby community. These include ways to create custom objects with metaprogramming, as well as the ambitious Rails-based “Convention Over Configuration” pattern, designed to help integrate entire applications and frameworks.

Refactoring to Patterns—Joshua Kerievsky 2004-08-05 In 1994, Design Patterns changed the landscape of object-oriented development by introducing classic solutions to recurring design problems. In 1999, Refactoring revolutionized design by introducing an effective process for improving code. With the highly anticipated Refactoring to Patterns, Joshua Kerievsky has changed our approach to design by forever uniting patterns with the evolution of the codebase. Theory and practices from this book intersect to produce refactoring: sequences of low-level refactorings that allow designers to safely move designs to, towards, or away from pattern implementations. Using code from real-world projects, Kerievsky documents the thinking and steps underlying over two dozen pattern-based design transformations. Along the way he offers insights into pattern differences and how to implement patterns in the simplest possible ways. Coverage includes: A catalog of twenty-seven pattern-directed refactorings, featuring real-world code examples Descriptions of twelve design smells that indicate the need for this book’s refactorings General insights and new insights about patterns and refactoring Detailed implementation mechanics: how low-level refactorings are combined to implement high-level patterns Multiple ways to implement the same pattern—and when to use each Practical ways to get started even if you have little experience with patterns or refactoring Refactoring to Patterns reflects three years of refinement and the insights of more than sixty software engineering thought leaders in the global patterns, refactoring, and agile development communities. Whether you’re focused on legacy or “greenfield” development, this book will make you a better software designer by helping you to learn how to make intelligent design changes safely and effectively.

Modern C++ Design—Andrei Alexandrescu 2001 Presents a collection of reusable design artifacts, called generic
components, together with the techniques that make them possible. The author describes techniques for policy-based design, partial template specialization, typelists, and local classes, then goes on to implement generic components for smart pointers, object factories, functor objects, the Visitor design pattern, and multimethod engines. c. Book News Inc.

Professional Joomla!: Dan Rahmel 2007-10-01 Provides information on using the open source content management system used to manage data on the World Wide Web, covering such topics as creating Joomla! expressions, using Ajax technology, and adopting design patterns, and incorporating source code control.

Object-Oriented Analysis and Design with Applications: Grady Booch 2007-04-30 Object-Oriented Design with Applications has long been the essential reference to object-oriented technology, which, in turn, has evolved to join the mainstream of industrial-strength software development. In this third edition—the first revision in 13 years—readers can learn to apply object-oriented methods using new paradigms such as Java, the Unified Modeling Language (UML) 2.0, and .NET. The authors draw upon their rich and varied experience to offer improved methods for object development and numerous examples that tackle the complex problems faced by software engineers, including systems architecture, data acquisition, cryptoanalysis, control systems, and Web development. They illustrate essential concepts, explain the method, and show successful applications in a variety of fields. You’ll also find pragmatic advice on a host of issues, including classification, implementation strategies, and cost-effective project management. New to this new edition are An introduction to the new UML 2.0, from the notation’s most fundamental and advanced elements with an emphasis on key changes New domains and contexts A greatly enhanced focus on modeling—as eagerly requested by readers—with five chapters that each delve into one phase of the overall development lifecycle. Fresh approaches to reasoning about complex systems An examination of the conceptual foundation of the widely misunderstood fundamental elements of the object model, such as abstraction, encapsulation, modularity, and hierarchy How to allocate the resources of a team of developers and manage the risks associated with developing complex software systems An appendix on object-oriented programming languages This is the seminal text for anyone who wishes to use object-oriented technology to manage the complexity inherent in many kinds of systems. Sidebar Preface Acknowledgements About the Authors Section I: Concepts Chapter 1: Complexity Chapter 2: The Object Model Chapter 3: Classes and Objects Chapter 4: Classification Section II: Method Chapter 5: Notation Chapter 6: Process Chapter 7: Pragmatics Chapter 8: System Architecture: Satellite-Based Navigation Chapter 9: Control System: Traffic Management Chapter 10: Artificial Intelligence: Cryptanalysis Chapter 11: Data Acquisition: Weather Monitoring Station Chapter 12: Web Application: Vacation Tracking System Appendix A: Object-Oriented Programming Languages Appendix B: Further Reading Notes Glossary Classified Bibliography Index

Test-driven Development: Kent Beck 2003 Write clean code that works with the help of this groundbreaking software method. Example-driven teaching is the basis of Beck’s step-by-step instruction that will have readers using TDD to further their projects.

The Design Patterns Smalltalk Companion: Sherman R. Alpert 1998 In this new book, intended as a language companion to the classic Design Patterns , noted Smalltalk and design patterns experts implement the 23 design patterns using Smalltalk code. This approach has produced a language-specific companion that tails the topic of design patterns to the Smalltalk programmer. The authors have worked closely with the authors of Design Patterns to ensure that this companion volume meets the same quality standards that made the original a bestseller and indispensable resource. The full source code will be available on the AWL web site.

Design Patterns in C#: Vaskaran Sarcar 2018-06-21 Get hands-on experience with each Gang of Four design pattern using C#. For each of the patterns, you’ll see at least one real-world scenario, a coding example, and a complete implementation including output. In the first part of Design Patterns in C#, you will cover the 23 Gang of Four (GoF) design patterns, before moving onto some alternative design patterns, including the Simple Factory Pattern, the Null Object Pattern, and the MVC Pattern. The final part winds up with a conclusion and criticisms of design patterns with chapters on anti-patterns and memory leaks. By working through easy-to-follow examples, you will understand the concepts in depth and have a collection of programs to port over to your own projects. Along the way, the author discusses the different creational, structural, and behavioral patterns and why such classifications are useful. In each of these chapters, there is a Q&A session that clears up any doubts and covers the pros and cons of each of these patterns. He finishes the book with FAQs that will help you consolidate your knowledge. This book presents the topic of design patterns in C# in such a way that anyone can grasp the idea. What You Will Learn Work with each of the design patterns Implement the design patterns in real-world applications Select an alternative to these patterns by comparing their pros and cons Use Visual Studio Community Edition 2017 to write code and generate output Who This Book Is For Software developers, software testers, and software architects.

Design Patterns: Erich Gamma 2003-09