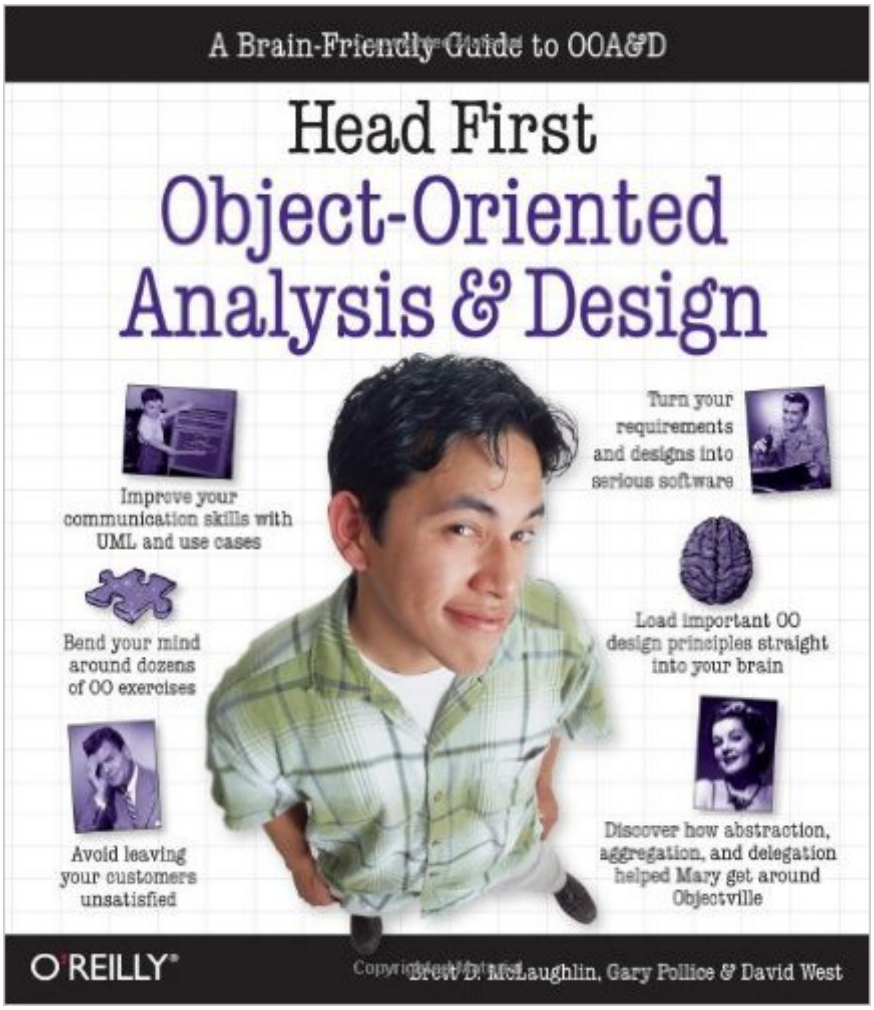
Head First Object-Oriented Analysis And Design



Synopsis

"Head First Object Oriented Analysis and Design is a refreshing look at subject of OOAD. What sets this book apart is its focus on learning. The authors have made the content of OOAD accessible and usable for the practitioner." --Ivar Jacobson, Ivar Jacobson Consulting "I just finished reading HF OOA&D and I loved it! The thing I liked most about this book was its focus on why we do OOA&D-to write great software!" --Kyle Brown, Distinguished Engineer, IBM "Hidden behind the funny pictures and crazy fonts is a serious, intelligent, extremely well-crafted presentation of OO Analysis and Design. As I read the book, I felt like I was looking over the shoulder of an expert designer who was explaining to me what issues were important at each step, and why." --Edward Sciore, Associate Professor, Computer Science Department, Boston College Tired of reading Object Oriented Analysis and Design books that only makes sense after you're an expert? You've heard OOA&D can help you write great software every time-software that makes your boss happy, your customers satisfied and gives you more time to do what makes you happy. But how? Head First Object-Oriented Analysis & Design shows you how to analyze, design, and write serious object-oriented software: software that's easy to reuse, maintain, and extend; software that doesn't hurt your head; software that lets you add new features without breaking the old ones. Inside you will learn how to: Use OO principles like encapsulation and delegation to build applications that are flexible Apply the Open-Closed Principle (OCP) and the Single Responsibility Principle (SRP) to promote reuse of your code Leverage the power of design patterns to solve your problems more efficiently Use UML, use cases, and diagrams to ensure that all stakeholders are communicating clearly to help you deliver the right software that meets everyone's needs. By exploiting how your brain works, Head First Object-Oriented Analysis & Design compresses the time it takes to learn and retain complex information. Expect to have fun, expect to learn, expect to be writing great software consistently by the time you're finished reading this!

Book Information

Series: Head First
Paperback: 636 pages
Publisher: O'Reilly Media; 1 edition (December 7, 2006)
Language: English
ISBN-10: 0596008678
ISBN-13: 978-0596008673
Product Dimensions: 8 x 1.4 x 9.2 inches

Shipping Weight: 2.6 pounds (View shipping rates and policies)
Average Customer Review: 3.7 out of 5 starsÂ Â See all reviewsÂ (71 customer reviews)
Best Sellers Rank: #53,555 in Books (See Top 100 in Books) #4 inÂ Books > Computers & Technology > Programming > Software Design, Testing & Engineering > UML #22 inÂ Books > Textbooks > Computer Science > Object-Oriented Software Design #31 inÂ Books > Computers & Technology > Computer Science > Systems Analysis & Design

Customer Reviews

I like the Head First series, and even Head Rush, for its innovative and fun approach for introductory software topics. I've had small concerns on all of them but I have never been as ambivalent as I have for this book. I know a big part of this problem was that it was rewritten expeditious (I am still not sure of the reason why) and it shows throughout the book with spelling, logic and code errors.You can tell that the first chapter was rushed. There are several spelling and programming mistakes. The most egregious is where they ask you to look through some code to find what "FIRST" you change and then they answer that question with a much smaller problem (the main problem was they forgot to add a return statement (pg.5) and they write about the inconsistency of using String based searching). It has also been mentioned by several reviewers of the use of the method name "matches" which only makes sense for regex not for an equals operation. I also did not like the search example (how can you not think of price in a search?). The best part of this chapter is the mantra that should be practiced by many engineers: "Make sure your software does what the customer wants it to do."The next few chapters are definitely better (though still some spelling mistakes). They are a good read for beginners and intermediate programmers on gathering requirements, change of these requirements and analysis. The ideas are a bit simplistic though it is good to get many programmers used to the idea of UML and use cases and using them to drive requirement gathering and textual analysis. Intermediate and advanced readers familiar with use cases will gain more from reading Alistair Cockburn's "Writing Effective Use Cases" (or will already have read it) and for further UML reading should go with "UML Distilled" by Martin Fowler.When the book gets back to design I see some problems with the coding. The designer has this bizarre idea of abstracting all properties (under the guise of "encapsulate what varies") into a Map attribute to lessen the amount of subclasses for instruments. While initially this may seem a good idea it gets rid of all type-safe coding (you can now safely assign an instrument type to a backwood for the instrument), you cannot have behavior from the instruments (this is mentioned in the book) and if you put a property with one letter misspelled or capitalized out-of-place you now have a bug, one

that you might have trouble finding thereby increasing maintenance costs. Too much flexibility makes the code ambiguous.After design, the studies get to solving really big problems, architecture, design principles, and iterating and testing. These chapters I enjoyed much more especially the chapter on design principles with the beginning mantra that "Originality is Overrated." This chapter goes over basic principles such as OCP (open-closed principle), DRY (don’t repeat yourself), SRP (single responsibility principle) and LSP (Liskov Substitution Principle).Then the book last chapter (the ooa&d lifecycle) sums the lessons in the book in one large (somewhat contrived but these type of examples always are) program for the Objectville Subway. Then two terse appendixes dealing with ten additional OOA&D topics and OO concepts should make the reader realize that this book is just an introductory sliver of what needs to be learned for a sagacious software acumen.This book is useful for programmers with a bit of Java (or C#) knowledge who want to get a good overview of OOA&D. This book is useful because it teaches important OO vernacular and a simple holistic approach to iterative development. If the book did not have a "quickly done" feeling, better design and fewer mistakes I would have liked this book more. This book is a good candidate for a second edition. If you want a more thorough explanation of these topics I recommend "The Object Primer" by Scott Ambler as one of my favorite books for a good introduction to OOA&D.

Am I really the first to write a review on this new installment? Well, let me start with a huge five stars for this new addition to the Head First series!I had been waiting for this book to hit the shelves a while, since I absolutely loved the innovative approach of the Head First Design patterns book. This one was no different in the way it clearly and creatively presented key principles to good object-oriented design and educated the reader on how to approach designing software for the real world from requirements gathering all the way to anticipating and designing for change.A few things about this book - in my opinion, there is probably no better way to present the world of software design to a beginner. Instead of talking about abstract concepts, the writers present the material using concrete scenarios, and through-out the book, the reader is encouraged heavily to think through the pitfalls and problems and come up with solutions - there is no better way to learn. There are lots of exercises and even specific places to write ones ideas down.Some topics covered are of course good object oriented principles like encapsulation and delegation, requirements gathering, use cases, anticipating changes, class diagrams, UML and more. The book only briefly touches (but does not go into too much detail) on state diagrams, sequence diagrams, unit testing and other concepts which are a huge part of software design, in the last chapter. While it does not go into these subjects deeply, it does not leave the reader completely without any knowledge on these

topics either.It does cover more than enough to enable a reader to become very well versed in architectural principles. Best of all, the information is presented in a way where it will stick forever. The whole point is not to cover everything there is to know, but for you to really GET IT, on what is truly crucial to know.This book is not for seasoned architects or for those who do not appreciate comical diagrams and pictures on every page (Even though, anyone with a sense of humor would appreciate the fun way the information was presented). If however, someone is confused about object oriented design and has only heard the buzz worlds but doesn't know how it all comes together - this book will ensure that they are never confused again. Not only that, but after reading this book, they will be armed with tried and tested principles of experience of what kind of design works for long term solutions vs what is a nightmare.Another thing to mention is that all the code examples in this book are in Java (as all Head First books are). This is certainly not a problem, even if you do not code in Java, because the principles are the same no matter the language, and C# .NET users in particular will not have any problems following the code examples.The book does assume prior programming knowledge of an object oriented language in order to follow the code. This is not a book to learn how to write code in a programming language. It teaches how to design and architect your project, with the whole software life-cycle in mind.There are a few useful appendixes in the back to quickly define and explain the basics of software design elements used in the book (like UML for instance).Overall, it is a great book for anyone interested in software design principles! Best of all, you will get through this book QUICKLY, because with the creative and fun way that you will be learning, it will be hard to put down.

First off, I'm already a fan of the Head First series - especially the Head First Design Patterns book. This book follows the same entertaining style and keeps your attention page after page. To me, there are two kinds of Head First books, ones relating to technologies like Java, Servlets & JSPs, EJB, etc and ones that cover a more traditionally academic topics like Design Patterns and this book, OO Analysis and Design. Personally, I like the Head First treatment on the academic topics better than the others. So, if you weren't a fan of Head First Java (for example) you might want to give this book (or the Design Patterns one) a try.Specifically for this book - I really liked the chapter layout and the progression as each chapter builds upon the next. The chapters explain the basics of OO principles, ease you into Use Cases and how to write good ones, and continues building upon OO Design principles. When the Head First Design Patterns book came out, we purchased a bunch for the office and held a few "lunch and learn" classes on design patterns for the team at work. I can easily see doing the same thing with this book, as the Head First books make it easy to use as

instructional manuals as well.If you have found other books (lectures, articles, etc) on OO Analysis and Design a bit intimidating or conceptually difficult to grasp, this is the book for you.

Object Success : A Manager's Guide to Object-Oriented Technology And Its Impact On the Corporation (Object-Oriented Series) Head First Object-Oriented Analysis and Design Reusable Software : The Base Object-Oriented Component Libraries (Prentice Hall Object-Oriented Series) Visual Object-Oriented Programming Using Delphi With CD-ROM (SIGS: Advances in Object Technology) Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd Edition) Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process (2nd Edition) Object-Oriented Analysis and Design for Information Systems: Modeling with UML, OCL, and IFML UML and the Unified Process: practical object-oriented analysis and design Systems Analysis and Design: An Object-Oriented Approach with UML Systems Analysis and Design with UML Version 2.0: An Object-Oriented Approach Object-Oriented Analysis and Design with the Unified Process (Available Titles CengageNOW) An Object-Oriented Approach to Programming Logic and Design Tools For Structured and Object-Oriented Design (7th Edition) Object-Oriented Modeling and Design with UML (2nd Edition) Practical Object-Oriented Design in Ruby: An Agile Primer (Addison-Wesley Professional Ruby) Design Patterns CD: Elements of Reusable Object-Oriented Software (Professional Computing) Object Oriented Design with Ada: Maximizing Reusability for Real-Time Systems Fundamentals of Object-Oriented Design in UML Design Patterns: Elements of Reusable Object-Oriented Software Design Patterns: Elements of Reusable Object-Oriented Software (Adobe Reader)