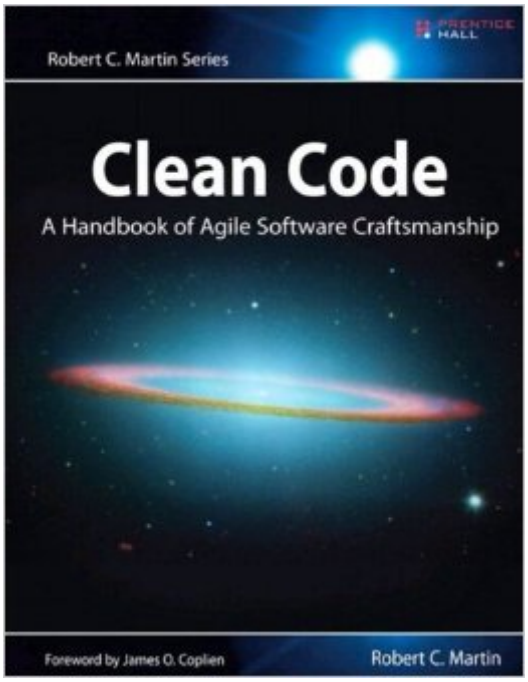
Clean Code: A Handbook Of Agile Software Craftsmanship



Synopsis

Even bad code can function. But if code isnâ ™t clean, it can bring a development organization to its knees. Every year, countless hours and significant resources are lost because of poorly written code. But it doesnâ ™t have to be that way. Noted software expert Robert C. Martin presents a revolutionary paradigm with Clean Code: A Handbook of Agile Software Craftsmanship . Martin has teamed up with his colleagues from Object Mentor to distill their best agile practice of cleaning code â œon the flyâ • into a book that will instill within you the values of a software craftsman and make you a better programmerâ ”but only if you work at it. What kind of work will you be doing? Youâ ™ll be reading codeâ ”lots of code. And you will be challenged to think about whatâ ™s right about that code, and whatâ ™s wrong with it. More importantly, you will be challenged to reassess your professional values and your commitment to your craft. Clean Code is divided into three parts. The first describes the principles, patterns, and practices of writing clean code. The second part consists of several case studies of increasing complexity. Each case study is an exercise in cleaning up codeâ ”of transforming a code base that has some problems into one that is sound and efficient. The third part is the payoff: a single chapter containing a list of heuristics and â œsmellsâ • gathered while creating the case studies. The result is a knowledge base that describes the way we think when we write, read, and clean code. Readers will come away from this book understanding How to tell the difference between good and bad code How to write good code and how to transform bad code into good code How to create good names, good functions, good objects, and good classes How to format code for maximum readability How to implement complete error handling without obscuring code logic How to unit test and practice test-driven development This book is a must for any developer, software engineer, project manager, team lead, or systems analyst with an interest in producing better code.

Book Information

Paperback: 464 pages

Publisher: Prentice Hall; 1 edition (August 11, 2008)

Language: English

ISBN-10: 0132350882

ISBN-13: 978-0132350884

Product Dimensions: 6.9 x 1.1 x 9.1 inches

Shipping Weight: 1.5 pounds (View shipping rates and policies)

Average Customer Review: 4.4 out of 5 starsÂ Â See all reviewsÂ (426 customer reviews)

Best Sellers Rank: #3,585 in Books (See Top 100 in Books) #1 inÂ Books > Computers & Technology > Programming > Software Design, Testing & Engineering > Testing #2 inÂ Books > Computers & Technology > Programming > Software Design, Testing & Engineering > Software Development #2 inÂ Books > Textbooks > Computer Science > Software Design & Engineering

Customer Reviews

When you do code maintenance, you can really "love" or "hate" a person that you do not even know just by the code he or she has written. Messy code almost always goes hand in hand with lower productivity, lower motivation, and a higher number of bugs. In the first chapter, Robert C. Martin presents in a very instructive way, the opinion from very well-known personalities about what "clean code" is, and also suggests we apply the Boy Scout Rule (Leave the campground cleaner that you found it) to our code. The following chapters present practical advice about how to do this cleaning (or even better, how to avoid the mess in the first place).The suggestions presented in the book (meaningful names, pertinence of comments, code formatting, etc) may sound very familiar to any experienced programmer but they are presented with such a level of detail and with very illustrative examples that it is almost impossible not to learn valuable things chapter by chapter. All the examples are in Java, but the guidelines they illustrate can be applied, in most of the cases, to other languages.The most challenging chapter to read (but also a very valuable one) was the Refactoring of the class SerialDate (from the JCommon library). It is a real-life example and the author shows step-by-step what it takes to do refactoring. The last chapter, "Smells and Heuristics" makes a very good closure presenting in categories and in a condensed way, potential problems and suggested ways to solve/mitigate them.I enjoyed reading this book and after finishing it, I decided to apply the Boy Scout Rule. I took a module written in a procedural language and not only managed to improve the clarity of the code, but also reduced the number of lines from more than 1,100 to 650. The next person to touch this code will certainly be happy to deal with cleaner code!

In response to the "Don’t get the Kindle version" review -- the problems appear to be resolved now. It looks fine on my Kindle 3 (aside from using proportional fonts in code examples, but this isn’t too bad once you get used to it). Also, as many other reviewers have pointed out, this is an excellent book. I really wish all of my colleagues, predecessors, and managers had read it. My job would be so much easier and more enjoyable if they had.

This book is good at providing a general overview of what it means to be a software professional.

Lots of good advice and provides many resources and a general framework for thinking about the subjects he presents.Sometimes the author presents strategies very specific to him that wouldn't work for me. For example, I tried the pomodoro method before and had mixed results. I think readers would benefit more looking at the goal (better time management) and finding a methodology that works for them to accomplish that goal.He is very bullish on unit tests, stating that there is no longer and controversy over TDD. As a huge fan of unit tests, I find many places I have worked at have very little interest in unit testing or don't see any real benefit.The book is also very strongly against being in the Flow to program which I found interesting. This is pretty much 100% the opposite of everything else I have ever heard/read.He is also against listening to music while programming. He provides a weird example where while listening to Pink Floyd his code comments had Pink Floyd references. The author has a tendency to confuse something that is true for him ("I don't listen to music while programming") to a general universal rule ("Programmers shouldn't listen to music while programming").Most programmers I know who listen to music do so as white noise. For instance, I listen to techno many times while programming. I don't like techno but the droning drum servies to drown out the office chitter chatter at my current gig.Like Clean Code, I don't always agree with the author but provides good food for thought and is worth the read!

When most people hear the term "bad writing" they understand the term: Confusing, inconsistent, rambling, big words used incorrectly.In fact, we have lots and lots of educational programs designed to teach grammar, composition, journalism, and fiction. Master's Degrees in the subject, even.But for software development we seemed obsessed with "architecture" (whatever that means), process and patterns.In this book, Bob Martin takes a specific stab at what good code looks like. He provides rules, examples, and even sample transformations.It is not an easy book. If you are a new developer, you can invest a lot of time and energy into really absorbing the concepts and practicing them yourself. If you are more senior, you may disagree, you may struggle, you may toss the book in a corner and yell at it ...But then you'll pick it back up again. And you will be a better developer for it.One thing that I struggle with about the traditional CS cirricula is that so little attention is spent on maintenance, which is the vast majority of actual development time. This book presents an aesthetic and the skills to write maintainable code. If you teach software development, you'll want to use this book in your courses.Student, Journeyman, Master, or Instructor - A book like this belongs on your bookshelf. Follow the advice in it, or have an explanation why not - either way you'll be a strong developer.Of course, there are other books in this area. What struck me about this one is the quality of the writing; it is truly engaging and -- a little inspiring. That quality is so rare in technical books that

I give this one five stars.

Agile: Agile Project Management CherryTree Style Guide(Scrum,Agile Scrum,agile methodology,Agile development,agile coaching,agile leader,agile methods,scrum master certification,agile introduction) Agile Product Management: (Box Set) Agile Estimating & Planning Your Sprint with Scrum and Release Planning 21 Steps (agile project management, agile software ... agile scrum, agile estimating and planning) Agile Project Management: Box Set - Agile Project Management QuickStart Guide & Agile Project Management Mastery (Agile Project Management, Agile Software Development, Agile Development, Scrum) Agile Estimating & Planning Your Sprint with Scrum (agile project management, agile software development, agile development, agile scrum, agile estimating and planning) Clean Code: A Handbook of Agile Software Craftsmanship Agile Project Management: QuickStart Guide - The Simplified Beginners Guide To Agile Project Management (Agile Project Management, Agile Software Development, Agile Development, Scrum) Agile Project Management: & Scrum Box Set - Agile Project Management QuickStart Guide & Scrum QuickStart Guide (Agile Project Management, Agile Software ... Scrum, Scrum Agile, Scrum Master) Agile Product Management: (Box Set) : Scrum: A Cleverly Concise Agile Guide and Agile: The Complete Overview of Agile Principles and Practices (scrum, ... development, agile software development) Agile Project Management: Mastery - An Advanced Guide To Agile Project Management (Agile Project Management, Agile Software Development, Agile Development, Scrum) Agile Project Management: Agile Revolution, Beyond Software Limits: A Practical Guide to Implementing Agile Outside Software Development (Agile Business Leadership, Book 4) Agile Project Management: For Beginners - A Brief Introduction to Learning the Basics of Agile Project Management (Agile Project Management, Agile Software Development, Scrum) Agile Product Management: (Box Set): Agile Estimating & Planning Your Sprint with Scrum & User Stories 21 Tips (scrum, scrum master, agile development, agile software development) Agile Project Management: An Inclusive Walkthrough of Agile Project Management (Agile Project Management, Agile Software Developement, Scrum, Project Management) Agile Project Management: The Agile PMO: Leading the Effective, Value Driven and Agile Project Management Office (Agile Business Leadership Book 1) Agile Product Management (Box Set): User Stories & Product Backlog 21 Tips (scrum, scrum master, agile development, agile software development) Agile Product Management: User Stories: How to capture, and manage requirements for Agile Product Management and Business Analysis with Scrum (scrum, ... development, agile software development) Agile Product Management: Product Vision:: 21 Steps to setting excellent goals for your product (scrum, scrum master, agile

development, agile software development) Agile Product Management: User Stories & Product Backlog 21 Tips (scrum, scrum master, agile development, agile software development) Agile Product Management: Release Planning: 21 Steps to plan your product releases from a product vision with Scrum (scrum, scrum master, agile development, agile software development) Agile Product Management (Box Set): Product Backlog 21 Tips , Release Planning 21 Steps (scrum, scrum master, agile development, agile software development)