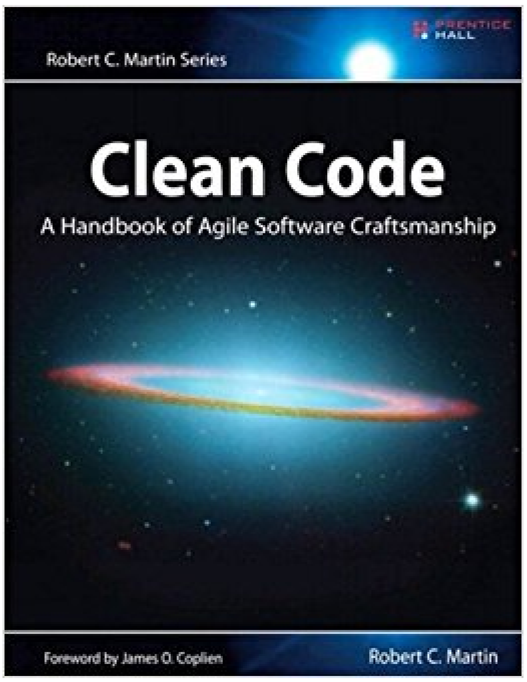
Clean Code: A Handbook Of Agile Software Craftsmanship


Robert C. Martin Series
PRENTICE HALL
Clean Code
A Handbook of Agile Software Craftsmanship
Foreword by James O. Coplien
Robert C. Martin

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: 7 x 1.1 x 9.1 inches

Shipping Weight: 1.7 pounds (View shipping rates and policies)

Average Customer Review: 4.5 out of 5 stars   355 customer reviews

Best Sellers Rank: #2,147 in Books (See Top 100 in Books)   #1 inÂ Books > Computers & Technology > Programming > Software Design, Testing & Engineering > Testing   #2 inÂ Books > Textbooks > Computer Science > Software Design & Engineering   #3 inÂ Books > Textbooks > Computer Science > Programming Languages

Customer Reviews

Even bad code can function. But if code isn’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 doesn’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 "on the fly" into a book that will instill within you the values of a software craftsman and make you a better programmer--but only if you work at it. What kind of work will you be doing? You’ll be reading code--lots of code. And you will be challenged to think about what’s right about that code, and what’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 code--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 "smells" 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 understandingHow to tell the difference between good and bad codeHow to write good code and how to transform bad code into good codeHow to create good names, good functions, good objects, and good classesHow to format code for maximum readabilityHow to implement complete error handling without obscuring code logicHow to unit test and practice test-driven developmentThis book is a must for any developer, software engineer, project manager, team lead, or systems analyst with an interest in producing better code.

Robert C. â œUncle Bobâ • Martin has been a software professional since 1970 and an international software consultant since 1990. He is founder and president of Object Mentor, Inc., a team of experienced consultants who mentor their clients worldwide in the fields of C++, Java, C#, Ruby, OO, Design Patterns, UML, Agile Methodologies, and eXtreme programming.

I’ve been programming about 17 years and consider myself above average programmer. Yet, this book made me feel like I’m actually horrible coder. I’ve always gotten my tasks done but I didn’t pay attention on refactoring to clean up the code. I’m already behind and got a demo coming up in few days. As I’m reading my guilty verdicts on all his ’bad code’ examples, it inspire me to care about ’coding’ yet again. It can be fun and it’s not all about getting the job done. What’s sad about the reality is that ’bad code’ will continually increase over time because people don’t realize what ’bad code’ can do in the long run.Every programmer regardless of experience should read this book. Thanks!

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!

My code reads so much better since I started implementing the books guidelines. Gone over the book twice since I liked it so much. Plus with functional programming becoming more prevalent, the book’s suggestions align nicely with this style.

We’re going through this book at work as a book club, and I can say there is a reason why the name of this book is thrown around frequently. It is written by several industry veterans, and the takeaways from the book are immediately reflecting our code.Our code is now much more readable and intelligently structured. If you want a quick fix to a low quality bar of your team’s software, purchase this book and go through each section as a group.

"Clean Code" informs developers how they can write code that is more readable and maintainable.The first section of the book covers identifying confusing code and rewriting it for topics such as variables, classes, and concurrency algorithms. These chapters contain snippets of code before and after some thought was taken to make the code’s intent clear. Although the examples are all in Java, the principles could be applied to most languages.The second section contains case studies of taking source code and applying the ideas from the first section to contrast how much code can be more readable and maintainable. It was difficult to flip back and forth between pages to follow the differences, but the case studies succeed in showing iterative improvement in the source code.The final section is a short summary identifying when code might need re-factoring to improve readability.I believe "Clean Code" does a great job showing how code quality improves when someone writes or rewrites code with the will to make it better. Before reading this book, I applied some of the ideas from Robert Martin and the other authors, but after reading the book, I learned a lot how to write better code that I apply to my programming today.Just asÂ Don’t Make Me Think: A Common Sense Approach to Web Usability, 2nd EditionÂ shows how poor web usability can confuse users and how to address it, "Code Clean" shows how poor code readability can confuse developers and how to address it.

I liked this book. I am still trying to get through some parts, but overall I am pleased with author. I am trying to learn the principles of computer programming and to understand all the intricate details. This book is a good start for a junior developer.

best book a software engineer could read. it changes your way of coding right away. suggestions may seem simple, but they are not simple at all, they are very effective and useful.

In Clean Code Robert Martin lays out a set of clear, well-reasoned software development heuristics. I believe that by applying the techniques described in this book developers will deliver cleaner, more expressive and more maintainable code. As an added benefit, the book is concise and easy to read, in contrast to many other books on software development. While I disagree with some of the stances of the author, I take a more tempered view of code comments for example, I consider the book essential reading for software developers.I have read the book multiple times and recommend it to every developer on my team.

Download to continue reading...

Contact Us