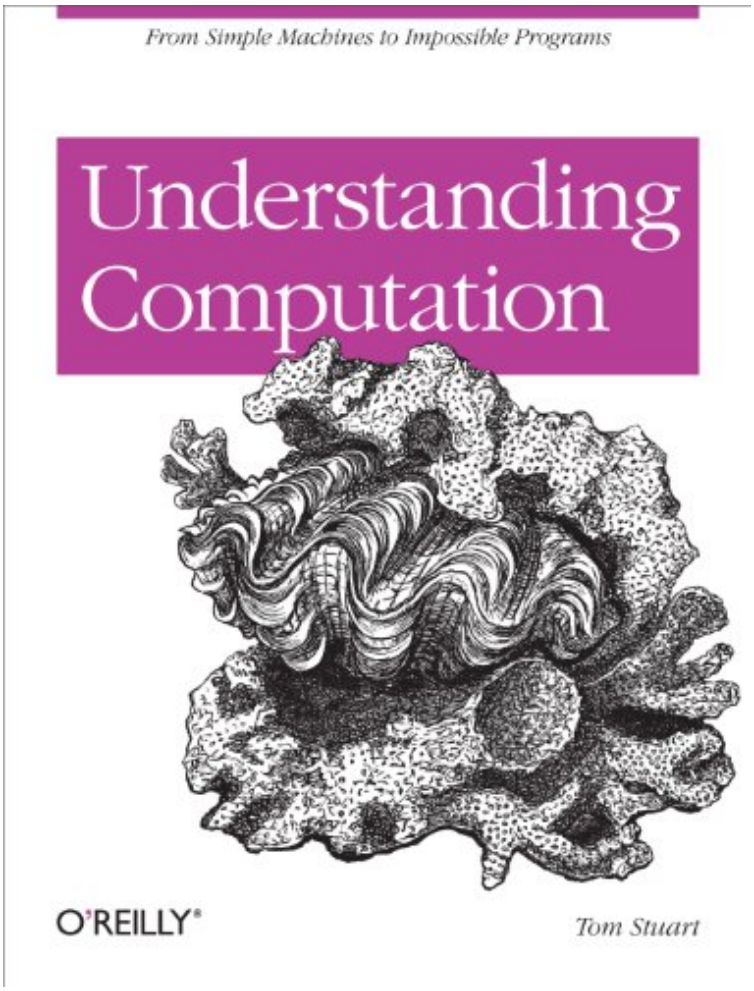
Understanding Computation: From Simple Machines to Impossible Programs

Von Tom Stuart
*DOC | *audiobook | ebooks | Download PDF | ePub*



Produktinformation -Verkaufsrang: #371694 in eBooksVerffentlicht am: 2013-05-15Erscheinungsdatum: 2013-05-15File Name: B00CT3C4IM | File size: 58.Mb

Von Tom Stuart : Understanding Computation: From Simple Machines to Impossible Programs before purchasing it in order to gage whether or not it would be worth my time, and all praised Understanding Computation: From Simple Machines to Impossible Programs:

KundenrezensionenHilfreichste Kundenrezensionen1 von 1 Kunden fanden die folgende Rezension hilfreich. The Missing Link: What the Computer Science Professors Left OutVon Dave CottlehuberUnderstanding Computation (From Simple Machines to Impossible Programs), by Tom Stuart, is an absolute gem of a book. It's described as:"Finally, you can learn computation theory and programming language design in an engaging, practical way. Understanding Computation explains theoretical computer science in a context youll recognize, helping you appreciate why these ideas matter and how they can inform your day-to-day programming."And it does exactly that. If you're a

self-taught programmer, or a hobbyist who wants to learn more about the theoretical side of computer science, without drowning in research papers, I cannot recommend a better book. Using the ruby language as the starting point, the author moves concisely and clearly through parsers, DFA and NFA, regular expressions, and alone for these components would be worth recommending. Readers will get a solid understanding of how these common and critical components of most programming languages work, without actually getting lost in the complex source code of a full real-world implementation in a modern day language.I don't recall another book that manages to treat Turing Machines in such a practical way, and expands into the lambda calculus, and the final section of the book covers core computer science concepts such as non-computability (the halting problem) in a way that will bring relief to many future computer science students.With its absolute clarity, practical and engaging style, this book will be a classic for years to come.1 von 1 Kunden fanden die folgende Rezension hilfreich. Great book if you speak Ruby or AutomataVon Christopher KamperI bought this book after the rave reviews it received from the Ruby Rogues Podcast (does not allow links, so you have to search for it) - and so far, I haven't been disappointed! I'm about halfway through and many of the concepts I had previously learned the mathematical way became much, much clearer still. I'm just not sure exactly if this book is really aimed at readers with "no formal background in mathematics or computer science" as the author suggests. Knowing about Turing machines, semantics, grammars and such certainly helped me through some of the denser parts.I have to confess that my Ruby isn't all that great though and I have to read and re-read the code blocks a lot. My verdict would be that you should speak Ruby quite well _or_ have some basic understanding of computation before you pick this one up. If you lack both, this book might be a little too hard for an enjoyable read.

KurzbeschreibungFinally, you can learn computation theory and programming language design in an engaging, practical way. Understanding Computation explains theoretical computer science in a context youll recognize, helping you appreciate why these ideas matter and how they can inform your day-to-day programming.Rather than use mathematical notation or an unfamiliar academic programming language like Haskell or Lisp, this book uses Ruby in a reductionist manner to present formal semantics, automata theory, and functional programming with the lambda calculus. Its ideal for programmers versed in modern languages, with little or no formal training in computer science.Understand fundamental computing concepts, such as Turing completeness in languagesDiscover how programs use dynamic semantics to communicate ideas to machinesExplore what a computer can do when reduced to its bare essentialsLearn how universal Turing machines led to todays general-purpose computersPerform complex calculations, using simple languages and cellular automataDetermine which programming language features are essential for computationExamine how halting and self-referencing make some computing problems unsolvableAnalyze programs by using abstract interpretation and type systems