



The Passionate Programmer

Chad Fowler , David Heinemeier Hansson (Foreword by)

[Download now](#)

[Read Online](#) ➔

The Passionate Programmer

Chad Fowler , David Heinemeier Hansson (Foreword by)

The Passionate Programmer Chad Fowler , David Heinemeier Hansson (Foreword by)

Success in today's IT environment requires you to view your career as a business endeavor. In this book, you'll learn how to become an entrepreneur, driving your career in the direction of your choosing. You'll learn how to build your software development career step by step, following the same path that you would follow if you were building, marketing, and selling a product. After all, your skills themselves **are** a product.

The choices you make about which technologies to focus on and which business domains to master have at least as much impact on your success as your technical knowledge itself--don't let those choices be accidental. We'll walk through all aspects of the decision-making process, so you can ensure that you're investing your time and energy in the right areas.

You'll develop a structured plan for keeping your mind engaged and your skills fresh. You'll learn how to assess your skills in terms of where they fit on the value chain, driving you away from commodity skills and toward those that are in high demand. Through a mix of high-level, thought-provoking essays and tactical "Act on It" sections, you will come away with concrete plans you can put into action immediately. You'll also get a chance to read the perspectives of several highly successful members of our industry from a variety of career paths.

As with any product or service, if nobody knows what you're selling, nobody will buy. We'll walk through the often-neglected world of marketing, and you'll create a plan to market yourself both inside your company and to the industry in general.

Above all, you'll see how **you** can set the direction of your career, leading to a more fulfilling and remarkable professional life.

The Passionate Programmer Details

Date : Published June 4th 2009 by Pragmatic Bookshelf (first published January 1st 2009)

ISBN : 9781934356340

Author : Chad Fowler , David Heinemeier Hansson (Foreword by)

Format : Paperback 219 pages

Genre : Computer Science, Programming, Science, Technology, Nonfiction, Software, Technical

 [Download The Passionate Programmer ...pdf](#)

 [Read Online The Passionate Programmer ...pdf](#)

Download and Read Free Online The Passionate Programmer Chad Fowler , David Heinemeier Hansson (Foreword by)

From Reader Review The Passionate Programmer for online ebook

Mark says

This book is a great example of the state of most non-technical programming writing, in that it mostly exists to stroke the author's ego and give a pat on the head to his gen-x American colleagues.

In its original form it was a list of tips aimed to help American programmers avoid having their jobs outsourced from under them, written as a series of short blog-style chapters that are not long enough to cover their topic, let alone dive deep enough to provide any insight.

Everything the book looks at is better discussed in other books, such as *Being Geek: The Software Developer's Career Handbook* which is a far better software career advice book, and *The Pragmatic Programmer: From Journeyman to Master* from the same publishers which has much more useful technical content.

There's nothing particularly terrible about the book, but I can't think of anyone I'd recommend this to who wouldn't be better served by other materials. It's a textbook example of the vacuous echo chamber of modern technology, rehashing concepts that were better covered decades ago with a sprinkling of self-help platitudes and quotes from better authors.

Arthur Santos says

Excelente livro para desenvolvedores de softwares (ou até mesmo outras profissões), com conselhos fundamentais para manter-se atualizado, motivado e, como diz o subtítulo, construir uma carreira memorável na área.

Sobre conhecimento de tecnologias:

"Too many of us seem to believe that specializing in something simply means not knowing about other things."

"Muitos (desenvolvedores de software) parecem acreditar que especializar-se em algo significa, simplesmente, não saber sobre outras coisas"

Sobre quando se é inquirido a respeito de prazos de entrega:

"Be courageous enough to be honest."

"Seja corajoso o suficiente para ser honesto."

Sobre a vantagem de saber escrever bem:

"The ability to write creates both a superficial perception of you and a real insight into how your mind works. If you can't organize your thoughts in your mother tongue so that others can clearly understand them, how can we expect that you can do it in a programming language?"

"A capacidade de escrever cria, além de uma idéia superficial sobre você, uma visão interna de como sua mente funciona. Se você não consegue organizar seus pensamentos em sua língua nativa para que outros consigam entendê-lo, como esperar que você consiga fazê-lo através de uma linguagem de programação?"

Amor ao trabalho:

"The real meat of your career is not the promotions and salary advances. It's the time you spend working toward those advances. Or, more important, it's the time you spend working regardless of the advances."

"A essência de sua carreira não são as promoções e aumentos salariais. É o tempo que você passa trabalhando para atingí-los. Ou, mais importante, é o tempo que você passa trabalhando sem se importar com estas conquistas."

Marco Emmanuel Patiño Acosta says

This is one of those books that every programmer must read eventually. It provides you a full list of tips and to-do's in your daily work that will improve your overall role in your current company. It also provides great analogies about how being a musician helped him to achieve goals and how you should not fear to lose your job against outsourcing but rather become a more valuable programmer. Also at the end of every chapter it has a full list of things to do so it just invites you to keep your hands in action from day 1.

Brian says

Let me start by saying I cringe whenever I hear people talk about what they're "passionate" about in a career sense. It's an over-used and hyperbolic adjective that's turned into yet another buzzword. If you're really "passionate" about customer service QA or something, perhaps you need to get out more. Life is short; save "passionate" for things that really deserve it.

That said, this is a great "idea" book, in that it's a fast, easy read, and each short chapter gives you a topic to think about, and see if it resonates with you. Some chapters you'll gloss over, others you might stop and think, "that's a good idea, I'll try that."

Nothing earth-shattering, but it's a good motivation if you're in a rut, in between jobs, or just looking for a quick read that relates to a software career.

André Pinto says

this looks to me like the point of view of someone that lost any passion for programming as an art/craft, to start seeing it from the typical egocentric manager point of view. I read the whole book simply to have a glimpse of what someone with this attitude really thinks. good technical managers should feel uncomfortable with the limited mindset that is promoted on the book.

there are a few interesting and useful "blog posts" but several others could possibly be part of an anti-pattern book on "programming with passion". they seem to have been written with that "don't think too much, just write something already, so you can promote yourself and go to conferences"-mentality he advocates for your own career promotion.

also, we don't really care that much for your jazz past. it seems that Chad should probably return to his jazz career, as it feels throughout the entire book that that's the one he truly loves. maybe he would be happier doing that and remember what "passionate" means.

Dillon says

Just finished this on the plane. Am happy to write this review.

This book falls broadly into two different but related categories - 1. How to be better at your job and 2. How to plan out a good career in the IT industry. I'll focus on the first part first. This review is really for myself so I'm going to summarize my key takeaways and perhaps editorialize a bit (though the book has plenty of opinions itself). I found a pleasant cohesiveness between what the book recommends and the Amazon leadership principles.

Being better at your job:

- * Focus on being an asset to the business. A computer programmer doesn't work in isolation. I am paid by a business that hired me to solve problems within a specific domain in order to positively influence the bottom line, and I'm trusted with the responsibility to thoroughly understand the trade offs and make the decisions that fit best within the business context. A software engineer is a problem solver first, and is not paid to be a technology zealot.
- * That said, be a technology zealot. Understand how technologies work from the bottom up so that you can thoroughly understand the solutions you recommend and implement. Learn different languages on the side and read up on the industry. Don't become the carpenter with only one tool that he fits into use for every job. Go deep and go wide.
- * Don't overpromise - learn to say no. Admit when you don't know. In short, this will help you earn trust.
- * Own your mistakes. Don't try to make mistakes, but don't be so afraid to own them, and when you do make a mistake, own it and start working towards a solution right away. Fail fast and never try to prolong the time between bug discovery and action.
- * Learn to communicate well with people who don't code. Assume that everyone your company hires is important to the business and when the need arises make a full effort to help them understand the problems you are working on and the solutions you are developing. Engineers are far from the only thing required for a successful team.
- * Related to the top point - learn the business. Have at least a rough understanding of where the money comes in from and how various teams contribute to business needs. This will help you make more enlightened first guesses, conversations, and final decisions when it comes to practical engineering decisions.
- * Don't silo yourself or try to make yourself "irreplaceable" through similar means (writing code only you understand) - that is, anything that limits rather than promotes good communication. What you are actually doing is making yourself a liability to your company. Single points of failure are never good.
- * Budget your time. Personally, I've found the time-blocking mechanism described here and in Deep Work to be very helpful when I know what I need to get done. Set a hard start and finish to your day. Like that dragon thing in the fantastic beasts movie, your tasks will expand and contract (within reason) to the time that you budget for them. The 40-hour week is generally a good guide. Make your 8 hours of work focused and impactful. I used to be skeptical about this point, and certainly there are some weeks where this isn't feasible, but more hours doesn't always equate to more productivity. The Effective Engineer and Rest: Why We Get a more Work Done When We Work Less also effectively drive this point home. Guard against burnout.
- * Always try and know what you need to get done. A mediocre employee needs to be told what to do, whereas a good one takes initiative and figures out what to focus on.
- * No task is beneath you. Do a good job even with the mundane tasks. Be present in whatever you're doing.
- * Learn to enjoy maintenance. Lower expectations actually provide a good opportunity to surprise and delight.
- * Learn a new language to learn to think differently. It's fun too.

* Hold yourself accountable. Every day, ask yourself if you've learned something new or become a better engineer than you were yesterday. In either case, ask yourself what about the day made it so.

Crafting a career:

- * Learn to predict industry trends. Keep up with the technology news. Don't let yourself become a dinosaur. Read books, articles, and papers. Learn learn learn.
- * Contribute to open source projects (erm..may not be possible for employees of all tech companies). A good place to start is by improving test suite coverage in a lot of projects.
- * Try to get on teams where there are an abundance of better engineers to learn from. This will always be possible excepting a few of the truly great. Being surrounded by better players will make you better. Don't look for situations where you will be the smartest person there.
- * Make the hang. Don't be afraid to approach technology leaders and try to glean wisdom from them. Most people are happy to help.
- * Follow your curiosity. It may lead to unforeseen opportunities. Having written that out just now, I realize that this is probably a common fortune cookie message.

There's more, but those were the top ones for me. I'd also recommend Bob Martin's The Clean Coder and Edmund Lau's The Effective Engineer, but I think this one's the best so far. May just be the recency effect though, I should go through those again.

caisah says

This was a great read. The author tries to explain what skills and habits a programmer has to develop in order to be successful.

The chapters are short and every one of them ends with a tiny to do paragraph meant to help you develop a new skill or improve an old one. There are some analogies between the processes a programmer should follow and the ones a musician would.(The author himself being a saxophone player).

Almost every chapter is well argued. Several examples and analogies are presented, so by the end of the book you are totally sold.(At least I was...)

This is not a "self help" book and the ideas that were exposed are hard to implement in your day to day life, pragmatically speaking. But it is a great practical guide for a programmer and how one should think and act.

Arthur Santos says

O livro define uma série de valores que um profissional deve adotar para conquistar uma carreira de destaque.

O autor mostra como motivação, foco, dedicação, desejo de melhoria pessoal contínua, curiosidade, empreendedorismo são importantes na construção de uma carreira.

Manter-se fora de sua zona de conforto é o mote. Nesse sentido, o livro é relevante não só para desenvolvedores de software, mas para qualquer profissional que deseje ter uma carreira memorável em sua

área de atuação.

Para seu público alvo (especificamente, desenvolvedores de software), o livro traz um roteiro de ações que podem ser executadas para ajudar a refletir sobre aspectos pessoais e de nossa rotina de trabalho que criam obstáculos para a evolução da carreira. Com base nessas reflexões, é possível adotar medidas que levem às mudanças necessárias para manter-se relevante no mercado de trabalho.

Uma leitura que recomendo a todos, especialmente aos profissionais em início de carreira.

Francis Fish says

I had this book on my phone and read it in small chunks. It's written in a way that lets you easily dip in and out of it.

It's an interesting read and gives you exercises to try at the end of each small vignette. I think that if you try them you will benefit greatly.

I've been having a career for a long time and found it interesting that Chad validates a lot of things I have discovered empirically over the years, particularly about understanding why your employer employs you to have fun writing code. I think the suggestion that you spend some time understanding how companies work, and how money is made is one that a lot of new graduates need to get on with.

I thought some of his asides about the problems with the inability to say "no" when he was setting things up in India that can really trip you up were useful to know. Chad is himself really interesting, you can find an interview with him on the pragmatic programmers iTunes feed that was done just after the book was published that's well worth a listen.

He was/is a sax player and a lot of the analogies he uses are from being a musician and having fun, plus also nice little tips like "always be the worst person in the band" - so you can learn new things.

Milutin says

I am glad that the recent ideas about a career as a programmer are written in one book. I get better insight than from reading bits in blogs and tweets. However the author is too personal, and he is writing too much about himself. So if you are not his friend or have similar experience for example your are not a jazz musician than it's boring and you skip half of the book. And it's short one.

It's enough to read the table of content:

Lead or Bleed?

Supply and Demand

Coding Don't Cut It Anymore

Be the Worst

Invest in Your Intelligence

Don't Listen to Your Parents

Be a Generalist

Be a Specialist
Don't Put All Your Eggs in Someone Else's Basket
Love It or Leave It
Learn to Fish
Learn How Businesses Really Work
Find a Mentor
Be a Mentor
Practice, Practice, Practice
The Way That You Do It
On the Shoulders of Giants
Automate Yourself into a Job
Right Now
Mind Reader
Daily Hit
Remember Who You Work For
Be Where You're At
How Good a Job Can I Do Today?
How Much Are You Worth?
A Pebble in a Bucket of Water
Learn to Love Maintenance
Eight-Hour Burn
Learn How to Fail
Say "No"
Don't Panic
Say It, Do It, Show It
Perceptions, Perschmeptions
Adventure Tour Guide
Me Rite Reel Nice
Being Present
Suit Speak
Change the World
Let Your Voice Be Heard
Build Your Brand
Release Your Code
Remarkability
Making the Hang
Already Obsolete
You've Already Lost Your Job
Path with No Destination
Make Yourself a Map
Watch the Market
That Fat Man in the Mirror
The South Indian Monkey Trap
Avoid Waterfall Career Planning
Better Than Yesterday
Go Independent
Have fun

Robert Boyd says

This is one book I wish I had at the beginning of my career. Or maybe I wish I could force any of my past non-tech managers or clients to read. It does a solid job of describing many of the realities of being a software craftsman.

That said, I have to agree with a lot of the lukewarm reviews that there aren't really any earth-shattering revelations that will change the way you work if you're already established in this profession. Self analysis is always a good thing though, and this book dishes it out in healthy servings. There are also some decent stories and phrases that are worth remembering. The closing section is nicely written (if somewhat abrupt) and will make you feel good about being a programmer.

The book does a fine job of tying together several areas of our profession, but for any particular subject hit there are probably a few books that would be more relevant than this one. All in all, it's a quick and worthwhile read.

Wouter says

As an experienced programmer it's safe to say that Chad fails to see the real important problems to be solved in the software industry, instead of trying to sell me to "research the market and pick a programming language that's hot". This read like a book for applied, applied programmers that haven't yet managed to really understand core principles: abstract structuring. And yet, there's a small chapter called "invest in your intelligence", right before "your biggest career fears"? I might have mistakenly identified the content with morale on being concerned about good programming. Instead I found hip one-liner tips about career (well, that word is actually on the cover, so whoops) advancement that I know from experience are just plain wrong. Luckily there's a foreword by D.H. Hansson!

Harish Sankar says

Personal Review: I liked the book, but did not "really like it"

Originally this book was intended for US citizens to protect their jobs.

"Your Job went to India" was its initial title.

And as an Indian, I cannot accept the motivation for the author to write the book- to save Americans their jobs.

Truth is- it is a level playing field, and there is no job security anymore.

The author realised this, and hence changed the title.

Its a good book- well written with action points to improve your programming career.

But it cannot make an unpassionate programmer (like me) into a passionate one. It has to come from within.

Yevgeniy Brikman says

Pros: it's good to see a book that encourages people to take a more disciplined look at their careers. The book covers many important ideas for being a successful programmer: work with people better than you; don't listen to your parents; be a generalist; find a mentor; learn how to fail; learn to say no; build your brand.

Cons: the voice used in the book feels wrong--at times, it sounds like an infomercial or self help book. There are a number of weird programmer stereotypes and a strange focus on Indian IT (perhaps from v1 of the book?). Finally, the level of discussion is often too simplistic, glossing over all the nuances and gray areas. This book goes broad, but not deep.

Himanshu says

This is by far the most inspirational book I have read so far this year. Chad Fowler is a respected authority in the Ruby programming community and this book is aimed at aspiring software such as myself. The main theme in the book and what I really got out of it in the end is the suggestion of paying attention to not only acquiring cutting edge skills but also indulging in self-marketing and networking in person. He suggests treating your career as a product that you work on improving non-stop and also marketing it. Other important things he suggests is keeping an out for emerging technologies and getting involved at an early stage to end up becoming an expert eventually. Overall, I highly recommend this book aspiring or currently involved in the software development world.
