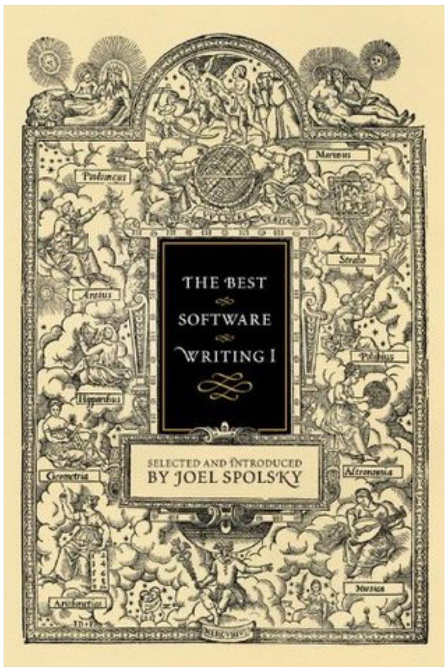

# The Best Software Writing I: Selected and Introduced by Joel Spolsky

*Joel Spolsky (Selections) , Gary Cornell (Editor)*

# The Best Software Writing I: Selected and Introduced by Joel Spolsky

*Joel Spolsky (Selections) , Gary Cornell (Editor)*

**The Best Software Writing I: Selected and Introduced by Joel Spolsky** Joel Spolsky (Selections) , Gary Cornell (Editor)
Frustrated by the lack of well-written essays on software engineering, **Joel Spolsky** (of www.joelonsoftware.com fame) has put together a collection of his favorite writings on the topic.

With a nod to both the serious and funny sides of technical writing, *The Best Software Writing I: Selected and Introduced by Joel Spolsky* is an entertaining read and a guide to the technical writing literati.

*The Best Software Writing I* contains writings from:

Ken Arnold Leon Bambrick Michael Bean Rory Blyth Adam Bosworth danah boyd Raymond Chen Kevin Cheng and Tom Chi Cory Doctorow ea_spouse Bruce Eckel Paul Ford Paul Graham John Gruber Gregor Hohpe Ron Jeffries Eric Johnson Eric Lippert Michael Lopp Larry Osterman Mary Poppendieck Rick Schaut Aaron Swartz Clay Shirky Eric Sink why the lucky stiff

## The Best Software Writing I: Selected and Introduced by Joel Spolsky Details

| | | |
|---|---|---|
| Date | : | Published October 21st 2005 by Apress (first published January 1st 2005) |
| ISBN | : | 9781590595008 |
| Author | : | Joel Spolsky (Selections) , Gary Cornell (Editor) |
| Format | : | Paperback 305 pages |
| Genre | : | Computer Science, Programming, Software, Nonfiction, Computers, Coding |

# From Reader Review The Best Software Writing I: Selected and Introduced by Joel Spolsky for online ebook

## Chris says

I liked this. Published in 2005, some of the essays in this collection are starting to show their twelve-years-plus age, especially with regard to specific technologies and products. And it's an interesting glimpse of a software industry before the eras of smartphones and their apps, Twitter and Facebook, and the resurgence of JavaScript. But overall there's a lot that's still worthwhile to be found here.

To the criticism I've seen in other reviews that this book is somehow worthless because it's "only a collection of blog posts" which can be read on their original sites, I completely disagree. For one, Spolsky has some of his own commentary, introduction, and explanatory footnotes here. Maybe they're not *essential*, but they're nice. Also, especially as the years go on, not all of these sites are going to still be around. But mainly a collection is nice because it's a collection: specific choices, brought together in a specific order, and held together to make a whole that's greater than the sum of its parts.

Read this for the Austin Computer Book Club, for the March, 2017 meetup. Consensus seemed to similar to mine here: this was a good one, and worth a read.

---

## Dhananjay Balan says

A good collection of essays on mordern programming culture and practice.

---

## Christy Ford says

Never has a book made the term 'dead tree' seem as true as this one. With minimal work, almost all of the content in this is freely available online, and I would highly recommend reading the introductions on paper, but viewing the pieces themselves in their original hyperlinked state. Cut off from their comments, with references reduced to footnotes rather than links, the writing shrinks. It feels much more alive in the context it was written.

That said, it is a fair collection of diverse and important technological writing. It feels like a good introduction to the online developer community. Someone who has been keeping up with software blogs for any time will already be familiar with some of the material - but that's how a best of collection should be.

at five years, the collection is already showing it's age. It was collected in a pre-iPhone, pre-facebook age. In some ways, this is a little funny, since you know the answers to many of the big questions being discussed. In other ways, it is a valuable snapshot of 2005 - giving perspective on just how fast things are changing in this world.

---

### Amar Pai says

There's some good stuff in here but it's all stuff you can find online, and there's a lot of filler as well. I like the idea of this book but feel like they weren't able to keep the bar high enough for writing quality. e.g. "ea spouse" -- clearly a significant blog post and one that had a huge impact on the industry-- but is it great writing? Not really.

My favorite essay from the book was Strong Typing vs Strong Testing by Bruce Eckels. Really good discussion of the merits/drawbacks of dynamically typed languages (e.g. Python) relative to statically typed languages (e.g. Java/C++.)

---

### Gavin says

Odd beast: a time capsule where half the items are of purely historical interest, and half are general and extremely wise arguments that are still not acted upon today. He had planned them to be annual collections, but they didn't happen, so this looks to represent more than one year's best.
Recent enough to tell us something about the internet, though with lots of anachronism. But it's more at the lexical level - "weblog", "Sociable media" - than the semantic.

Found (eminent media researcher) danah boyd ludicrous: she calls developers autistic, and calls people with several online identities multiple-personality disordered (a person is one person. So all their activities have to be one person!) Disappointing, typical social theory. She aggressively pushes a horrendous risky single-sign-in for all sites based on these shitty polemics and nothing else.

Contains helpful principles which will not age:, e.g. "if you can't understand the spec for a new technology, don't worry: nobody else will understand it either, and the technology won't be that important".

[Various]

---

### Troy Mcconaghy says

A mixed bag.

---

### Kristjan says

Notice that these are "selected and introduced by Joel Spolsky".

Not all are equaly good in writing and since topics vary then also not equality interesting.

---

### Derek says

Reading well-written material on technology is still something exciting to me. We're at a cusp where the world is becoming more permeated with technology, my generation is growing up with tools that are as old as we are and thus our understanding and acceptance of technology is growing. Yet the stereotypical nerd is still out there, hammering out code while being incredibly inept at writing documentation. So finding collections of material like this, where people have a deep understanding of technology and software code yet also the eloquence to put their thoughts and opinions to paper, is a wonderful thing to me. But perhaps I'm only catching up to where the internet has been for some time. A lot of the writing in this book is taken from websites and blogs where good technology writers pour out well-written essays and clearly-defined opinions. As such, sometimes this book feels like it could have been better. Of course, I did read it a couple of months ago now and thus don't remember the content very clearly, but that should say something about the lasting power of this book. Still, nothing beats some clear technological rhetoric.

---

### J.D. Sandifer says

There's a lot of good stuff in this one with a healthy variety. I think there are a few references that are outdated, but on the whole, most of it is timeless wisdom and tips for software. I also love the breadth covered as it will probably stretch most developers in one way or another.

Definitely worth a look if you're a software developer.

---

### Ed says

Kind of hard to get through this one, probably because it's a compilation of different authors' works and that articles are included not just because they're interesting but also because they are examples of good writing. Some of the articles are interesting and some not so much but that is probably due to my personal bias. I still haven't decided how many of the articles are not read-once. For me the book of collected JoelOnSoftware articles would be more relevant.

I haven't actually read all of the articles, but marking it read as I haven't picked it up in 6 months or so (which is telling in itself...)

---

### Jonathan says

A nice selection of essays about software development. Not every story is great, but there aren't too many stinkers either. And the ones that are good are really good. Here are some of my highlights:

Style is Substance - Makes the argument that style (i.e. where you put your curly braces, how you indent, etc) should be built into the compiler (like python does) The argument is basically this: No formatting style adds any significant productivity at all. Unfortunately supporting whatever style a developer wants to use has decreased productivity over the history of programming significantly. Time is wasted arguing over the styles. Time is wasted as file diffs are messed up due to differing whitespace, etc. If the language designers picked a style and enforced it with the compiler all of these problems would go away, and people could just worry about their business problems. A great read. My synopsis won't do it justice.

EA the human story - A good description of a corporate sanctioned permanent death march. EA sounds evil. I'm glad I don't work there.

Strong Typing vs Strong Testing - Makes the argument that the compiler doesn't help you nearly as much as unit tests...

Great Hackers - Describes things that makes one a great hacker.

Team Compensation - Well written description of how bonuses and merit raises can hurt morale. Very interesting stuff.

Hazards of Hiring - discusses some of the common wisdom about hiring, but moves past the surface level to get into the different ways you can try to hire really good people, such as the difference between developers and programmers, and making sure that you hire people who would improve your team in some significant way.

---

### Frank says

2006?? ?? ??? ?? ???? ? ?????? ? ???. ??? ?? ?????.

---

### David says

I liked *Joel on Software*, which is mostly a compilation of articles Joel Spolsky wrote for his blog. I thought the paper copy offered something beyond what I would get from just browsing through Spolsky's archives on the net. Sadly, I do not really have the same feeling about *Best Software Writing*. Most of these articles first appeared online somewhere, and while many of them are reasonably entertaining -- and the editorial footnotes are sometimes hilarious -- I'm no longer so sure that I would prefer to have the collection compiled as a book rather than as an annotated list of hyperlinks.

---

### Eduardo says

It's always interesting to peek into the past, even if we're talking about a very recent one. This book is a collection of essays by several authors picked by Joel Spolsky to serve as the "best software articles of the year", it was published in 2005. Let that sink in, this is pre-social media. Stack overflow *did not* exist yet. It's a world where people still referred to blogs as weblogs. Quite honestly, it was difficult to hold a laugh.

For the most part, most of the articles didn't age well, but there are some pearls, for example the Rands in repose one and the article by Aaron Swartz, (yes, the internet hacktivist).

---

## Yevgeniy Brikman says

A nice collection of blog posts and essays on software. Even though most of these are available online for free, there is so much crappy writing out there, that it's nice to come across a curated, pre-vetted collection from a trusted source. I also wholeheartedly agree with Spolsky's desire to see more quality writing about software, and applaud him for encouraging this sort of work by publishing a book like this. I'm a fan of anything that tries to make the software industry more accessible and interesting.

As with any curated collection, some of the essays were better than others. In particular, "The Pitfalls of Outsourcing Programmers" (Michael Bean), "ICSOC04 Talk" (Adam Bosworth), "Great Hackers" (Paul Graham), and "A Group is its Own Worst Enemy" (Clay Shirky) stand out above all the others (Shirky's work in particular is a must-read). A few of the other essays feel a bit dated, which is understandable, given the book came out in 2005, and the software industry moves quickly. And a few are just silly, short jokes or comics, which serve as nice breaks between the more serious writing.

Overall, a nice quick read.

As always, I saved some of my favorite quotes from the book:

"I've read entire books about outsourcing, and fundamentally nobody seems to understand that software development is design, not manufacturing. Every single line of code that gets written involves making a decision about the design of the software. And for software companies, and any other company that derives competitive advantage from proprietary software, outsourcing design is, eventually, fatal." -- Michael Bean

"The thing about the too-complicated specs is that nobody wants to look stupid, so they never call the designers on designing something too complicated." -- Adam Bosworth

"It is an ironic truth that those who seek to create systems that most assume the perfectibility of humans end up building the systems that are the most soul destroying and most rigid—systems that rot from within, until like great, creaking, rotten oak trees, they collapse on top of themselves, leaving a sour smell and decay. We saw it happen in 1991 with the astonishing fall of the USSR. Conversely, those systems that best take into account the complex, frail, brilliance of human nature and build in flexibility, checks and balances, and tolerance tend to survive beyond all hopes.

So it goes with software. That software which is flexible, simple, sloppy, tolerant, and altogether forgiving of human foibles and weaknesses turns out to be actually the most steel-cored, able to survive and grow, while that software which is demanding, abstract, rich but systematized turns out to collapse in on itself in a slow and grim implosion." -- Adam Bosworth

"An unacknowledged war goes on every day in the world of programming. It is a war between the humans and the computer scientists. It is a war between those who want simple, sloppy, flexible, human ways to write code and those who want clean, crisp, clear, correct ways to write code. It is the war between PHP and C++/Java." -- Adam Bosworth

"Along with interesting problems, what good hackers like is other good hackers. Great hackers tend to clump together—sometimes spectacularly so, as at Xerox Parc. So you won't attract good hackers in linear proportion to how good an environment you create for them. The tendency to clump means it's more like the square of the environment. So it's winner take all. At any given time, there are only about ten or twenty places where hackers most want to work, and if you aren't one of them, you won't just have fewer great hackers, you'll have zero." -- Paul Graham

"Because you can't tell a great hacker except by working with him, hackers themselves can't tell how good they are. This is true to a degree in most fields. I've found that people who are great at something are not so much convinced of their own greatness as mystified at why everyone else seems so incompetent." -- Paul Graham

"So if you ask a great hacker how good he is, he's almost certain to reply, I don't know. He's not just being modest. He really doesn't know. And none of us know, except about people we've actually worked with. Which puts us in a weird situation: we don't know who our heroes should be. The hackers who become famous tend to become famous by random accidents of PR." -- Paul Graham

"If there is a Michael Jordan of hacking, no one knows, including him." -- Paul Graham

"Software for groups is different. Prior to the Internet, the last technology that had any real effect on the way people sat down and talked together was the table." -- Clay Shirky

"Nothing causes a group to galvanize like an external enemy. So even if someone isn't really your enemy, identifying them as an enemy can cause a pleasant sense of group cohesion. And groups often gravitate toward members who are the most paranoid and make them leaders, because those are the people who are best at identifying external enemies." -- Clay Shirky

"Groups often have some small set of core tenets, beliefs, or interests that are beyond criticism, because they are the things that hold the group together. Even in groups founded for fairly intellectual discussion, the emotional piece comes out whenever you threaten one of these core beliefs, because when you take on those beliefs, you are not just offering an opinion, you are threatening group cohesion." -- Clay Shirky

"People who work on social software are closer in spirit to economists and political scientists than they are to people making compilers. They both look like programming, but when you're dealing with groups of people as one of your runtime phenomena, you have an incredibly different practice." -- Clay Shirky

"Reputation is also not generalizable or portable. There are people who will cheat on their spouse but not at cards, and vice versa, and both, and neither. Reputation in one situation is not necessarily directly portable to another." -- Clay Shirky

"Ease of use is the wrong goal for social software. Ease of use is the wrong way to look at the situation, because you've got the Necker cube flipped in the wrong direction, toward the individual, when in fact, the user of a piece of social software is the group." -- Clay Shirky

"You have to find some way to protect your own users from scale. This doesn't mean the scale of the whole system can't grow. But you can't try to make the system large by taking individual conversations and blowing them up like a balloon; human interaction, many-to-many interaction, doesn't blow up like a balloon. It either dissipates, or turns into broadcast, or collapses." -- Clay Shirky

---