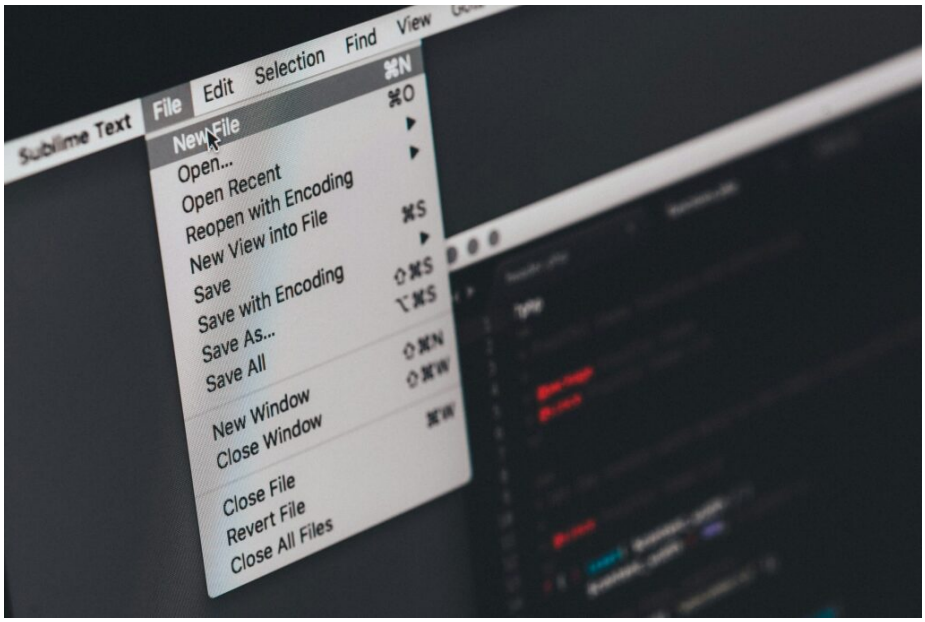
Issue 067: Text Editors

Adrian Kosmaczewski

April 1st, 2024



Welcome to the sixty-seventh issue of *De Programmatica Ipsum*, about *Text Editors.*

In this edition:

- We argue that "editor wars" are a pointless loss of time[1].
- In the Library section[2], we review "Code" by Charles Petzold[3].
- In our Vidéothèque section[4], we watch a video from the Fireship channel[5].

We would like to thank our patrons who generously contribute every month (or have contributed in the past) to our work and help us run this magazine. Thank you so much! In alphabetical order: Adam Guest, Adrian Tineo Cabello, Benjamin Sheldon, Christopher Nascone, Colin Powell, Franz Lucien Moersdorf, Guillermo Ramos Álvarez, Jean-Paul de Vooght, Dr. Juande Santander-Vela, Patryk Matuszewski, Paul Hudson, Quico Moya, Roger Turner, and Szymon Licau.

Enjoy this issue! Please subscribe to our free newsletter[6] to stay updated about new releases, share the articles on social media, or contribute[7] if you would like to support our work with a donation via Liberapay[8].

[1] https://deprogrammaticaipsum.com/for-lack-of-a-better-word/
[2] https://deprogrammaticaipsum.com/category/library/
[3] https://deprogrammaticaipsum.com/charles-petzold/
[4] https://deprogrammaticaipsum.com/category/videotheque/
[5] https://deprogrammaticaipsum.com/fireship/
[6] https://deprogrammaticaipsum.com/newsletter/
[7] https://deprogrammaticaipsum.com/contribute/
[8] https://liberapay.com/

Cover photo by Ilya Pavlov[9] on Unsplash[10].

[9]https://unsplash.com/@ilyapavlov?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

[10]https://unsplash.com/photos/a-close-up-of-a-computer-screen-with-a-menu-hXrPSgGFpqQ?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

For Lack Of A Better Word

Adrian Kosmaczewski

April 1st, 2024

Text is a surprisingly dense medium. Despite what the common pretension of online folklore states, I consider the adage, "an image is worth a thousand words" to be a blatant slur, and this magazine is my feeble attempt at demonstrating such a thesis. Text is powerful, deep, and intricate; if anything because it can be misinterpreted and reinterpreted *a piacere*.

Text is among the most powerful inventions of mankind. Religions have grown upon sacred texts, and those same religions have been broken apart because of blasphemous texts. Writers have been threatened, banned, incarcerated, even immolated because of their texts. Whole libraries have been burned by ignorant mobs at various points in human history. Nations have been created with the sole purpose of granting freedom of creation and distribution of written texts banned somewhere else. Popular works of literature have brought immense prestige and celebrity to their authors, some of which are regularly given various international prizes. Blockbuster movies are based on novels, short stories, and scripts, and the same movie industry is brought to its knees every time writers strike. Speaking about blockbusters, the Joker once joyfully claimed[1] that "the pen is truly mightier than the sword".

Text, text, text.

The aforementioned sacred scriptures give text a predominant role. Taken literally and stripping all context, the first verse in the opening chapter of the Gospel of John[2] ("In the beginning was the Word, and the Word was with God, and the Word was God") literally sounds like advertising copy from Microsoft[3] in the 1990s.

[1] https://www.youtube.com/watch?v=IEuB3bAgdjY
[2] https://en.wikipedia.org/wiki/John_1%3A1
[3] https://deprogrammaticaipsum.com/where-does-microsoft-want-to-go-today/

From a historical perspective, text has always been the primary interaction medium used to interact with computers. Before GUIs, we used command lines and terminals, "piping" commands from `stdout` to `stdin`. Before the World Wide Web, we interacted via email or through BBS, Usenet, and Gopher, all using pure ASCII text. Long before Skype or Zoom, there were text-based tools such as talk[4], ICQ[5] and MSN Messenger[6].

And lo and behold, our first, Turing-test winning interaction with Generative AI was, without surprise, a text-based chatbot. The biblical analogy, in this case, is particularly striking: the first chapter of the Book of Genesis[7] sounds exactly as if God were talking to DALL-E or Midjourney: "let there be light", "let the dry land appear", you get the gist.

For many authors, the written word can be taken to an extreme of alchemy and esoteric knowledge. Suzanne Jill Levine[8], translator and publisher of works by Jorge Luis Borges[9] in English, reminds[10] us of "The Aleph"[11], in which

> the first letter of the Hebrew alphabet becomes the point in time and space that contains all time and everything in the universe.

Was Borges a software developer? Diving deeper in the internal workings of our beloved computing machines, we find that `TEXT` is the familiar name of the "code segment"[12], the subdivision of virtual memory assigned to a single process running in your computer holding, precisely, the code of the program to be executed–also ready to be overwritten by hazardous stack overflow bugs, unless your code is written with Rust[13], of course.

Text is also the code driving the mind behind the keyboard, and the keyboard, in turn, drives the mind that writes the text. Various authors (Friedrich Kittler[14], Arthur Krystal[15], Philip LeVine, and Ron Scollon[16], for example) claim Friedrich Nietzsche[17] said that "our writing tools are also working on our thoughts". Unfortunately, I have not been able to find the precise source (and the original German formulation) of this phrase, but it apparently had to do with Nietzsche being among the first owners of a typewriter[18], and probably the first philosopher to do so, at the end of the nineteenth century.

Ask any diehard Vim user who installs the "vim plugin" or "vim mode" for whatever other editor they have to use, and we can start to see a hint of truth in Nietzsche's assertion. Our tools shape our writing, whether we are conscious of this fact or not.

But let us go back to typewriters for a minute. Throughout the twentieth century, writing machines evolved from crude early experiments to the sophistication of the IBM MT/ST[19] in 1967. Word processor software subsequently relegated them to the status of eBay-worthy collector items during the 1980s and 1990s. This final step was better described in detail

[4] https://en.wikipedia.org/wiki/Talk_(software)
[5] https://en.wikipedia.org/wiki/ICQ
[6] https://en.wikipedia.org/wiki/MSN_Messenger
[7] https://www.biblegateway.com/passage/?search=Genesis+1&version=ESV
[8] https://en.wikipedia.org/wiki/Suzanne_Jill_Levine
[9] https://en.wikipedia.org/wiki/Jorge_Luis_Borges
[10] https://www.bbc.com/culture/article/20140902-the-20th-centurys-best-writer
[11] https://en.wikipedia.org/wiki/The_Aleph_%28short_story%29
[12] https://en.wikipedia.org/wiki/Code_segment
[13] https://deprogrammaticaipsum.com/the-state-of-rust-in-2022/
[14] https://en.wikipedia.org/wiki/Friedrich_Kittler
[15] https://en.wikipedia.org/wiki/Arthur_Krystal
[16] https://repository.library.georgetown.edu/bitstream/handle/10822/558208/GURT_2002_text.pdf?seq
[17] https://en.wikipedia.org/wiki/Friedrich_Nietzsche
[18] http://www.malling-hansen.org/friedrich-nietzsche-and-his-typewriter-a-malling-hansen-writing-ball.html
[19] https://deprogrammaticaipsum.com/jim-henson/

by some important authors, albeit rather unknown in tech circles: Ray Hammond[20], in his 1984 book "The Writer and The Word Processor"[21]; then Zachary Leader[22], in his 1991 book "Writer's Block"[23]; and Matthew Kirschenbaum[24], in his 2016 book "Track Changes: A Literary History of Word Processing"[25]. Oh, and "Word Processor of the Gods"[26] by Stephen King, who most probably wrote it on his legendary Wang computer[27].

We still have plenty of typewriter lovers around. The most well known is Tom Hanks, who not only wrote a book about them[28], but released a famous iOS application[29] to go with it.

(For the record, spoiler alert: I agree with him[30].)

Writers are a finicky bunch. The variety of writing software available today boggles the mind, yet we learn that J.K. Rowling and Stephen King use Microsoft Word[31]. We chuckle upon learning that George R.R. Martin, the author of Game of Thrones, is still using WordStar for MS-DOS[32], or that Christopher Nolan considers himself the "ultimate Luddite"[33] when it comes to screenwriting software:

> Christopher: Yes, it's time looping back on itself. I remember very clearly. I've always used ScriptThing, which then became Movie Magic. I remember when it went to Windows, and it slowed down tremendously. I was like, "Can you run it on DOS?" You could run it on DOS. It has a DOS emulator within Windows.
>
> John: Wild.
>
> Christopher: I am the ultimate Luddite. I still go back to my Royal manual typewriter and do the odd scene on that just to reconnect with it.

(Ironically enough, a scriptwriting AI tool[34] bears his name.)

Applications such as Scrivener[35], Dabble[36], LivingWriter[37], Jutoh[38], and Ulysses[39] are the equivalent of IDEs for novelists and screenwriters.

But I am digressing. Enough with philosophers, well-known writers, and movie directors. Let us focus on the preferred kind of creator for readers of this magazine, the software developer, and their continuous infatuation with text editors.

---

[20] https://en.wikipedia.org/wiki/Ray_Hammond
[21] https://archive.org/details/writerwordproces0000hamm/mode/2up
[22] https://en.wikipedia.org/wiki/Zachary_Leader
[23] https://archive.org/details/isbn_9780801840326/mode/2up
[24] https://english.umd.edu/directory/matthew-kirschenbaum
[25] https://mitpressbookstore.mit.edu/book/9780674417076
[26] https://en.wikipedia.org/wiki/Word_Processor_of_the_Gods
[27] https://archive.org/details/2011-12-stephen-kings-wang/01-introduction-ben-vershbow.wav
[28] https://www.npr.org/2017/10/16/557636219/tom-hanks-is-obsessed-with-typewriters-so-he-wrote-a-book-about-them
[29] https://apps.apple.com/us/app/hanx-writer/id868326899
[30] https://akos.ma/books/Tales_Of_Editors_And_Keyboards/Tales_Of_Editors_And_Keyboards.html
[31] https://www.bookrunch.com/news/which_writing_software_do_famouse_authors_use/
[32] https://www.theverge.com/2014/5/14/5716232/george-r-r-martin-uses-dos-wordstar-to-write
[33] https://johnaugust.com/2024/scriptnotes-episode-622-the-one-with-christopher-nolan-transcript
[34] https://www.nolanai.app/
[35] https://en.wikipedia.org/wiki/Scrivener_(software)
[36] https://www.bookrunch.com/alternatives/Dabble/
[37] https://livingwriter.com/
[38] https://www.jutoh.com/
[39] https://ulysses.app/

Let us be honest: our field is all about infatuation. Tabs versus spaces. Modal versus non-modal. Plain-text editors versus IDEs. JetBrains Mono versus IBM Plex. Vim versus Emacs. EditPlus versus UltraEdit. BBEdit versus TextMate. AI-enabled versus none. And now we are perceiving yet another battle in this never-ending holy war: Zed versus Visual Studio Code.

What is the bare minimum set of features that one expects in a coding text editor? Syntax highlighting, Git[40] support, multiple undo and redo, multiple encodings with UTF-8 as default, (very) large file handling (log files, anyone?), line numbers, dark mode, configurable fonts... and inserting spaces instead of tabs.

(Hey, my online magazine, my rules.)

Oh, and plugins and extensions. Lots of them. Gotta love ecosystems.

Thankfully, we are in 2024, and many code editors provide such feature set. On Windows, we have EditPlus[41], UltraEdit[42], Notepad++[43] (based on Scintilla[44]), or KEDIT[45]. On Unix, we have the saint trinity of Vim[46], Emacs[47], and pico[48] / nano[49]. Oh, and JOE[50]. On the Mac, Espresso[51], Nova[52], BBEdit[53], Smultron[54], or TextMate[55] (made famous by the now historic 2005 demo of Ruby on Rails[56] by David Heinemeier Hansson). And across platforms, the venerable jEdit[57], the obscure Thonny[58], or the upcoming JetBrains Fleet[59].

In terms of market share and popularity, the big winners at the time of this publication are Visual Studio Code[60] and Sublime Text[61], followed by a few newcomers aiming for the top spot in the upcoming years: Neovim[62], Zed[63], and Brackets[64].

There is no shortage of IDEs for those expecting more features... or putting up with their

[40] https://deprogrammaticaipsum.com/twenty-years-is-nothing/
[41] https://editplus.com/
[42] https://www.ultraedit.com/
[43] https://notepad-plus-plus.org/
[44] https://www.scintilla.org/
[45] https://www.kedit.com/
[46] https://www.vim.org/
[47] https://en.wikipedia.org/wiki/Zachary_Leader
[48] https://en.wikipedia.org/wiki/Pico_(text_editor)
[49] https://www.nano-editor.org/
[50] https://joe-editor.sourceforge.io/
[51] https://espressoapp.com/
[52] https://nova.app/
[53] https://www.barebones.com/products/bbedit/
[54] https://www.peterborgapps.com/smultron/
[55] https://macromates.com/
[56] https://www.youtube.com/watch?v=Gzj723LkRJY
[57] http://www.jedit.org/index.php
[58] https://thonny.org/
[59] https://www.jetbrains.com/fleet/
[60] https://code.visualstudio.com/
[61] https://www.sublimetext.com/
[62] https://neovim.io/
[63] https://zed.dev/
[64] https://brackets.io/

"magic"[65]: the JetBrains[66] suite, Xcode[67], Visual Studio[68], Eclipse[69], NetBeans[70], Qt Creator[71], Lazarus[72], KDevelop[73], or GNOME Builder[74].

If you are lazy and do not want to install anything else on your system, every desktop environment comes bundled with a default editor that is usually just good enough to get started: vi[75], Notepad[76], TextEdit[77], gedit[78], Kate[79], Leafpad[80], FreeDOS Edit[81], and so on.

The market for Markdown editors is exploding: Bear[82], Byword[83], Obsidian[84], MarkdownPad[85], Apostrophe[86], Typora[87] (the choice of this author), iA Writer[88], Joplin[89], and so many more[90].

If you are in the market for LaTeX-capable text editors, you might want to consider TeXShop[91], Texmaker[92], Overleaf[93], Texifier[94], Compositor[95], Kile[96], TeXstudio[97], or LyX[98].

History boffins will probably be trying WordStar[99], TECO[100], MacWrite[101], Bravo[102], Lo-

[65] https://stackoverflow.blog/2020/11/09/modern-ide-vs-vim-emacs/
[66] https://www.jetbrains.com/
[67] https://developer.apple.com/xcode/
[68] https://visualstudio.microsoft.com/
[69] https://www.eclipse.org/
[70] https://netbeans.apache.org/
[71] https://www.qt.io/product/development-tools
[72] https://www.lazarus-ide.org/
[73] https://kdevelop.org/
[74] https://wiki.gnome.org/Apps/Builder
[75] https://en.wikipedia.org/wiki/Vi_(text_editor)
[76] https://en.wikipedia.org/wiki/Windows_Notepad
[77] https://en.wikipedia.org/wiki/TextEdit
[78] https://en.wikipedia.org/wiki/Gedit
[79] https://kate-editor.org/
[80] https://en.wikipedia.org/wiki/Leafpad
[81] https://opensource.com/article/21/6/freedos-text-editor
[82] https://bear.app/
[83] https://www.bywordapp.com/
[84] https://obsidian.md/
[85] http://www.markdownpad.com/
[86] https://apps.gnome.org/Apostrophe/
[87] https://typora.io/
[88] https://ia.net/writer
[89] https://joplinapp.org/
[90] https://www.markdownguide.org/tools/
[91] https://pages.uoregon.edu/koch/texshop/texshop.html
[92] https://www.xm1math.net/texmaker/
[93] https://www.overleaf.com/
[94] https://www.texifier.com/
[95] https://compositorapp.com/
[96] https://kile.sourceforge.io/
[97] https://texstudio.org/
[98] https://www.lyx.org/
[99] https://arstechnica.com/information-technology/2017/03/wordstar-a-writers-word-processor/
[100] https://en.wikipedia.org/wiki/TECO_(text_editor)
[101] https://en.wikipedia.org/wiki/MacWrite
[102] https://en.wikipedia.org/wiki/Bravo_(editor)

coScript[103], jim[104], ed[105], edlin[106], edit.com[107], and the recently discontinued Atom[108] on their retrocomputing[109] systems.

If you are the experimental kind of person, you will be interested in trying out Wily[110], Sam[111], or Acme[112]; the latter two created by Rob Pike, of Go programming language fame. Also noteworthy, Acme's UI was inspired by Niklaus Wirth[113]'s Oberon system.

Finally, the adventurous among you might consider the possibility of writing yet another text editor. What could possibly go wrong? In that case, take the time to learn how they work internally[114], including concepts such as memory mapping[115], gap buffers[116], piece tables[117] or ropes[118]. You can also peruse the source code of Microsoft Word for Windows 1.1a[119] for ideas.

(But please, please, please, do not take inspiration from the abomination that is WordPress' own Gutenberg[120] editor. Please, just do not.)

There are many more[121] text editors that could fit in this article. This is the reason why I pray my readers to indulge this author, should their preferred one not appear in any of the lists above.

The immense variety of writing tools is a direct reflection of our own neurodiversity. Without falling into clichés, and despite what Apple might say about this, we all think differently, all the time. We all need a different tool to express our thoughts, and our modern software market has no shortage of them. Most importantly: we all have something to say, and we all have a different way to say it.

Even more important still: the whole idea of an "editor war"[122] is a pointless exercise, worthy only of the lesser mind. Those who insist on indulging in such stupid matters might want to review their priorities in life. Let people use whatever editor they fancy and need at every single moment of their careers, and teach younger generations to respect those choices; just avoid entering arguments over the subject. There are plenty of more urgent matters to take care of, in and out of our fairly limited (and decaying) programming world.

But beware of the fact that your writing tool of choice is also shaping, if not your content, at least your form. Cross-training, in the form of switching editors every so often or for different tasks, might be a healthy option for your brain.

[103] https://en.wikipedia.org/wiki/LocoScript
[104] https://man.cat-v.org/unix_8th/9/jim
[105] https://en.wikipedia.org/wiki/Ed_(text_editor)
[106] https://en.wikipedia.org/wiki/Edlin
[107] https://en.wikipedia.org/wiki/MS-DOS_Editor
[108] https://github.blog/2022-06-08-sunsetting-atom/
[109] https://deprogrammaticaipsum.com/return-to-innocence/
[110] http://www.cs.yorku.ca/~oz/wily/
[111] http://sam.cat-v.org/
[112] http://acme.cat-v.org/
[113] https://deprogrammaticaipsum.com/niklaus-wirth/
[114] https://ecc-comp.blogspot.com/2015/05/a-brief-glance-at-how-5-text-editors.html
[115] https://en.wikipedia.org/wiki/Memory-mapped_file
[116] https://en.wikipedia.org/wiki/Gap_buffer
[117] https://en.wikipedia.org/wiki/Piece_table
[118] https://en.wikipedia.org/wiki/Rope_(data_structure)
[119] https://computerhistory.org/blog/microsoft-word-for-windows-1-1a-source-code/
[120] https://wordpress.org/gutenberg/
[121] https://en.wikipedia.org/wiki/List_of_text_editors
[122] https://en.wikipedia.org/wiki/Editor_war

And, if all else fails, paper notebooks and pens are always available at a nearby store. You should try them, they are wonderful.

Cover photo by Amador Loureiro[123] on Unsplash[124].

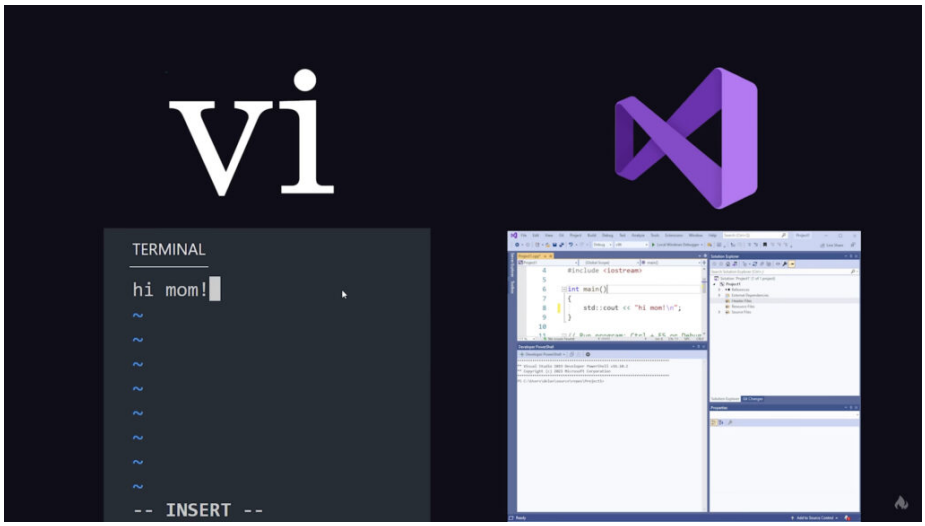
[123]https://unsplash.com/@amadorloureiro?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

[124]https://unsplash.com/photos/letter-wood-stamp-lot-BVyNlchWqzs?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

Fireship

Adrian Kosmaczewski

April 1st, 2024



In our 50th issue[1], we reviewed some of the greatest classics in the field of programming and computing humor. Before that, we had reviewed the work of Kathy Sierra[2], a pioneer in the art of making computer programming books accessible and fun. Today, we will review a YouTube channel that combines the best of both.

The Fireship YouTube channel[3] published its first video[4] in April 2017. Comparing it to more recent ones from the same channel gives a few hints of what stayed and what changed since those beginnings. The author has found a voice (in both a figurative and literal sense) and a winning format. Considering the constant progression of the number of views, their current formula is decidedly a very successful one: from low tens of thousands of views to almost three million (and counting) for the video we will be talking about today.

Precisely, this week's Vidéothèque movie is "I tried 10 code editors"[5]. Fitting the subject of this month's edition, this video also fits the style of their more recent productions: fast-paced, featuring relevant content, and with plenty of memes, all set on a black background. The final result is interesting, precise, and most importantly, very, very funny.

Many of their recent videos feature a recurrent "newsreel" theme, including a title card labeled "The Code Report" and the date of publication. These videos provide a quick update or discuss some major news at any particular moment.

(Computer historians: bookmark this channel, you are going to refer to these videos in the future for sure.)

[1] https://deprogrammaticaipsum.com/issue-50-humor/
[2] https://deprogrammaticaipsum.com/kathy-sierra/
[3] https://www.youtube.com/@Fireship
[4] https://www.youtube.com/watch?v=iaXhn5Wuk2c
[5] https://www.youtube.com/watch?v=8PhdfcX9tG0

This month's Vidéothèque video starts with the quintessential founding myth of our craft: Grace Hopper finding the legendary moth ("bug") stuck in her computer. According to the narrator, and quite truthfully so, developers these days can create their own bugs thanks to, you guessed it, code editors.

(We should probably rename the whole product category to "bug editors", frankly.)

The meaty part of the movie comes next. It describes the major characteristics of ten different code editors and IDEs, mostly related to the field of web (or, as the kids call it these days, "full-stack") as well as mobile app development. In order:

1. vi[6] (frankly, an excellent introduction, if you ask me)
2. Emacs[7] (with special mention of the resulting editor wars)
3. Vim[8] (and how it "improved" vi)
4. Neovim[9] (and how it "improves" Vim, in particular thanks to its choice of Lua[10] over Vimscript[11])
5. nano[12] (part of the "g-nu" project, listen carefully to the pronunciation of the name)
6. Notepad[13] (seriously?) and NotePad++[14] (which, according to the author, "feels like using Microsoft Excel to write code")
7. Adobe Dreamweaver[15] (my HTML editor of choice in 1998, when it was still called Macromedia Dreamweaver)
8. Visual Studio Code[16] (arguably one of the most popular code editors nowadays, even if somewhat bloated thanks to its Electron[17] underpinnings)
9. Visual Studio[18] (with an interesting observation about how complex IDEs are great if you are committed to a particular platform, .NET in this case)
10. and JetBrains WebStorm[19] (with the required disclaimer that this is not a sponsored video)

"I tried 10 code editors"[20] is available on the highly recommended Fireship YouTube channel. The author has since 2019 expanded their work with a dedicated learning website[21] featuring courses and projects to get into programming in a fun way, definitely worth a try.

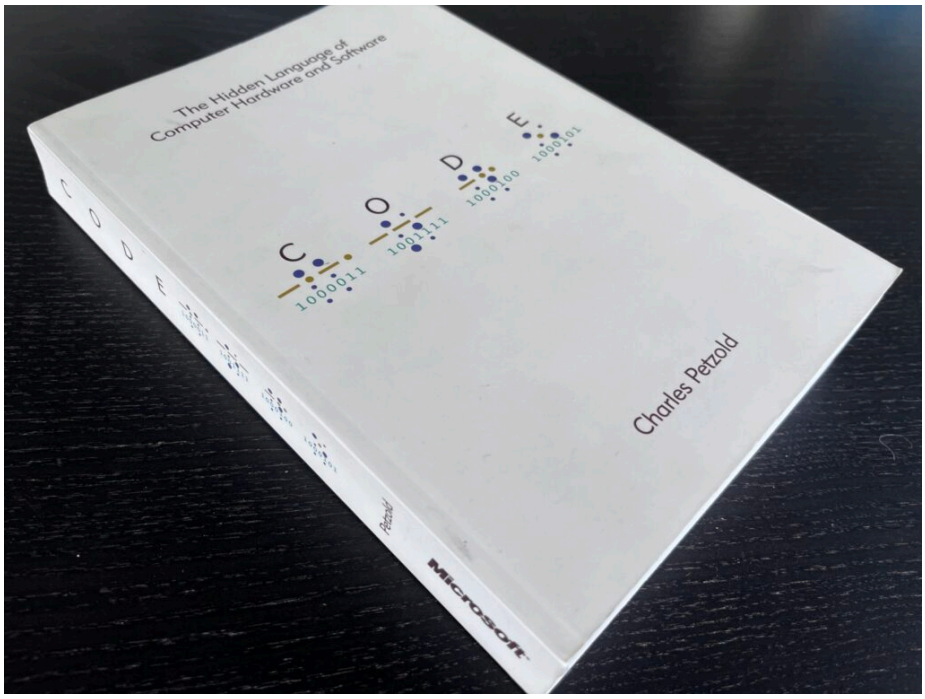
Cover snapshot chosen by the author from the video.

[6] https://en.wikipedia.org/wiki/Vi_(text_editor)
[7] https://www.gnu.org/software/emacs/
[8] https://www.vim.org/
[9] https://neovim.io/
[10] https://www.lua.org/
[11] https://learnvimscriptthehardway.stevelosh.com/
[12] https://www.nano-editor.org/
[13] https://en.wikipedia.org/wiki/Windows_Notepad
[14] https://notepad-plus-plus.org/
[15] https://www.adobe.com/products/dreamweaver.html
[16] https://code.visualstudio.com/
[17] https://www.electronjs.org/
[18] https://visualstudio.microsoft.com/
[19] https://www.jetbrains.com/webstorm/
[20] https://www.youtube.com/watch?v=8PhdfcX9tG0
[21] https://fireship.io/

Charles Petzold

Adrian Kosmaczewski

April 1st, 2024



How do you start learning about computers? The opinions about this particular subject have a cardinality close to the number of computer scientists or IT professionals on the planet. Everyone will have their own opinion, but a single book published in 2000 might have helped everyone reach an agreement, and that is no small feat.

Charles Petzold[1] was the first and foremost evangelizer[2] of Windows technology at the end of the 1980s, on the other side of the street of (and with a slightly more technical tone than) Guy Kawasaki[3]. The fact that he has (or at least had) a tattooed Windows Logo on his right arm[4] should tell you something.

In his own words[5]:

> I acquired my first IBM PC in early 1984 and began writing for *PC Magazine* that same year. This led to a full-time freelance career that included writing for *Microsoft Systems Journal* and *MSDN* magazines.

His 2000 book "Code: The Hidden Language of Computer Hardware and Software" published by Microsoft Press[6] (a second edition of which was released in 2022[7]) deserves its own

[1] https://en.wikipedia.org/wiki/Charles_Petzold
[2] https://deprogrammaticaipsum.com/less-evangelization-more-honestization/
[3] https://deprogrammaticaipsum.com/guy-kawasaki/
[4] https://blog.codinghorror.com/the-cognitive-style-of-visual-studio/
[5] http://www.charlespetzold.com/about/
[6] https://deprogrammaticaipsum.com/microsofts-writings-on-security/
[7] http://www.charlespetzold.com/books/

Wikipedia page[8] and many reads and re-reads. It has been widely praised as a classic in the field, and it is about time that we dedicate a few words to this masterpiece in this magazine, too.

However, we will not dive in the usual review of its contents or style because a lot has been written about it, including a quote by Jeff Atwood[9] I wholeheartedly agree with:

> I also own Charles Petzold's book, Code. It's another love letter to the computer.

(And no, I do not always agree with what Jeff says, but when I do, it is with all my heart.)

I bought my copy at the bookstore of the Microsoft TechEd Europe (the predecessor of the Microsoft Ignite[10] series of events) in Barcelona, in July 2003, among other books that I still have on my shelves.

The cover says it all. The word "Code", spelled in Braille, Morse, ASCII, and Latin script. An invitation to many different places, all at once, in a single 400-page volume.

I remember starting reading it as soon as I got back to my hotel that night, unable to keep my eyes off it for the duration of the event (outside the conference sessions, of course!) As a self-taught software developer, one who would complete a university degree only five years later, the discovery of this book opened my eyes to many aspects of computing beyond programming that I was not aware of.

(Maybe the fact that I clicked with Petzold's writing style so much is that, according to his own biography[11], he also had a "a self-taught education in digital electronics")

In the humble opinion of this author, "Code" remains the authoritative book to recommend to anyone asking themselves the question: "how do computers work?" To this day, there is no other written work that provides a better answer to that question than this one.

Is it complete? No, of course not. Is it relevant? Absolutely. Is it illustrated? Please. Is it fun? Hell yes.

In particular, and dear to my own passion for computer history, Mr. Petzold provides all the references (back to the 19th century) to explain why and how the computer came to be the way it is. These references are not only technical (including snippets of code in Assembly and Algol whenever necessary) but also scientific (with the required introduction to Boolean logic, circuitry, etc.)

All in all, "Code" is a timeless gem that will remain one of the greatest contributions of Charles Petzold to our field. Because if it were the only one, it would be enough already, but his 2008 "Annotated Turing"[12] book deserves a separate entry. And this is without mentioning the long list of books[13] about Windows, OS/2, C#, and even Windows Phone (!) that he published during his 34-year illustrious career as a technical writer.

If you are a young person interested in the (slowly waning) craft of software engineering, start with this book, and then read Steve McConnell's "Code Complete"[14]. Both titles published

[8] https://en.wikipedia.org/wiki/Code:_The_Hidden_Language_of_Computer_Hardware_and_Software
[9] https://blog.codinghorror.com/if-loving-computers-is-wrong-i-dont-want-to-be-right/
[10] https://en.wikipedia.org/wiki/Microsoft_Ignite
[11] http://www.charlespetzold.com/about/
[12] http://www.theannotatedturing.com/
[13] http://www.charlespetzold.com/books.html
[14] https://deprogrammaticaipsum.com/steve-mcconnell/

by Microsoft Press, both bearing similar names, and both equally useful and enjoyable, even decades after their publication. Because our craft is reaching a point of stability in its evolution, and there are quite a few timeless truths worthy of attention, despite what the hype[15] mongers would like you to believe.

Cover photo by the author.

[15] https://deprogrammaticaipsum.com/issue-1-hype/