WeGet letters

by Michael W Lucas

> Greetings and felicitations,
> oh mighty Letters Column Master!
>
> Now that I've gotten the obligatory "sucking up so you'll pay attention" out of the way, I'll ask: are you crazy? You're supposed to be answering people's sincere and heartfelt letters, and instead you tell them that they're doomed for even asking. This is an issue dedicated to development, and I bet anything you're going to spend your pages slagging on developers.
>
> How dare you, sir? How *dare* you?
>
> —Not A Fan

Dearest NAF,

I have absolutely nothing against developers. Most—er, many of them are lovely human beings. I simply wish that they had dedicated their lives to something that might improve civilization, like volunteering to pick up trash by the roadside.

My problem is with code, not coders.

We treat computer code like a precious treasure worthy of hoarding, when in reality it's like nuclear waste with a few rubies scattered in it. While every line that emerged from the CSRG is unalloyed platinum, most code repositories contain a whole bunch of barely functional spew supporting occasional scintillating scraps of brilliance. Some of those luminous lines are shackled into supporting the great threats dooming our civilization, like Facebook.

Yes, the world—really the Internet, but if you're a developer isn't the Internet your entire world?—is bloated with documentation on how to write better code, but none of it agrees with one other and most of you can't be bothered to read the instructions anyway. No, don't argue. I write that documentation; I have nearly unholy knowledge of how many of you read the stuff.

If you want to be a developer and yet improve civilization, use your hard-won acumen towards *reducing* the amount of code the world uses.

Every line of code is a seed of technical debt waiting for an opportunity to sprout into a malignant blossom, and every program is a farm of their horrific sprouts. Every package you install begins suffering from neglect the instant you log out, which is why some of you have terminal sessions that have been open for six years and think it's okay because the server is behind the firewall, and we're all doomed anyway. Computer people always think that there's a technical solution when the only solution is to shut off the laptop and hang out in meatspace for a lifetime or two.

Very few developers spend their careers writing clean, new, perfect implementations. The university churns out these bright-eyed maniacs who think that they'll be writing IP routers in Java just like their senior project demanded, then they get a job where they're tipped face-first into the nuclear waste vat and told to make it not radioactive. They spend aeons fixing bugs caused by other people's insufficient grasp of how their code works, until they achieve enough seniority that they're allowed to write their own bugs.

It's enough to make someone clean-field write a nearly useless program in the hope of demonstrating what good software should look like and post it on Github, just to prove that they exist. Or that they used to exist. Did you know Github has a feature to set the heir to your code? That horrible program you wrote for your own satisfaction, but other people discovered and filed bugs against until it took over your life and finally made you stroke out? Before you retire and start choosing which brand of dollar store cat food you'll be dining on for your twilight years, be sure to choose your code's next victim. If you pick me, I'll immediately auction off all rights to the least savory bidder and exploit the proceeds to soil your legacy.

The most heroic developers are those who delete code.

So much code hasn't been touched in decades because it seems to work, when the reality is it's failed in ways nobody has noticed yet. Study it. Should it be ripped out because it's old? Certainly not! It should be ripped out only if and when there's a more maintained method of doing the same thing.

Probably a library. One of my least loathed "innovations" of the last couple decades is FreeBSD's libarchive.

Unix has too many formats for compressing and collating data because most of them were invented on and for Unix. Does anyone with less than a decade of experience understand when to use compress(1) versus Microsoft CAB archives? No, because nobody with any amount of experience remembers that except for a few hard-core archive format specialists. What about the hydra-headed tar format? Eliminate one tar format, and two more grow to take its place. Worse, each of those new tar formats are optimized for increasingly particular use cases.

Every archiving program supported its own format. Many of them had marginal support for other formats. When I started as a sysadmin I could use tar(1) to unzip archives, except when the zip format was really compressed and some Idiot (me, I'm Idiot) slapped the wrong extension on the filename.

Libarchive provided a single central source of compression and archiving truth. Programs that relied on libarchive could work with any file format. Bugs discovered and fixed in libarchive instantly propagated to every program that linked it.

The real benefit of libarchive was that it reduced the amount of code in use.

Instead of dozens of programs sketchily implementing their own so-called support of whatever formats they preferred, these programs discarded their own engines and pulled in libarchive. This library might have tens of thousands of lines of code, but using it removed hundreds of thousands of lines of code. Plus, it let sysadmins use their preferred archiving tool to open anything. Early in my career, I learned to be comfortable using tar(1) in the some way certain circus performers are comfortable slipping a tractor/trailer tow chain up their nose and out their ear. Today, I use tar to open those pesky CAB files that are such a burden on sysadmins.

Meanwhile, GNU tar still relies on file extensions.

I don't know how Linux people cope. Maybe that's why they so fiercely cuddle their penguins.

Can libraries be taken too far? No. Only vision can fail. Why, one night at BSDCan a few FreeBSD developers who'd had more liquor than sleep had the spark of genius to implement and publish libtrue, a back-end to the true(1) program that could be linked into any program. Sadly, the world failed to pick up on this magnificent innovation and libtrue remains underadopted.

If you want to be a developer and make the world better, study your nuclear waste with an eye towards reducing it. Does it have ancient functions that can now be served by a well-maintained—mostly maintained—er, maintained at all, in any way—library? Are there common features that should be in a library?

How can you reduce the amount of code in the world?

Because code is unquestionably poison. Just look at what it's done to you, making you question my ethics when it's obvious I don't care.

Have a question for Michael?
Send it to letters@freebsdjournal.org

MICHAEL W LUCAS is the author of *TLS Mastery*, *Absolute FreeBSD*, and the *$ git commit murder series*. His *DNSSEC Mastery* and *Domesticate Your Badgers* should be out first thing in 2022, and it's far too late for you to stop him. Submit your questions to letters@freebsdjournal.org.