WeGet letters

by Michael W Lucas

Oh Fiercely Fearless Feckless Letters Entity,

I've used open-source software like FreeBSD for ages now. It's improved my life and I want to give back. But I can't seem to find a way into a project. Either my contributions are ignored or my skills don't apply or my work is rejected. I feel totally stuck on the outside. How can I help my favorite project?

—Eager But Baffled

Dear EBB,

I can't decide if you thought I wouldn't know what the word "feckless" meant or if you thought I'd recognize its truth and go with it. For the record, it's the latter.

I suspect you've fallen victim to Sysadmin Rule #17: You are solving the wrong problem.

It's easy to look at a major open-source project and say, "Wow, oooh, I want to do that." It's also easy to look at professional skateboarders, rugby players, and cage fighters and say the same. You want to become a particular sort of major player. The vision of yourself as a digital version of Samuel L. Jackson or Jason Statham might delight you, but the fantasy overlooks a tiny, minor, minuscule detail.

Becoming a major player takes a lot of work.

Skateboarders spend years faceplanting before they can do those fancy tricks. Cage fighters get punched and kicked and wedgied for years, nonstop, before winning their bouts. And let's not discuss rugby.

The major players of technology achieved that skill level by spending decades getting face-punched by compiler errors, kicked by bugs in hobnailed boots, and deluged by petulant, ineffective, and downright nonsensical technology changes. Many folks who work at the peaks of open source have additional honorifics after their name, like "PhD" or "Snobby McSnooty Prize for Computing" or "hacker or hackers unknown."

Are you on the couch watching rugby and wishing you could do that? Or are you spending each day lifting weights, running marathons, and thwacking your tenderest anatomy with a ball peen hammer before each meal to get ready to join the pub team and grind your way up?

Ask yourself that question. Then decide which of three paths to take.

The first, "give up," is unworthy of my time and attention. You are fully qualified to implement this plan without my guidance.

Solution two would be to change yourself. If you want to be the next Major Player of FreeBSD, do the work to become that. Can you program in C? If not, learn that. Yes, fancy languages like Perl and Smalltalk and Fortran get all the glory, and the kids like some weird thing named after a snake, but the project's language is C, and they're not going to change to fit

your prejudices. If you want to develop a Haskell kernel, join a different project. I'm sure you'll have many to choose from.

The best way to learn a programming language is to want to accomplish a task in that language.

A smart person would start with the bug database. Look for bugs that you can replicate with your system. Replicate them, contact the reporter for more information, and start digging. Develop patches that resolve the problem. You'll earn the respect of the community.

Maybe you don't want to clean up someone else's bugs. You want to do something big and world changing. Large open-source projects are not about writing exciting new components from scratch, but rather about slow evolution. Major projects happen, but they go to respected people in the community. You can try it, though.

Back in a previous century, early in my systems administration career, I found that I needed an FTP client that could do recursive fetches. While three or four FTP clients existed—and half of them even worked, more or less—my life would have been much simpler if the client included in FreeBSD 2.x had that feature. I wanted to hone my knowledge of C. I'd read the first edition of Kernighan and Ritchie's The C Programming Language and had written a few petty buffer overflows disguised as software. The cliché was that open source was about scratching itches—well, I had the itch, I should be the one to scratch it.

I started reading source code.

And rereading source code.

Then I printed the source code for ftp(1), taped it on my home office wall, and went at it with several colors of highlighters, red and green pens, marking sections of code that my inadequate brain thought were important or relevant and connecting disparate functions together. It was like one of those conspiracy theorists' corkboards all tied together with string, except it was demonstratably real and every time I touched any bit of it the whole thing imploded.

Had I taken the time to ask any FreeBSD committer about my little project, they would have steered me to something more realistic, like juggling saber-toothed porcupines. Or at least advised me to wear protective gear against the inevitable ball-peen hammers. I learned vast amounts about C. I learned how to handle core dumps. I learned exactly how inadequate my programming skills were.

I could have become a programmer. All I had to do was keep pounding away at the problem.

After several months of bending my brain, I removed everything except the printouts from the office and moved out of that house, in the hope that the new tenants would be wiser than myself and stay off the Internet forever.

I chose the third way.

You have skills in something, presumably. In your years, you've continued breathing and maintained essential digestive functions. When you consider that most of the people who have ever lived are now dead, you're doing pretty well.

Take the thing that you do and do it for your project.

Maybe you like helping people with problems you've already solved. Every project has formal or informal support channels, mailing lists, forums, whatever. Hang out on those forums, helping people.

You know who gets plaudits and kudos from developers? Testers.

Before release day, download the release-candidate install media. Make a checklist of all installer options, and methodically test them all. Report errors.

If for some deranged reason you want something more programmery, learn how to write and use software tests.

If your job involves some weird edge of the software, blog about your experiences in that edge. Or scribble some notes for the wiki. Perhaps you're among the truly unfortunate and can write. No further action is necessary on your part at this time, but don't worry. The cabal will be by shortly with tasers and nets. Don't struggle. It won't help.

The point is: do the thing you do. But do it for FreeBSD.

It's okay to change what you do. Time spent helping other users makes people's lives better. Time spent documenting how you can tune a filesystem but you can't tuna fish does the same. The time I spent methodically studying source code improved my code reading skills, which I rely on today. On a community level, people will remember you.

Specifically, they'll remember the quality of your work. They will treat you with the seriousness that the quality of your previous work merits.

You can't wedge your way into a community. Open source doesn't work that way. It's not a block party where the right mailing address grants you admission. The only way to gain admission to an open-source community is by completing quality work. If you can do that, you won't have to fight to gain entrance. More and more community members will tie social strings to you until you find yourself sentenced to life in that community. People will bribe you into taking on the community's absolute worst jobs, simply because you have a reputation for completing that kind of task successfully.

What kind of horrid jobs? You know, like answering a letters column for the community journal.

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

MICHAEL W LUCAS (https://mwl.io)'s newest books are *SNMP Mastery* and *Terrapin Sky Tango*.