Can free software be made sustainable?

Adding money to FOSS without fucking it up

Drew DeVault

SourceHut

September 2, 2022

First: Can FOSS work without money?

Yep! Most projects are written by volunteers.

Important: Volunteers should have boundaries. You don’t owe anything to random strangers on the internet — and they don’t owe you anything, either.

This is the hard part

Starting a FOSS project is easy. Starting a business is difficult.

There is no magic pill.

It will be hard work.

The ground rules

- Free & open source *only*, as defined by the FSF or OSI
- No BSL, SSPL, Commons Clause, etc
- No open core

Who owns the project?

In the absence of a Contributor License Agreement (CLA), **each contributor retains the copyright for their contributions**.

Thus, the project literally – and legally – belongs to all of its contributors.

Each contributor *licenses* their contributions to everyone else under the terms of the project's license.

Who is allowed to monetize it?

Everyone has the right to monetize any free software project: maintainers or contributors or users, individuals, governments, small companies, megacorps. **In the absence of this right, the project is non-free.**

So how do you do it?

Let’s cover a few of the ways.

Just sell the damn thing!

The simplest way to monetize free software is to simply sell the software!

The ground rules: Anyone who receives the program, in source or executable form, is entitled to the source code with a free software license. Thus you can:

- Sell binaries and give away the source code
- Sell the source code and the binaries
- Sell the source code and let the user build it themselves

However: If you sell *or* give away the executables, you must provide the source code at no extra charge.

But if it’s free software...

Problem: Can’t users re-distribute (or even sell) the source code without paying you?

Answer: Yep! So what are you going to do about it?
Make users *want* to get it from you rather than from someone else.

Another approach: trademark enforcement. Tread carefully.

Asking for donations

The easiest solution to implement, but also the least effective.

- Tools: OpenCollective, LiberaPay, fosspay
- Individual donations or commercial sponsorships? Both?

Applying for grants

Free money! Organizations like NLNet and RIPE are good for this.

- Requires less overhead and works for a wide variety of projects
- But not guaranteed nor reliable in the long term

Industry collaborations

Many free software projects are jointly developed by several independent companies within the industry that relies on them. Prominent examples include Linux, Mesa, Kubernetes, OpenStack, etc.

- Optimize for and encourage commercial contributors
- Get them invested in the project, give them responsibilities
- Don't let them run away with the code: use copyleft
- Encourage an upstream-first culture

Consulting

Simply make yourself available for hire.

- Works best if your project is commercially useful
- Works on a large or small scale
- You can be hired to work on any project

Selling learning resources

You can produce and sell:

- Books
- Magazines
- Online courses
- In-person courses
- Certifications

Potential conflict of interest: Try not to make your public documentation worse in order to drive conversions.

Sell pre-configured hardware

Another approach is sell hardware with free software pre-installed.

- Works great for open hardware projects with a software component
- Consumers are willing to pay a premium on good open hardware (within reason)
- However: This is the most logistically challenging monetization model

Free to self-host, paid for SaaS

Facts:

- Servers are expensive
- Sysadmins are expensive
- Learning the skills to do this *well* is expensive
- People will pay for convenience and reliability

These points of friction create a market opportunity for many projects. You are the most qualified person to operate your product — sell that expertise. And consider using AGPL!

Bonus: mix and subsudize

You can build more than one project and use different monetization models as appropriate to each, and let more profitable ventures subsudize less profitable ones.

This is SourceHut's strategy.

All of the above

Mix and match strategies according to the needs of your project. Come up with new ways. Apply your hacker mindset to the problems of the business as equally as to the problems of the software. Be creative!

How can I help?

Any questions?

`https://drewdevault.com`
sir@cmpwn.com