Marks of a Good FOSS Project

•••

Neal Gompa (Conan Kudo [ニール・ゴンパ])

Who am I?

- Professional technologist
- Contributor and package maintainer in the Fedora Project
- Contributor and package maintainer in Mageia Linux
- Contributor to RPM, DNF, and various related projects
- Diligent follower of the telecommunications industry
- Production Engineer at Datto, Inc.

Contact Points:

- Twitter: @Det_Conan_Kudo
- Google+: +NealGompa

There are all kinds of FOSS projects...

Just like there are all kinds of people, there are all kinds of software projects. There are plenty of projects that are small, one-person efforts (hosted on sites like GitLab, SourceForge, BitBucket, GitHub, or even on a personal website), as well as massive projects involving hundreds of people across multiple regions of the world.

Some projects are rigidly developed, others are more fluid. Some are highly targeted, while some are very broad. There are even different levels of maturity for a software project.

So how do you judge if a project is healthy and worth paying attention to (or even contributing to)?

Judging FOSS projects is hard...

Where to begin…?

How to judge a project on its merits depends on what merits you care about most. However, there are definitely some common qualities that matter no matter what the focus is for judging a particular project.

- Contributor base
- Interaction points
- Code quality
- Quality of documentation
- Direction for Contributing

Contributor base

The contributor base of a project encompasses the project creators/leaders, and various people who are regularly involved in the advancement of the project.

For most FOSS projects, these are usually heavily slanted towards programmer-types, which doesn't necessarily imply that other types aren't wanted or needed. In fact, usually projects with this kind of slant welcome people to offer other kinds of contributions (artwork, documentation, user support, and so on).

A large base of contributors *can* (but does not always) imply that the project has found substantial use. Who those contributors are (and to some extent, who they work for) matters as well. For example, if there's a large contributor base, but ***only*** made up of past and current employees of a firm, there is a weakness of some kind. A diversity of contributors shows that people with different views and goals are able to effectively work together in the project.

Interaction points

A big part of what makes projects successful is communication. Without being able to effectively communicate ideals, timelines, milestones, goals, tasks, and so on, it’s simply not possible to grow a project.

Small projects that are new typically only have an issue tracker and an email address to the developer (especially if hosted on sites like GitLab, GitHub, BitBucket, etc.). This is not necessarily bad as long as the developer is responsive to comments and other types of feedback.

Larger projects that are well-organized typically have a forum of some kind (mailing list, BBS, etc.) and IRC channels along with issue tracker and email contacts.

Code Quality

This is usually more of an issue for prospective contributors or users that intend to modify the code of the project.

Quality of code is comprised of multiple factors: the language, the code style, patterns for implementing features (such as including unit tests to verify the code works as expected), and so on.

Many of these attributes also define the types of code contributors who are likely to help improve the code. Knowing the “culture” of the programming language is critical to understand what is expected for code contributions!

Quality of Documentation

In most cases, quality of the documentation provided by the project is naturally weak, because initial users and contributors often just “get” how everything works. As projects grow, the need for better documentation becomes more important.

However, it’s generally accepted that larger projects usually have semi-decent documentation, though there’s usually a level of self-reliance by user communities that form around them to help people use the software.

In almost every project, documentation is constantly in need of improvement!

Direction for Contributing

Most small projects are just hosted on a code hosting site like GitLab, BitBucket, or GitHub, which offer an easy model for contributing to the advancement of a particular project. Larger projects may have a particular workflow they want people to use for contributing, and it’s important that the information is readily available.

However, some projects require legal agreements in order to begin the process to contribute, which can range from simple contributor agreements (assuring it’s your own work and you’re authorized to contribute) to complex license agreements that require handing over special rights to another entity that isn’t otherwise granted.

Projects with complex license agreements should be carefully scrutinized, as the project may be structured in such a way that it is possible to be FOSS only long enough to get to a point, and then switch to being proprietary afterward. Alternatively, it may be set up to make you give away rights you may not have (either due to laws or employment restrictions). Be careful with these, and be sure to understand the community and what you are allowed to do before proceeding with projects like these. Ideally, projects shouldn’t need such complex agreements.

If there’s no obvious directions on how to contribute, simply ask in one of the available discussion channels!

The most important aspect is the people involved!

At the end of the day, if the people aren’t friendly and welcoming and making contributing as easy as possible, the project isn’t likely to be very successful.

The personalities involved in a project are also key. But also keep in mind that “common knowledge” on people may not necessarily be true, so keep an open mind and investigate yourself. If the people in the project are personable and helpful, it’s usually easier to get involved, which makes it more likely for a project to grow and advance.

Beware of people that try to abuse the community forming around a project! These people will happily accept contributions that fit in their strict model without discourse.

The End

Any Questions?