Getting Involved in FOSS

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 Unity Linux
- Contributor to RPM, DNF, and various related projects
- Diligent follower of the telecommunications industry
- Systems Engineer at Datto, Inc.

Contact Points:

- Twitter: @Det_Conan_Kudo
- Google+: +NealGompa

Introduction to “FOSS”

Free Software / Software Libre

- FOSS began as simply “Free Software”, in reference to freedom
- Defined and codified by the Free Software Foundation in 1986
- Free Software guarantees four freedoms
  - The freedom to run the program as you wish, for any purpose. (freedom for purpose)
  - The freedom to study how the program works, and change it so it does your computing as you wish. (freedom to access and modify the source code)
  - The freedom to redistribute copies so you can help your neighbor. (freedom for distribution)
  - The freedom to distribute copies of your modified versions to others. (freedom to distribute changes)
- Due to ambiguity of the word “Free” in English, it is often called “Libre Software” or “Software Libre”, borrowing “Libre” meaning freedom in Romance languages like French and Spanish

Open Source Software

- The term “Open Source” was introduced by the Open Source Initiative in 1998 just after Netscape released the Mozilla codebase to the public.
- Like “Free Software”, the intent is to provide software that follows the four freedoms.
- The focus by the OSI is to shift from the philosophical perspective to a utilitarian one, focusing on the typical tangible effects of developing Free Software. The sociopolitical aspects of Free Software are completely ignored.
- This fundamental difference has created a schism between the communities that follow the Free Software philosophy vs the Open Source one.

So you’ve found a project you like and want to help out...

Step 1: Know how people communicate in the project

- If the project has more than one person involved, there usually is some preferred project-wide mechanism to communicate.
- Multi-person projects typically prefer one or a combination of the following:
  - IRC (Internet Relay Chat)
  - Project mailing lists
  - Project bug tracker
- If the project has only a single person primarily involved, then usually the communication mechanism is email or through the project bug tracker

Step 2: Know what you can do for the project

- Note that this need not be code contributions. Many projects also have need for people to work on things beyond code.
- Most larger projects have dedicated pages describing how people can join and contribute.
  - Examples: Contribute to Mageia, What can I do for Mozilla?, Get Involved in LibreOffice
- Many smaller projects sometimes include a document describing how to contribute in their source code tree, usually called HACKING, CONTRIBUTING, or some variant of that.
  - Examples: Neovim CONTRIBUTING.md, PackageKit HACKING

Step 3: Follow the rules for the type of contribution

- Many projects have rules, guides, or procedures for how they want contributions of any given type.
  - For example, many projects don’t accept translations directly, but point to a translation management service like Zanata or Transifex.
- Read up on these and follow them, as they usually are there for a reason!
  - Not following them results in wasted time for members of the project and you, as usually you have to go back and redo everything properly.
- If there are requirements before contributing, then do this *before* committing to contributing.
  - For example, the Free Software Foundation requires contributors to sign an agreement that transfers the copyrights of their contributions to them for the purposes of defending the software.

Step 4: Announce Yourself! (and do it well!)

- Putting your best foot forward as part of your introduction will make a huge difference in how others interact with you and review your work.
- Almost all projects have some (formal or informal) mechanism for reviewing contributions which involve regular communication.
- Maintain civility as best as possible, and do not be offended when strong critiques are given to you. Everyone wants to make the project better, and keeping this in mind will help in maintaining perspective.

You've now contributed to FOSS, congratulations!!!

Post-contribution tips

- Unless the project is deliberately structured for “drive-by” contributions, it’s usually a good idea to stick around. Especially for code contributors, as code changes might have other project members asking you about your code or requesting assistance to update code.
- Many projects desire for new contributors to become regular contributors, as whatever you assisted with are usually things they need on a regular basis.

And remember...
Have fun!

The End

Any Questions?