Librem 5 Development Policy Documentation

Release 0

Guido Günther

Dec 04, 2019

CONTENTS:

CHAPTER

ONE

INTRODUCTION

This document desribes the policy requirements for the Librem 5. It's purpose is to reflect current practice for developing on the Librem 5. It's uses the wording from rfc2119.

CHAPTER TWO

SOFTWARE

General policy for software we build and distribute:

0. All source code should be maintained in a public git repository in the *Librem5* space on https://source.puri.sm.
1. All developed software must conform to a OSI compatible license.
2. For software where we're upstream the GPLv3 should beused for programs while the LGPLv3 should be used for libraries.
3. If we're not upstream, upstream's license should be used.
4. We must not require a Contributor License Agreement (CLA).
5. The projects should use a Developer Certificate of Origin (DCO).

CHAPTER

THREE

GIT(LAB) REPOSITORY

Policy regarding git repository layout and configuration. This applies to repositories in the *Librem5* space on https://source.puri.sm:

0. The default branch of the repository must be the branch where people should contribute to if they want to advance development. It must not be an outdated, unused branch or the master branch of the upstream repository if we maintain a downstream fork.
1. Canonical branch naming should be used. If we're upstream the main development branch should be called *master*, if we're maintaing a phone specific fork of a software, the branch should be called *librem5*. These branches must be protected branches that don't allow force pushes.
2. The description in Gitlab must describe the repositories purpose. If it's a mirror from somewhere else this must be mentioned. If it's no longer maintained too.
3. The repository should have a *README.md* that tells people where to report bugs (which is the minimal entry point to interact with the maintainers). Other channels like irc, matrix, mailing lists should also be mentioned there.
4. Each repository should have a doap file that documents the current maintainers.
5. Contributions should happen via merge requests in Gitlab to allow for code review and automated testing. Direct pushes should be avoided.
6. Merge request should happen from a personal fork of the repository not from branches within the repository.
7. The repository should have a *.gitignore* file that sets up git to ignore generated artifacts. The aim is that after a build the repositories status is still clean.

CHAPTER

FOUR

GITLAB CI

0. Each repository should have a Gitlab CI configuration. For software where we're upstream it should be stored in *./.gitlab-ci.yml* in the git repository. This configuration must at least build the artifacts (binaries, documentation) and should run the unit tests that don't require any special setup.
1. The CI pipeline of each repository should be green since everything else drives possible contributors away and makes it harder for contributors to evaluate the impact their changes. Temporary breackage will happen but it should not be the norm.
2. The CI pipeline should produce artifacts for easy consumption. A Debian package is a must for system components. Applications should produce a flatpak.
3. The CI should run against multiple platforms. E.g. prepare for the future by building/testing against Debian testing as well.

CHAPTER

FIVE

DEVELOPMENT

1. Allow for short build, test cycles. Software that needs lots of manual commands to run is annoying for contributors.
2. Being able to run directly from the source tree should be possible so contributors don't need to clutter their systems for smoke testing. If things need special setup provide a *run* script that performs that. This does imply that *every* aspect of the software is testable that way.
3. Since the project is GNOME based GTK should be used as GUI toolkit.
4. Developed libraries should be GObject introspectable to ease use from other programming languages.
5. Applications should follow the GNOME Hig.

CHAPTER

SIX

DEBIAN PACKAGING

This chapter documents Debian packaging policy. It only applies to packages building at least one Debian package.

0. Debian policy should be followed, preferably version 4.4.0 or later.
1. *debian/README.source* should contain instructions if there are special steps required to import new upstream sources or to build the package
2. If you're modifiying / backporting non-native Debian packages you should use pristine-tar and a *pristine-tar* branch. This will make sure the build systems are able to reproduce a bit identical upstream tarball from the git repository.

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search