# Free/Libre and Open Source Software:

# How to get started?

Vanier FLOSS Day 25
Tuesday, October 14th, 2025
Montreal, Quebec, Canada

Qian Qian (@Qubik65536)
https://qubik.top

# Remember:

1. It’s not just about the code.
2. It doesn’t have to be big.

# Why contribute?

- Help improving the software —— that **you** are using.
- Learn new skills & Gain experience.
    - Code
    - Documentation/Writing
    - Design
    - Testing
    - Community Building
    - Team Work
- Meet new people & Make friends.
- Build your portfolio.
- Get recognized or even paid.
- Have fun!

**“How do I start contributing?”**

based on *my own* experiences

# What can I do?

# What kind of contributions can you make?

```
>code_
```

# What kind of contributions can you make?

event organization

bug reports

beta testing

marketing

`>code_`

translation

documentation

design

community support

# Code is only one part of open source.

By Pratham Patel, posted on IT’S FOSS.
https://itsfoss.com/hacktoberfest-guide/

**You don’t have to contribute code:**
A common *misconception* about contributing to open source is that you need to contribute code. In fact, it’s often the other parts of a project that are most neglected or overlooked. You’ll do the project a huge favor by offering to pitch in with these types of contributions!

Open Source Guides: How to Contribute to Open Source
https://opensource.guide/how-to-contribute/

> I’ve been renowned for my work on CocoaPods, but most people don’t know that I actually don’t do any real work on the CocoaPods tool itself. My time on the project is mostly spent doing things like documentation and working on branding.

@orta, “Moving to OSS by default”
Open Source Guides: How to Contribute to Open Source
https://opensource.guide/how-to-contribute/

# Remember:

1. It’s **not** just about the code.
2. It **doesn’t** have to be big.

# What kind of contributions can you make?

event organization

bug reports

beta testing

marketing

>engineering

translation

documentation

design

community support

# “How do I get started?”

# *1. Get to know & understand the project.*

Things may seem overwhelming and confusing at first.
**Don’t worry!** It’s not just you, and helps are around.

For anything more than a typo fix, contributing to open source is like walking up to a group of strangers at a party. If you start talking about llamas, while they were deep in a discussion about goldfish, they’ll probably look at you a little strangely.

Before jumping in blindly with your own suggestions, start by learning how to read the room. Doing so increases the chances that your ideas will be noticed and heard.

Open Source Guides: How to Contribute to Open Source
https://opensource.guide/how-to-contribute/

Every open source community is different.

You may have a lot of experiences with a certain projects that you spent years one. But when you move to a different one, you may find that things are done very differently (this can be code style, workflow, communication style, etc.).

***Good News***: many open source projects follow a similar organizational structure, and following a few general guidelines can help you get started in most projects.

**Do your own research *before asking about everything*.**

## **Do your own research *before asking about everything*.**

People are often happy to help out and are usually very friendly.

**But**, they have their own work to do. So it is often expected that you do your own research first, before asking questions that can be easily answered by reading the documentation, searching the issue tracker, or searching the web.

They *can* become frustrated if you ask too many questions that you could have answered yourself with a little effort.

# RTFM!

Read The F*&%king Manual!

# Read The Documentations!

Every project should have some form of documentation, which is often the first place to look for answers. This may include:

- `README` file: a file containing an overview of the project, and information to welcome new community members to the project (i.e. how to get started).
- `CONTRIBUTING` file: a file containing guidelines for contributing to the project.
- `CODE_OF_CONDUCT` file: a file containing guidelines for community behavior.
- Some other documentation files/sites: often linked from the `README` file, containing more detailed information, for example, detailed tutorials on how to use the project, or technical details. Large projects often have this kind of documentation, sometimes even multiple sites for different audiences (e.g. users vs developers). For example, *Python* has:
  - a documentation site for users: https://docs.python.org/3/, and
  - a dev guide for developers: https://devguide.python.org/

## Read The Documentations!

Note that even though not all projects have `CONTRIBUTING`, `CODE_OF_CONDUCT`, or a developer guide, those who have one are usually more welcoming to new contributors and takes community contributions more seriously.

## Read The Documentations!

By definition, all open source projects SHOULD also have a `LICENSE` file, containing the open source license the project is under.

It *might not* be super important for you to read the entire license, but you should at least know which license the project is under, and what it means for you as a contributor (e.g. whether you need to sign a Contributor License Agreement (CLA) before contributing, what rights you have regarding your contributions, how can your code be used in the future, etc.).

## Know where to talk.

Most open source projects use one or more of the following tools to organize discussion. Make sure you know where to go for discussion, and what is used for what purpose.

## 1. Issue Tracker

(e.g. GitHub Issues, GitLab Issues, JIRA, Bugzilla, etc.)

These are the places where bugs/issues and feature requests are tracked and discussed.

If you have a bug to report and a want to suggest a new feature, this is the place to go. (Make sure you read the guidelines first and make sure what you want to report/suggest is not already reported/suggested.)

## 2. Pull/Merge Requests

(e.g. GitHub Pull Requests, GitLab Merge Requests, etc.)

These are the places where current in-progress works and changes are being discussed and reviewed (i.e. if changes needs to be made before the contribution can be accepted into the project).

This is also the place where you will submit your own contributions and get feedback on them.

Note that each project will have its own specific guidelines and workflow, get to know them before submitting your own contributions.

## 3. Forums, Mailing Lists, Synchronous Chat Rooms

(e.g. GitHub Discussions, Google Groups, Slack, Discord, etc.)

Some projects have forums, mailing lists, and/or chat rooms for synchronous and/or asynchronous discussions.

This is often the place to go for general questions, discussions on conversational topics, and community building.

Tips:

- Synchronous means that people are online at the same time, so you can get real-time responses. It often means quicker responses but also more casual discussions.
  (e.g. Slack, Discord)
- Asynchronous means that people are not necessarily online at the same time, so responses may take longer. It often means more thoughtful responses and more organized discussions.
  (e.g. Forums, Mailing Lists)
- Some projects may have both, some may have only one, and some may have none at all.

# *2. Talk to people.*

## Talk to people.

Now that you know something about the project and know where to go, it’s time to reach out to the community, discuss, ask questions, and start finding things you can work on.

You **can** just by jumping into the issue tracker and start working on an issue, but it’s often better to start at least by talking to someone. You can:

- under the issue tracker of the issue you are interested in, ask about some advice on how to get started and let others know you are interested in working on it, or
- in a general discussion forum/chat room to, introduce yourself, ask for advice on how to get started, and what you can work on.

This way, you can get some guidance and support from the community, and also know what are the goals and priorities of the project.

## Who should I talk to?

A typical open source community has the following types of people:

- **Author**: The person/s or organization that created the project.
- **Owner**: The person/s who has administrative ownership over the organization or repository (not always the same as the original author).
- **Maintainers**: Contributors who are responsible for driving the vision and managing the organizational aspects of the project (they may also be authors or owners of the project).
- **Contributors**: Everyone who has contributed something to the project.
- **Community Members**: People who use the project. They might be active in conversations or express their opinion on the project's direction.

Open Source Guides: How to Contribute to Open Source
https://opensource.guide/how-to-contribute/

## Who should I talk to?

If you are new to the project, it’s often best to start by talking to the maintainers or active contributors. They are often the most knowledgeable about the project and can help you get started.

# *3. Contribute.*

# Contribute.

*Just do it.*

- Start small. Fix a typo, improve a sentence, or add a missing link in the documentation, do some beta testing.
- Translate the software or its documentation to your language.
- Report bugs you encounter, or suggest new features you would like to see.
- Help others in the community by answering questions or providing support.
- Find an issue, ask for advice, and start working on it.
  - It can be code related work, *or not*.
  - Make sure to follow the contribution guidelines and workflow of the project.
  - Sometimes issues are labeled with `good-first-issue`, `help-wanted`, or similar labels to indicate that they are suitable for new contributors.
- Design a logo, create marketing materials, or help organize an event.
- … etc.

# “Where can I find projects?”

*Always try to start with something* ***you*** *are already using and interested in.*

## Some places to find projects:

- The `awesome-for-beginners` repository on GitHub: a curated list of beginner-friendly projects that organizes its issues in a way that makes it easy for beginners to get started.
- Events like Hacktoberfest: a month-long event in October that encourages contributions to open source projects, with many projects labeling issues as `hacktoberfest` to indicate that they are suitable for new contributors.
- Programs like Google Summer of Code (GSoC): a global program that offers stipends to university students for working on free and open source projects during the summer with a mentor from the project.

***Good Luck, Have Fun!***

# That's it!

⇧ The Slide on GitHub

## Get in touch 👋

📨 me@qubik.top

🌐 https://qubik65536.top

Qubik65536