# Contributing to Octobox

Want to contribute to Octobox? That's great! Here are a couple of guidelines that will help you contribute. Before we get started: Please note that this project is released with a [Contributor Code of Conduct](CODE\_OF\_CONDUCT.md) to ensure that this project is a welcoming place for \*\*everyone\*\* to contribute to. By participating in this project you agree to abide by its terms.

#### Overview

\* [Contribution workflow](#contribution-workflow)

\* [Setup instructions](#setup-instructions)

\* [Reporting a bug](#reporting-a-bug)

\* [Contributing to an existing issue](#contributing-to-an-existing-issue)

\* [Our labels](#our-labels)

\* [Additional info](#additional-info)

## Contribution workflow

\* Fork the project.

\* Make your feature addition or bug fix.

\* Add tests for it. This is important so we don't break it in a future version unintentionally.

\* You can use `bundle exec guard` if you'd like the relevant tests to automatically run each time a file is modified.

\* Send a pull request. Bonus points for topic branches.

## Setup instructions

You can find in-depth instructions to install the correct Ruby version, Postgres, and to set up the database in our [README](README.md#development).

## Reporting a bug

So you've found a bug, and want to help us fix it? Before filing a bug report, please double-check the bug hasn't already been reported. You can do so [on our issue tracker](https://github.com/octobox/octobox/issues?q=is%3Aissue+is%3Aopen+label%3Abug). If something hasn't been raised, you can go ahead and create a new issue with the following information:

\* On which page did the error happen?

\* How can the error be reproduced?

\* If possible, please also provide an error message or a screenshot to illustrate the problem.

If you want to be really thorough, there is a great overview on Stack Overflow of [what you should consider when reporting a bug](http://stackoverflow.com/questions/240323/how-to-report-bugs-the-smart-way).

It goes without saying that you're welcome to help investigate further and/or find a fix for the bug. If you want to do so, just mention it in your bug report and offer your help!

## Contributing to an existing issue

### Finding an issue to work on

We've got a few open issues and are always glad to get help on that front. You can view the list of issues [here](https://github.com/octobox/octobox/issues). Most of the issues are labelled, so you can use the labels to get an idea of which issue could be a good fit for you. (Here's [a good article](https://medium.freecodecamp.com/finding-your-first-open-source-project-or-bug-to-work-on-1712f651e5ba) on how to find your first bug to fix).

Before getting to work, take a look at the issue and at the conversation around it. Has someone already offered to work on the issue? Has someone been assigned to the issue? If so, you might want to check with them to see whether they're still actively working on it.

If the issue is a few months old, it might be a good idea to write a short comment to double-check that the issue or feature is still a valid one to jump on.

Feel free to ask for more detail on what is expected: are there any more details or specifications you need to know?

And if at any point you get stuck: don't hesitate to ask for help.

### Making your contribution

We've outlined the contribution workflow [here](#contribution-workflow). If you're a first-timer, don't worry! GitHub has a ton of guides to help you through your first pull request: You can find out more about pull requests [here](https://help.github.com/articles/about-pull-requests/) and about creating a pull request [here](https://help.github.com/articles/creating-a-pull-request/).

## Our labels

- \*\*beginner\*\*: One of our most important kinds of issues! These issues have been marked by the maintainers as small, low-impact changes and fixes, ideal for those just dipping their toes for the first time into the world of contributing to Open Source software. These are by no means any less important than any other issues! We'd appreciate if our more experienced contributors would leave these to our newer compatriots so we can be as fair as possible to everyone.

- \*\*blocked\*\*: We will generally mark an issue as blocked if it requires more context for us to fix, is waiting on an updated version of a library that the issue depends on, or is waiting on a maintainer to do something before it can be answered. There may be other conditions in which we'll mark an issue as blocked, and we'll make this clear in the issue comments.

- \*\*bug\*\*: An issue gets marked as a bug if (surprise!) the issue is linked to a bug in our code or something broken in our infrastructure. Bugs are usually the highest priority kind of issue as they will affect our users.

- \*\*bugsnag\*\*: If an issue or exception in production code has been flagged via Bugsnag, it is labelled with this automatically.

- \*\*documentation\*\*: If an issue requires changes or clarification to our documentation (READMEs, wiki, code comments, etc.) then it will be marked with this issue. These issues normally do not pertain to code changes and are ideal for those who don't necessarily want to get their hands dirty with code.

- \*\*duplicate\*\*: If an issue has been reported before, it will be marked with this label. This flags that a previous issue may not have been fixed as expected, or in the case of things we don't intend to fix or issues currently in discussion in another open issue, it means that we are likely to close the issue.

- \*\*enhancement\*\*: These are improvements to the codebase that are "nice to have", and are usually medium to high impact but low importance. This includes improvements to features that already exist and may be assigned a milestone to give you an idea as to when it might be implemented. These are great for contributors that might have some extra spare time on their hands and would prefer to change existing code rather than add large chunks of brand new code.

- \*\*feature\*\*: An issue will be marked with this label if it adds a new, discrete functional element to the codebase. They will generally require large amounts or effort to implement, require changes to both front-end view templates and back-end code, and assume prior knowledge of the codebase, making these the most tricky kinds of issues to close. However, they can be the most rewarding to close for the intrepid!

- \*\*help wanted\*\*: Issues will be marked with this label by the maintainers if we'd like outside contributors to pitch in with code or opinions before we close them. These do not assume any skill level and are great opportunities for all members of our community to steer the direction of the project.

- \*\*i18n\*\*: Issues related to localisation or translation are marked with these labels. They can range from missing translations for a specific language to grammatical issues and cultural differences. These will usually require a specialist in a non-English language to close.

- \*\*invalid\*\*: These are rare, and normally have been submitted in error.

- \*\*needs rebase\*\*: You will find this label normally attached to pull requests, and means that the maintainers would like to you squash commits or rebase existing commits from master into your branch before we can merge your pull request. A maintainer will clarify this in the comment thread.

- \*\*question\*\*: Issues can be marked with this label by anybody who would like other contributors or maintainers to answer a specific question before an issue can be closed. These normally do not assume any skill level (although may sometimes require maintainers to have the final say on them) and are great opportunities for all members of our community to steer the direction of the project.

- \*\*refactoring\*\*: If an issue requires code to be refactored before a particular change can be made, or if you spot inelegant patterns or implementations in code that you feel could be better, then feel free to add this label to an issue. If the issue is the latter type, please be careful about the language you use in these threads. For example, things like "this code sucks!" or "you must be an idiot!" are unacceptable!. Programmers have feelings too and there are ways to suggest code changes without insulting people!

- \*\*security\*\*: If you spot an issue related to security issues (e.g. invalid SSL certs, potential CSRF issues), mark it with this label. Don't forget to be a good OSS citizen and always report zero-day issues through a private channel to minimise impact to your fellow users! Check our [vulnerability disclosure policy](VULNERABILITY\_DISCLOSURE\_POLICY.md) for more information.

- \*\*UX\*\*: These issues relate to how the site works, or more holistically, how the site \_feels\_ to the end user, and will normally be related to the front-end of the website. These issues might typically relate to confusion stemming from navigation, form elements, input validation, or breaks in user flows.

- \*\*visual design\*\*: This relates to visual elements of the site that could be improved or are broken. This may be as simple as a change to a CSS colour or font size but may stretch all the way to things that do not render as expected on mobile devices. These are usually good issues for people with front-end development experience to tackle.

- \*\*wontfix\*\*: This means that, after considering your issue in full, your issue is outside of the intended scope of the project and is not something we'd like to add to the codebase right now or in the future. These are used sparingly and are intended to be rare, and are never used without reasoned justification.

## Additional info

Especially if you're a newcomer to Open Source and you've found some little bumps along the way while contributing, we recommend you write about them. [Here](https://medium.freecodecamp.com/new-contributors-to-open-source-please-blog-more-920af14cffd)'s a great article about why writing about your experience is important; this will encourage other beginners to try their luck at Open Source, too!