# Contributing to Chef Projects

We're glad you want to contribute to a Chef project! This document will help answer common questions you may have during your first contribution.

## Submitting Issues

Not every contribution comes in the form of code. Submitting, confirming, and triaging issues is an important task for any project. At Chef we use GitHub to track all project issues.

If you are familiar with Chef and know the component that is causing you a problem, you can file an issue in the corresponding GitHub project. All of our Open Source Software can be found in our [Chef GitHub organization](https://github.com/chef/). All projects include GitHub issue templates to help gather information needed for a thorough review.

We ask you not to submit security concerns via GitHub. For details on submitting potential security issues please see <https://www.chef.io/security/>

In addition to GitHub issues, we also utilize a feedback site that helps our product team track and rank feature requests. If you have a feature request, this is an excellent place to start <https://www.chef.io/feedback/>

## Contribution Process

We have a 3 step process for contributions:

1. Commit changes to a git branch, making sure to sign-off those changes for the [Developer Certificate of Origin](#developer-certification-of-origin-dco).

2. Create a GitHub Pull Request for your change, following the instructions in the pull request template.

3. Perform a [Code Review](#code-review-process) with the project maintainers on the pull request.

### Pull Request Requirements

Chef Projects are built to last. We strive to ensure high quality throughout the experience. In order to ensure this, we require that all pull requests to Chef projects meet these specifications:

1. \*\*Tests:\*\* To ensure high quality code and protect against future regressions, we require all the code in Chef Projects to have at least unit test coverage. We use [RSpec](http://rspec.info/) for unit testing.

2. \*\*Green CI Tests:\*\* We use [Buildkite](https://buildkite.com/chef-oss) to test all pull requests. We require these test runs to succeed on every pull request before being merged.

### Code Review Process

Code review takes place in GitHub pull requests. See [this article](https://help.github.com/articles/about-pull-requests/) if you're not familiar with GitHub Pull Requests.

Once you open a pull request, project maintainers will review your code and respond to your pull request with any feedback they might have. The process at this point is as follows:

1. Two or more members of the owners, approvers, or reviewers groups must approve your PR. See the [Chef Infra OSS Project](https://github.com/chef/chef-oss-practices/blob/master/projects/chef-infra.md) for a list of all members.

2. Your change will be merged into the project's `master` branch

3. Our Expeditor bot will automatically increment the version and update the project's changelog with your contribution. For projects that ship as a package, Expeditor will kick off a build which will publish the package to the project's `current` channel.

If you would like to learn about when your code will be available in a release of Chef, read more about [Chef Release Cycles](#release-cycles).

### Developer Certification of Origin (DCO)

Licensing is very important to open source projects. It helps ensure the software continues to be available under the terms that the author desired.

Chef uses [the Apache 2.0 license](https://github.com/chef/chef/blob/master/LICENSE) to strike a balance between open contribution and allowing you to use the software however you would like to.

The license tells you what rights you have that are provided by the copyright holder. It is important that the contributor fully understands what rights they are licensing and agrees to them. Sometimes the copyright holder isn't the contributor, such as when the contributor is doing work on behalf of a company.

To make a good faith effort to ensure these criteria are met, Chef requires the Developer Certificate of Origin (DCO) process to be followed.

The DCO is an attestation attached to every contribution made by every developer. In the commit message of the contribution, the developer simply adds a Signed-off-by statement and thereby agrees to the DCO, which you can find below or at <http://developercertificate.org/>.

```

Developer's Certificate of Origin 1.1

By making a contribution to this project, I certify that:

(a) The contribution was created in whole or in part by me and I

have the right to submit it under the open source license

indicated in the file; or

(b) The contribution is based upon previous work that, to the

best of my knowledge, is covered under an appropriate open

source license and I have the right under that license to

submit that work with modifications, whether created in whole

or in part by me, under the same open source license (unless

I am permitted to submit under a different license), as

Indicated in the file; or

(c) The contribution was provided directly to me by some other

person who certified (a), (b) or (c) and I have not modified

it.

(d) I understand and agree that this project and the contribution

are public and that a record of the contribution (including

all personal information I submit with it, including my

sign-off) is maintained indefinitely and may be redistributed

consistent with this project or the open source license(s)

involved.

```

For more information on the change see the Chef Blog post [Introducing Developer Certificate of Origin](https://blog.chef.io/2016/09/19/introducing-developer-certificate-of-origin/)

#### DCO Sign-Off Methods

The DCO requires a sign-off message in the following format appear on each commit in the pull request:

```

Signed-off-by: Julia Child <juliachild@chef.io>

```

The DCO text can either be manually added to your commit body, or you can add either \*\*-s\*\* or \*\*--signoff\*\* to your usual git commit commands. If you are using the GitHub UI to make a change you can add the sign-off message directly to the commit message when creating the pull request. If you forget to add the sign-off you can also amend a previous commit with the sign-off by running \*\*git commit --amend -s\*\*. If you've pushed your changes to GitHub already you'll need to force push your branch after this with \*\*git push -f\*\*.

### Chef Obvious Fix Policy

Small contributions, such as fixing spelling errors, where the content is small enough to not be considered intellectual property, can be submitted without signing the contribution for the DCO.

As a rule of thumb, changes are obvious fixes if they do not introduce any new functionality or creative thinking. Assuming the change does not affect functionality, some common obvious fix examples include the following:

- Spelling / grammar fixes

- Typo correction, white space and formatting changes

- Comment clean up

- Bug fixes that change default return values or error codes stored in constants

- Adding logging messages or debugging output

- Changes to 'metadata' files like Gemfile, .gitignore, build scripts, etc.

- Moving source files from one directory or package to another

\*\*Whenever you invoke the "obvious fix" rule, please say so in your commit message:\*\*

```

------------------------------------------------------------------------

commit 370adb3f82d55d912b0cf9c1d1e99b132a8ed3b5

Author: Julia Child <juliachild@chef.io>

Date: Wed Sep 18 11:44:40 2015 -0700

Fix typo in the README.

Obvious fix.

------------------------------------------------------------------------

```

## Release Cycles

Our primary shipping vehicle is operating system specific packages that includes all the requirements of Chef. The packages are built with our [Omnibus](https://github.com/chef/omnibus) packing project.

We also release our software as gems to [Rubygems](https://rubygems.org/) but we strongly recommend using Chef packages since they are the only combination of native libraries & gems required by Chef that we test thoroughly.

Our version numbering roughly follows [Semantic Versioning](http://semver.org/) standard. Our standard version numbers look like X.Y.Z which mean:

- X is a major release, which may not be fully compatible with prior major releases

- Y is a minor release, which adds both new features and bug fixes

- Z is a patch release, which adds just bug fixes

After shipping a release of Chef we bump the `Minor` version by one to start development of the next minor release. All merges to master trigger an increment of the `Patch` version, and a build through our internal testing pipeline. We do a `Minor` release approximately every month, which consist of shipping one of the already auto-incremented and tested `Patch` versions. For example after shipping 12.10.24, we incremented Chef to 12.11.0\. From there 18 commits where merged bringing the version to 12.11.18, which we shipped as an omnibus package.

Announcements of releases are made to the [chef mailing list](https://discourse.chef.io/c/chef-release) when they are available and are mirrored to the #announcements channel on the [Chef Community Slack](https://community-slack.chef.io/).

## Chef Community

Chef is made possible by a strong community of developers and system administrators. If you have any questions or if you would like to get involved in the Chef community you can check out:

- [Chef Mailing List](https://discourse.chef.io/)

- [Chef Community Slack](https://community-slack.chef.io/)

Also here are some additional pointers to some awesome Chef content:

- [Chef Docs](https://docs.chef.io/)

- [Learn Chef](https://learn.chef.io/)

- [Chef Software Inc. Website](https://www.chef.io/)

- [Chef Project Website](https://www.chef.sh/)