# Contributing to Markdownlint

We're glad you want to contribute markdownlint! This document will help answer common questions you may have during your first contribution. Please, try to follow these guidelines when you do so.

## Issue Reporting

Not every contribution comes in the form of code. Submitting, confirming, and triaging issues is an important task for any project. We use GitHub to track all project issues. If you discover bugs, have ideas for improvements or new features, please start by [opening an issue](https://github.com/markdownlint/markdownlint/issues) on this repository. We use issues to centralize the discussion and agree on a plan of action before spending time and effort writing code that might not get used.

### Submitting An Issue

\* Check that the issue has not already been reported

\* Check that the issue has not already been fixed in the latest code (a.k.a. `master`)

\* Select the appropriate issue type and open an issue with a descriptive title

\* Be clear, concise, and precise using grammatically correct, complete sentences in your summary of the problem

\* Include the output of `mdl -V` or `mdl --version`

\* Include any relevant code in the issue

## Code Contributions

Markdownlint follows a [forking workflow](https://guides.github.com/activities/forking/), and we have a simple process for contributions:

1. Open an issue on the [project repository](https://github.com/markdownlint/markdownlint/issues), if appropriate

1. If you're adding or making changes to rules, read the [Development docs](#local-development)

1. Follow the [forking workflow](https://guides.github.com/activities/forking/) steps:

1. Fork the project ( <http://github.com/markdownlint/markdownlint/fork> )

1. Create your feature branch (`git checkout -b my-new-feature`)

1. Commit your changes (`git commit -am 'Add some feature'`)

1. Push to the branch (`git push origin my-new-feature`)

1. Create a [GitHub Pull Request](https://help.github.com/articles/about-pull-requests/) for your change, following all [pull request requirements](#pull-request-requirements) and any instructions in the pull request template

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

### Your First Code Contribution

Unsure where to begin contributing to Markdownlint? You can start by looking through these beginner and help-wanted issues:

[Beginner issues](https://github.com/markdownlint/markdownlint/issues?q=is%3Aissue+is%3Aopen+label%3A%22good+first+issue%22+sort%3Acomments-desc) - issues which should only require a few lines of code, and a test or two.

[Help wanted issues](https://github.com/markdownlint/markdownlint/issues?q=is%3aissue+is%3aopen+label%3a%22help+wanted%22+sort%3Acomments-desc) - issues which should be a bit more involved than beginner issues.

Both issue lists are sorted by total number of comments. While not perfect, number of comments is a reasonable proxy for impact a given change will have.

### Local Development

\*\*Under Development (Oct 2019)\*\*

<!--

\* Please try not to mess with the Rakefile, version, or history. If you want to have your own version, or is otherwise necessary, that is fine, but please isolate to its own commit so I can cherry-pick around it.

\* [ ] Run `bundle exec rake default`. It executes all tests and Markdownlint for itself, and generates the documentation

-->

### Pull Request Requirements

Markdownlint strives to ensure high quality for the project. In order to promote this, we require that all pull requests to meet these specifications:

\* \*\*Tests:\*\* To ensure high quality code and protect against future regressions, we require tests for all new/changed functionality in Markdownlint. Test positive and negative scenarios, try to break the new code now.

\* \*\*Green CI Tests:\*\* We use [Travis CI](https://travis-ci.org/markdownlint/markdownlint) 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. A review is required from at least one of the project maintainers. See the master maintainers document for Markdownlint project at <https://github.com/markdownlint/markdownlint/blob/master/MAINTAINERS.md>.

1. Your change will be merged into the project's `master` branch, and all [commits will be squashed](https://help.github.com/en/articles/about-pull-request-merges#squash-and-merge-your-pull-request-commits) during the merge.

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

# Releases

We release Markdownlint as a gem to [Rubygems](https://rubygems.org/gems/mdl) and maintain a [Dockerfile](https://hub.docker.com/r/mivok/markdownlint)

Markdownlint follows the [Semantic Versioning](http://semver.org/) standard. Our standard version numbers look like `X.Y.Z` and translates to:

\* `X` is a major release: has changes that may be incompatible with prior major releases

\* `Y` is a minor release: adds new functionality and bug fixes in a backwards compatible manner

\* `Z` is a patch release: adds backwards compatible bug fixes

\*Exception: Versions < 1.0 may introduce backwards-incompatible changes in a minor version\*