# How to contribute #

We'd love to accept your patches and contributions to this project. There are

a just a few small guidelines you need to follow.

## Contributor License Agreement ##

Contributions to any Google project must be accompanied by a Contributor

License Agreement. This is not a copyright \*\*assignment\*\*, it simply gives

Google permission to use and redistribute your contributions as part of the

project.

\* If you are an individual writing original source code and you're sure you

own the intellectual property, then you'll need to sign an [individual

CLA][].

\* If you work for a company that wants to allow you to contribute your work,

then you'll need to sign a [corporate CLA][].

You generally only need to submit a CLA once, so if you've already submitted

one (even if it was for a different project), you probably don't need to do it

again.

[individual CLA]: https://developers.google.com/open-source/cla/individual

[corporate CLA]: https://developers.google.com/open-source/cla/corporate

## Submitting a patch ##

1. It's generally best to start by opening a new issue describing the bug or

feature you're intending to fix. Even if you think it's relatively minor,

it's helpful to know what people are working on. Mention in the initial

issue that you are planning to work on that bug or feature so that it can

be assigned to you.

1. Follow the normal process of [forking][] the project, and setup a new

branch to work in. It's important that each group of changes be done in

separate branches in order to ensure that a pull request only includes the

commits related to that bug or feature.

1. Any significant changes should almost always be accompanied by tests. The

project already has good test coverage, so look at some of the existing

tests if you're unsure how to go about it.

1. All contributions must be licensed Apache 2.0 and all files must have

a copy of the boilerplate licence comment (can be copied from an existing

file. Files should be formatted according to Google's [java style guide][].

1. Do your best to have [well-formed commit messages][] for each change.

This provides consistency throughout the project, and ensures that commit

messages are able to be formatted properly by various git tools.

1. Finally, push the commits to your fork and submit a [pull request][].

[forking]: https://help.github.com/articles/fork-a-repo

[java style guide]: https://google.github.io/styleguide/javaguide.html

[well-formed commit messages]: http://tbaggery.com/2008/04/19/a-note-about-git-commit-messages.html

[pull request]: https://help.github.com/articles/creating-a-pull-request