# Contributing to this project

Please take a moment to review this document in order to make the contribution

process easy and effective for everyone involved.

Following these guidelines helps to communicate that you respect the time of

the developers managing and developing this open source project. In return,

they should reciprocate that respect in addressing your issue or assessing

patches and features.

## Using the issue tracker

The [issue tracker](issues) is the preferred channel for [bug reports](#bug-reports),

[features requests](#feature-requests) and [submitting pull requests](#pull-requests),

but please respect the following restrictions:

\* Please \*\*do not\*\* use the issue tracker for personal support requests (use

[Stack Overflow](http://stackoverflow.com/search?q=antigen) or [Gitter](https://gitter.im/antigen-zsh/develop)).

\* Please \*\*do not\*\* derail or troll issues. Keep the discussion on topic and

respect the opinions of others.

## Bug reports

A bug is a \_demonstrable problem\_ that is caused by the code in the repository.

Good bug reports are extremely helpful - thank you!

Guidelines for bug reports:

1. \*\*Use the GitHub issue search\*\* &mdash; check if the issue has already been

reported.

2. \*\*Check if the issue has been fixed\*\* &mdash; try to reproduce it using the

latest `master` or development branch in the repository.

3. \*\*Isolate the problem\*\* &mdash; create a reduced test

case and a live example.

A good bug report shouldn't leave others needing to chase you up for more

information.

Please try to be as detailed as possible in your report. Be sure to include:

- Antigen, Zsh and OS version

- Antigen and Zsh configurations (usually `~/.zshrc` and `~/.antigenrc`)

- What would you expect to be the outcome?

- What are the steps to reproduce the issue

All these details will help people to fix any potential bugs.

Example:

> Short and descriptive example bug report title

>

> A summary of the issue and the Zsh version/OS environment in which it occurs.

> If suitable, include the steps required to reproduce the bug.

>

> 1. This is the first step

> 2. This is the second step

> 3. Further steps, etc.

>

> `<url>` - a link to the zsh configuration

>

> Any other information you want to share that is relevant to the issue being

> reported. This might include the lines of code that you have identified as

> causing the bug, and potential solutions (and your opinions on their

> merits).

## Feature requests

Feature requests are welcome. But take a moment to find out whether your idea

fits with the scope and aims of the project. It's up to \*you\* to make a strong

case to convince the project's developers of the merits of this feature. Please

provide as much detail and context as possible.

## Pull requests

Good pull requests (patches, improvements, new features) are a fantastic

help. They should remain focused in scope and avoid containing unrelated

commits.

\*\*Please ask first\*\* before embarking on any significant pull request (e.g.

implementing features, refactoring code, porting to a different language),

otherwise you risk spending a lot of time working on something that the

project's developers might not want to merge into the project.

Please adhere to the coding conventions used throughout a project (indentation,

accurate comments, etc.) and any other requirements (such as test coverage).

Follow this process if you'd like your work considered for inclusion in the

project:

1. [Fork](http://help.github.com/fork-a-repo/) the project, clone your fork,

and configure the remotes:

```bash

# Clone your fork of the repo into the current directory

git clone https://github.com/<your-username>/<repo-name>

# Navigate to the newly cloned directory

cd <repo-name>

# Assign the original repo to a remote called "upstream"

git remote add upstream https://github.com/zsh-users/antigen

```

2. If you cloned a while ago, get the latest changes from upstream:

```bash

git checkout <dev-branch>

git pull upstream <dev-branch>

```

3. Create a new topic branch (off the main project development branch `develop`) to

contain your feature, change, or fix:

```bash

git checkout -b <topic-branch-name>

```

> Please use `documentation/<descriptive-branch-name>`, `feature/<descriptive-branch-name>` or `bugfix/<descriptive-branch-name>` for branch names if you are enhancing documentation, adding a new feature or creating a bugfix.

4. Commit your changes in logical chunks. Please adhere to these [git commit

message guidelines](http://tbaggery.com/2008/04/19/a-note-about-git-commit-messages.html)

or your code is unlikely be merged into the main project. Use Git's

[interactive rebase](https://help.github.com/articles/interactive-rebase)

feature to tidy up your commits before making them public.

5. Locally merge (or rebase) the upstream development branch into your topic branch:

```bash

git pull [--rebase] upstream <dev-branch>

```

6. Push your topic branch up to your fork:

```bash

git push origin <topic-branch-name>

```

7. [Open a Pull Request](https://help.github.com/articles/using-pull-requests/)

with a clear title and description.

\*\*IMPORTANT\*\*: By submitting a patch, you agree to allow the project owner to

license your work under the same license as that used by the project.