# Contributing

\*\*Drogon\*\* is an open source project at its heart and every contribution is

welcome. By participating in this project you agree to stick to common sense and

contribute in an overall positive way.

## Getting Started

1. Fork, then clone the repository: `git clone

git@github.com:your-username/drogon.git`

1. Follow the [official installation steps on

DocsForge](https://drogon.docsforge.com/master/installation/). It’s best to

make sure to have the `drogon\_ctl` executable in your shell’s `PATH`

environment variable in case you use a terminal.

Now you can create branches, start adding features & bugfixes to the code, and

[create pull requests](https://github.com/an-tao/drogon/compare).

## Pull Requests

Feel free to [create a pull request](https://github.com/an-tao/drogon/compare)

if you think you can contribute to the project. You will be listed as a

[contributor](https://github.com/an-tao/drogon/graphs/contributors), agree to

this document, and the

[LICENSE](https://github.com/an-tao/drogon/blob/master/LICENSE).

There are also some recommendations you can follow. These aren’t requirements,

but they will make the development more straightforward:

1. If you are unsure about a specific change, have questions, or want to get

feedback about a feature you want to introduce, [open a new

issue](https://github.com/an-tao/drogon/issues) (please make sure that there

is no previous issue about a similar topic).

1. You should branch off the current state of the `master` branch, and also

merge it into your local branch before creating a pull request if there were

other changes introduced in the meantime.

1. You can use the following branch names to make your intent clearer:

\* `bugfix/123-fix-template-parser` when you want to fix a bug in the

template parser.

\* `feature/123-add-l10n-and-i18n` if you want to add localization (l10n) and

internationalization (i18n) as a new feature to the project.

\* If there’s no open issue and no need to open one you can skip the number,

and just use the descriptive part: `bugfix/fix-typo-in-docs`.

1. Write a brief, but good, and descriptive commit message / pull request title in English,

e. g. “Added Internationalization and Localization”.

If you follow these recommendations your pull request will have more success:

1. Keep the style consistent to the project, when in doubt refer to the [Google

C++ Style Guide](https://google.github.io/styleguide/cppguide.html#C++\_Version).

1. Please write all comments in English. Comments for new public API introduced by

your pull request must be added and written in [Doxygen](http://www.doxygen.nl/) format.

1. Format the code with `clang-format` (>= 8.0.0). The configuration is already

provided in the `.clang-format` file, just run the `./format.sh` script

before submitting your pull request.

1. Install [Google Test](https://github.com/google/googletest), and write a test

case.

1. In case it is a bugfix, it’s best to write a test that breaks in the old

version, but works in the new one. This way regressions can be tracked

over time.

1. If you add a feature, it is best to write the test as if it would be an

example how to use the newly introduced feature and to test all major,

newly introduced code.

## Project Maintainers & Collaborators

In addition to the guidelines mentioned above, collaborators with write access

to the repository should also follow these guidelines:

1. If there are new tests as part of the pull request, you should make sure that

they succeed.

1. When merging \*\*Pull Requests\*\* you should use the option \*Squash & Merge\* and

chose a descriptive commit message for the bugfix / feature (if not already

done by the individual contributor).

This way the history in the `master` branch will be free of small

corrections and easier to follow for people who aren’t engaged in the

project on a day-to-day basis.