# Contributing to Gitgraph.js

? 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.

Whether it's code, design, typo or documentation, every contribution is welcomed! So again, thank you very, very much ??

## ??? Using the issue tracker

The issue tracker is the preferred channel for [bug reports](#bugs), [features requests](#features) and [submitting pull requests](#pull-requests), but please \*\*do not\*\* derail or troll issues. Keep the discussion on topic and respect the opinions of others.

<a name="bugs"></a>

## ? 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` branch in the repository.

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

case](http://css-tricks.com/reduced-test-cases/) and a live example.

4. \*\*Create a new issue\*\* using the "Bug Report" template. Fill the different

sections to provide a maximum of details. They will help us to fix potential bugs.

5. [\*\*Use all-contributors bot command\*\*][all-contributors-bot-command] to add yourself to the list of contributors. It's very easy to do, you basically need to mention the bot in a comment of your PR.

<a name="features"></a>

## ?? 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.

If you have a clear feature in mind, please \*\*create a new issue\*\* using the "Feature Request"

template. Please provide as much detail and context as possible.

<a name="pull-requests"></a>

## ??? 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/<upstream-owner>/<repo-name>

```

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) to

contain your feature, change, or fix:

```bash

git checkout -b <topic-branch-name>

```

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.

8. [Use all-contributors bot command][all-contributors-bot-command] to add yourself to the list of contributors. It's very easy to do, you basically need to mention the bot in a comment of your PR.

\*\*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.

---

\_This was inspired from [@necolas issue guidelines](https://github.com/necolas/issue-guidelines/blob/master/CONTRIBUTING.md)\_