# Contribution Guide

If you have any comment or advice, please report your [issue](https://github.com/antvis/g2plot/issues),

or make any change as you wish and submit an [PR](https://github.com/antvis/g2plot/pulls).

## Reporting New Issues

- Please specify what kind of issue it is.

- Before you report an issue, please search for related issues. Make sure { you } are not going to open a duplicate issue.

- Explain your purpose clearly in tags(see \*\*Useful Tags\*\*), title, or content.

AntV group members will confirm the purpose of the issue, replace more accurate tags for it, identify related milestone, and { assign } developers working on it.

## Submitting Code

### Pull Request Guide

If you are developer of AntV repo and you are willing to contribute, feel free to create a new branch, finish your modification and submit a PR. AntV group will review your work and merge it to master branch.

```bash

# Create a new branch for development. The name of branch should be semantic, avoiding words like 'update' or 'tmp'. We suggest to use feature/xxx, if the modification is about to implement a new feature.

$ git checkout -b branch-name

# Run the test after you finish your modification. Add new test cases or change old ones if you feel necessary

$ npm test

# If your modification pass the tests, congradulations it's time to push your work back to us. Notice that the commit message should be wirtten in the following format.

$ git add . # git add -u to delete files

$ git commit -m "fix(role): role.use must xxx"

$ git push origin branch-name

```

Then you can create a Pull Request at [g2plot](https://github.com/antvis/g2plot/pulls).

No one can garantee how much will be remembered about certain PR after some time. To make sure we can easily recap what happened previously, please provide the following information in your PR.

1. Need: What function you want to achieve (Generally, please point out which issue is related).

2. Updating Reason: Different with issue. Briefly describe your reason and logic about why you need to make such modification.

3. Related Testing: Briefly descirbe what part of testing is relevant to your modification.

4. User Tips: Notice for g2plot users. You can { skip } this part, if the { PR } is not about update in API or potential compatibility problem.

### Style Guide

Eslint can help to identify styling issues that may exist in your code. Your code is required to pass the test from eslint. Run the test locally by `$ npm run lint`.

### Commit Message Format

You are encouraged to use [angular commit-message-format](https://github.com/angular/angular.js/blob/master/CONTRIBUTING.md#commit-message-format) to write commit message. In this way, we could have a more trackable history and an automatically generated changelog.

```xml

<type>(<scope>): <subject>

<BLANK LINE>

<body>

<BLANK LINE>

<footer>

```

?1?type

Must be one of the following:

- feat: A new feature

- fix: A bug fix

- docs: Documentation-only changes

- style: Changes that do { not } affect the meaning of the code (white-space, formatting, missing semi-colons, etc)

- refactor: A code change that neither fixes a bug nor adds a feature

- perf: A code change that improves performance

- test: Adding missing tests

- chore: Changes to the build process or auxiliary tools and libraries such as documentation generation

- deps: Updates about dependencies

?2?scope

The scope could be anything specifying place of the commit change.

?3?subject

Use succinct words to describe what did you do { in the } commit change.

?4?body

Feel free to add more content in the body, if you { think } subject is not self-explanatory enough, such as what it is the purpose or reasone of you commit.

?5?footer

- \*\*If the commit is a Breaking Change, please note it clearly in this part.\*\*

- related issues, like `Closes #1, Closes #2, #3`

e.g.

```

fix($compile): [BREAKING\_CHANGE] couple of unit tests for IE9

Older IEs serialize html uppercased, but IE9 does not...

Would be better to expect case insensitive, unfortunately jasmine does

not allow to user regexps for throw expectations.

Document change on antvis/g2plot#123

Closes #392

BREAKING CHANGE:

Breaks foo.bar api, foo.baz should be used instead

```

Look at [these files](https://docs.google.com/document/d/1QrDFcIiPjSLDn3EL15IJygNPiHORgU1\_OOAqWjiDU5Y/edit) for more detials.

## Release

g2plot uses semantic versioning in release process based on [semver].

### Branch Strategy

`master` branch is the latest stable version.

- just checkout develop branch from `master`

- All new features will be added into `master` or `next` branch as well as all bug-fix except security issues. In such way, we can motivate developers to update to the latest stable version.

### Release Strategy

In the release of every stable version, there will be a PM who has the following responsibilities in different stages of the release.

#### Preparation

- Set up milestone. Confirm that request is related to milestone.

#### Before Release

- Confirm that performance test is passed and all issues in current Milestone are either closed or can be delayed to later versions.

- Open a new [Release Proposal MR], and write `History` as [node CHANGELOG]. Don't forget to correct content in documentation which is related to the releasing version. Commits can be generated automatically.

```bash

$ npm run commits

```

- Nominate PM for next stable version.

[semver]: http://semver.org/lang/zh-CN/

[release proposal mr]: https://github.com/nodejs/node/pull/4181

[node changelog]: https://github.com/nodejs/node/blob/master/CHANGELOG.md

[????????? npm ???]: https://fengmk2.com/blog/2016/how-i-publish-a-npm-package