# Contributing to Liquid Prompt

### I don't want to read this whole thing, I just have a question!!!

Please don't file an issue to ask a question. You'll get faster results by

using the resources below.

## Did you find a bug?

\* \*\*Check the [FAQs](https://github.com/nojhan/liquidprompt/wiki/FAQ)\*\*. We

try to keep it updated with the latest hot questions.

\* \*\*Ensure the bug was not already reported\*\* by searching on GitHub under

[Issues](https://github.com/nojhan/liquidprompt/issues). If there is an open

issue that matches your problem, you can add a comment to the issue if you

have something new or helpful to add. If your comment would be ":+1:",

please add that as a reaction on the issue instead.

\* \*\*Ensure you can reproduce the issue on the latest version\*\*. Not only the

latest released version, but the latest commit on the `master` branch. While

it might not be the version you are using right now, if the bug doesn't exist

on the `master` branch, we have already fixed it.

\* If you're unable to find an open issue addressing the problem, [open a new

one](https://github.com/nojhan/liquidprompt/issues/new/choose). Be sure to

include a \*\*title and clear description\*\*, as much relevant information as

possible, such as shell version, and a \*\*code sample\*\* or a \*\*test case\*\*

demonstrating the expected behavior that is not occurring. If we can't

reproduce the bug, we can't fix it, so make sure to include as much

information as possible.

## Enhancement / Feature request

Same as with bugs, \*\*check the [FAQs](https://github.com/nojhan/liquidprompt/wiki/FAQ)

and open [Issues](https://github.com/nojhan/liquidprompt/issues)\*\* so you don't

duplicate a feature request.

[Open a new Issue](https://github.com/nojhan/liquidprompt/issues/new/choose),

and add as much detail as you can. Make sure to at least explain:

1. \*\*Why\*\* you want this feature. How will it be useful to users?

2. \*\*What\*\* tools it will interact with (ex: `git`). Provide links.

3. \*\*An example\*\* of what it would look like in the prompt.

The more details you can give, the more likely someone will be inspired to work

on it.

## Contributing a patch

### Your first contribution / what can I help with?

If you just want to help, but don't have a specific issue in mind, you can

look at the [help wanted](https://github.com/nojhan/liquidprompt/labels/help%20wanted)

tagged issues.

### Code style and standards

[Google's shell style guide](https://google.github.io/styleguide/shellguide.html)

is our style guide, with the following modifications:

\* 4 spaces indentation

\* Don't always prefer `${var}` over `$var` (see standards)

\* Use `typeset`, not `local` or `declare` (see standards)

See our [shell standards wiki page](https://github.com/nojhan/liquidprompt/wiki/Shell-standards)

for our standards on shell language.

### How do I make a pull request??

$ git clone -o upstream git://github.com/nojhan/liquidprompt.git

$ cd liquidprompt

# Run liquidprompt and check that your issue is still on that branch

$ source liquidprompt

# Prepare a fix (include the issue number in the branch name if an issue

# already exists)

$ git checkout -b bugfix/my-fix

# Prepare a new feature

$ git checkout -b feature/my-feature

# Hack, commit, hack, amend commit...

# Fork the project on GitHub (if you haven't yet)

# Add the remote target for pushes

$ git remote add github git@github.com:$GITHUB\_USER/liquidprompt.git

# Check that your local repo is up to date

$ git fetch upstream

# Rebase your work on the latest state of `master`

$ git rebase upstream/master

# Push your commits

$ git push github fix/my-fix

$ git push github feature/my-feature

# Create the pull request on GitHub. Check that Github chose the `master`

# branch as the starting point for your branch.

### How do I make a good pull request?

1. Check that your Git authorship settings are correct:

$ git config -l | grep ^user\.

2. All the commits in the pull request must be on the same topic. If instead

you propose fixes on different topics, use separate branches in your repo

and make a pull request for each.

3. Good commit message:

- first line must be 50 chars max and is a summary of the commit. The

first charcter should be capitalized, no ending period. Use the

imperative mood ("fix thing", not "fixes thing" or "fixing thing" or

"fixed thing")

- second line must be empty

- following lines (72 chars max) should describe the what, why, and how

of your change. If your change is anything more than one line, this is

probably not optional. Think about coming back to this patch in 6

months: if you won't remember why you made this change, you need to

write it down.

Use references to GitHub issues number (ex: `#432`) if applicable

A good commit message isn't optional. If your commit message is bad (ex:

"fix temp for thing"), your PR will be rejected.

[See this post](https://chris.beams.io/posts/git-commit/) for even more

details on the topic.

4. Use a good title for your pull request.

5. Put details, web links, Github issue numbers, in the pull request body. Use

Markdown fully to format the content (see

[Markdown syntax](https://guides.github.com/features/mastering-markdown/)).

For example use triple backquotes for code blocks.

Note that if information is important enough to put in the PR description,

it is also important enough to put in the commit message.

If your branch ever ends up "out of sync" or "conflicts" with the `master`

branch of the main repo, never, ever, merge the `master` branch into your own

branch. Instead, always rebase your own work on top of the `master` branch.

### Can I make a pull request without a separate issue for a bug/enhancement?

Yes, \*\*but\*\*, in that case, the pull request \*\*must have a full description of

the bug or feature\*\*. Just because you have fixed/implemented it already

doesn't mean you can skip the details. The reviewers of the pull request still

need to know all of the information you would normally put in the bug report or

feature request.

### How my patch will be applied?

Before being applied, your pull request will be reviewed by the maintainer

and also by other users. You can also help the project by reviewing others

pull requests.

If your patch is accepted it will be applied either:

- by "merging" your branch

- by cherry-picking your commit on top of the `master` branch. This makes the

history linear, and so easier to track.

In any case, your authorship will be preserved in the commit.

### What if my patch is not applied?

If you don't even get a review, add a "ping" comment with increasing delay

between pings: 1 week, 2 weeks, then every month. But I'm trying to do better

on this than what was happening before.

If a stable version is released while your pull request has still not been

merged on any working branch of the main repo, it would be helpful to ease

the maitainer's work by rebasing your branch on top of the latest `master`

and push it again to your GitHub repo. Be careful (for example create a

branch or a tag before your rebase) because your may lose all your work in

that process.

[@Rycieos](https://github.com/Rycieos), maintainer.