Writing issue reports

=====================

### Bugs and features only

First things first: \*\*the issue tracker is NOT for tech support\*\*. It is for

reporting bugs and requesting features. If your issue amounts to "I can't get

ycmd to work on my machine" and the reason why is obviously related to your

machine configuration and the problem would not be resolved with \_reasonable\_

changes to the ycmd codebase, then the issue is likely to be closed.

### Where to go for help

\*\*A good place to ask questions is the [ycmd-users][] Google group\*\*. Rule

of thumb: if you're not sure whether your problem is a real bug, ask on

the group.

### Installation problem - read the docs

\*\*ycmd compiles just fine\*\*; [the build bots say so][build-bots]. If the bots are

green and ycmd doesn't compile on your machine, then \_your machine is the root

cause\_. Now read the first paragraph again.

Realize that quite literally \_thousands\_ of people have gotten ycmd to work

successfully so if you can't, it's probably because you have a peculiar

system or configuration or you didn't go through the docs carefully enough.

It's very unlikely to be caused by an actual bug in ycmd because someone would

have already found it and reported it.

This leads us to point #2: \*\*make sure you have checked the docs before

reporting an issue\*\*. The docs are extensive and cover a ton of things; there's

also an FAQ at the bottom that quite possibly addresses your problem. NOTE:

installation of ycmd is predominantly documented in the YouCompleteMe (Vim client)

README.

For installation problems, make sure that any issue report includes the entire

output of any build or installation commands, including \*\*the command used to

run them\*\*.

### Other problems - check the issue tracker

Further, \*\*search the issue tracker for similar issues\*\* before creating a new

one. There's no point in duplication; if an existing issue addresses your

problem, please comment there instead of creating a duplicate. However, if the

issue you found is \*\*closed as resolved\*\* (e.g. with a PR or the original user's

problem was resolved), raise a \*\*new issue\*\*, because you've found a new

problem. Reference the original issue if you think that's useful information.

If you do find a similar open issue, \*\*don't just post 'me too' or similar\*\*

responses. This almost never helps resolve the issue, and just causes noise for

the maintainers. Only post if it will aid the maintainers in solving the issue;

if there are existing diagnostics requested in the thread, perform

them and post the results.

When replying, follow the instructions for getting the required diagnostics for

posting a new issue (see below), and add them to your response. This is likely

to help the maintainers find a fix for you, rather than have them spend time

requesting them again. To be clear, the maintainers \*always\* need the

diagnostics (debug info, log files, versions, etc.) even for responses on

existing issues.

You should also \*\*search the archives of the [ycmd-users][] mailing list\*\*.

### Check your ycmd version

Lastly, \*\*make sure you are running the latest version of ycmd\*\*. The issue you

have encountered may have already been fixed. \*\*Don't forget to recompile

ycm\_core.so too\*\* (usually by just running `build.py` again).

## Creating an issue

OK, so we've reached this far. You need to create an issue. First realize that

the time it takes to fix your issue is a multiple of how long it takes the

developer to reproduce it. The easier it is to reproduce, the quicker it'll be

fixed.

Here are the things you should do when creating an issue:

1. \*\*Write a step-by-step procedure that when performed repeatedly reproduces

your issue.\*\* If we can't reproduce the issue, then we can't fix it. It's

that simple. The procedure can be written the Vim client or the API

directly. If you're using another client, we can try to help, but it's

easier if we can reproduce in the Vim client. Usually it's best to report

such issues to the client maintainer, who can pass you on to ycmd if that

looks like the cause.

2. Explain \*\*what you expected to happen\*\*, and \*\*what actually happened\*\*.

This helps us understand if it is a bug, or just a misunderstanding of the

behavior.

2. Add the output of the `/debug\_info` request for the file that is

experiencing the issue.

3. Ensure that the `server\_log\_level` option is set to `debug`, then reproduce

your issue and attach the contents of the logfiles reported in the debug

info.

4. \*\*Create a test case for your issue\*\*. This is critical. Don't talk about how

"when I have X in my file" or similar, \_create a file with X in it\_ and put

the contents inside code blocks in your issue description. Try to make this

test file \_as small as possible\_. Don't just paste a huge, 500 line source

file you were editing and present that as a test. \_Minimize\_ the file so that

the problem is reproduced with the smallest possible amount of test data.

5. \*\*Include your OS and OS version.\*\*

Creating good pull requests

===========================

1. \*\*Follow the code style of the existing codebase.\*\*

- The Python code \*\*DOES NOT\*\* follow PEP 8. This is not an oversight, this

is by choice. You can dislike this as much as you want, but you still need

to follow the existing style. Look at other Python files to see what the

style is.

- The C++ code has an automated formatter (`style\_format.sh` that runs

`astyle`) but it's not perfect. Again, look at the other C++ files and

match the code style you see.

2. \*\*Your code needs to be well written and easy to maintain\*\*. This is of the

\_utmost\_ importance. Other people will have to maintain your code so don't

just throw stuff against the wall until things kinda work.

3. \*\*Split your pull request into several smaller ones if possible.\*\* This

makes it easier to review your changes, which means they will be merged

faster.

4. \*\*Write tests for your code\*\*. Your pull request is unlikely to be merged

without tests. See [TESTS.md][ycmd-tests] for instructions on running the

tests.

5. \*\*Explain in detail why your pull request makes sense.\*\* Ask yourself, would

this feature be helpful to others? Not just a few people, but a lot of

ycmd's users? See, good features are useful to many. If your feature is only

useful to you and \_maybe\_ a couple of others, then that's not a good

feature. There is such a thing as “feature overload”. When software

accumulates so many features of which most are only useful to a handful,

then that software has become “bloated”. We don't want that.

Requests for features that are obscure or are helpful to but a few, or are

not part of ycmd's "vision" will be rejected. Yes, even if you provide a

patch that completely implements it.

Please include details on exactly what you would like to see, and why. The

why is important - it's not always clear why a feature is really useful. And

sometimes what you want can be done in a different way if the reason for the

change is known. \_What goal is your change trying to accomplish?\_

[build-bots]: https://dev.azure.com/YouCompleteMe/YCM/\_build/latest?definitionId=2&branchName=master

[ycmd-users]: https://groups.google.com/forum/?hl=en#!forum/ycmd-users

[ycmd-tests]: https://github.com/ycm-core/ycmd/blob/master/TESTS.md