# Contributing to Jasper

Want to contribute to Jasper? Great! We're always happy to have more contributors. Before you start developing, though, we ask that you read through this document in-full. It's full of tips and guidelines--if you skip it, you'll likely miss something important (and your pull request will probably be rejected as a result).

Throughout the process of contributing, there's one thing we'd like you to remember: Jasper was developed (and is maintained) by what might be described as "volunteers". They earn no money for their work on Jasper and give their time solely for the advancement of the software and the enjoyment of its users. While they will do their best to get back to you regarding issues and pull requests, \*\*your patience is appreciated\*\*.

## Reporting Bugs

The [bug tracker](https://github.com/jasperproject/jasper-client/issues) at Github is for reporting bugs in Jasper. If you encounter problems during installation or compilation of one of Jasper's dependencies for example, do not create a new issue here. Please open a new thread in the [support forum](https://groups.google.com/forum/#!forum/jasper-support-forum) instead. Also, make sure that it's not a usage issue.

If you think that you found a bug and that you're using the most recent version of Jasper, please include a detailed description what you did and how to reproduce the bug. If Jasper crashes, run it with `--debug` as command line argument and also include the full stacktrace (not just the last line). If you post output, put it into a [fenced code block](https://help.github.com/articles/github-flavored-markdown/#fenced-code-blocks). Last but not least: have a look at [Simon Tatham's "How to Report Bugs Effectively"](http://www.chiark.greenend.org.uk/~sgtatham/bugs.html) to learn how to write a good bug report.

## Opening Pull Requests

### Philosophies

There are a few key philosophies to preserve while designing features for Jasper:

1. \*\*The core Jasper software (`jasper-client`) must remain decoupled from any third-party web services.\*\* For example, the Jasper core should never depend on Google Translate in any way. This is to avoid unnecessary dependences on web services that might change or become paid over time.

2. \*\*The core Jasper software (`jasper-client`) must remain decoupled from any paid software or services.\*\* Of course, you're free to use whatever you'd like when running Jasper locally or in a fork, but the main branch needs to remain free and open-source.

3. \*\*Jasper should be \_usable\_ by both beginner and expert programmers.\*\* If you make a radical change, in particular one that requires some sort of setup, try to offer an easy-to-run alternative or tutorial. See, for example, the profile populator ([`jasper-client/client/populate.py`](https://github.com/jasperproject/jasper-client/blob/master/client/populate.py)), which abstracts away the difficulty of correctly formatting and populating the user profile.

### DOs and DON'Ts

While developing, you \*\*\_should\_\*\*:

1. \*\*Ensure that the existing unit tests pass.\*\* They can be run via `python2 -m unittest discover` for Jasper's main folder.

2. \*\*Test \_every commit\_ on a Raspberry Pi\*\*. Testing locally (i.e., on OS X or Windows or whatnot) is insufficient, as you'll often run into semi-unpredictable issues when you port over to the Pi. You should both run the unit tests described above and do some anecdotal testing (i.e., run Jasper, trigger at least one module).

3. \*\*Ensure that your code conforms to [PEP8](http://legacy.python.org/dev/peps/pep-0008/) and our existing code standards.\*\* For example, we used camel case in a few places (this could be changed--send in a pull request!). In general, however, defer to PEP8. We also really like Jeff Knupp's [\_Writing Idiomatic Python\_](http://www.jeffknupp.com/writing-idiomatic-python-ebook/). We use `flake8` to check this, so run it from Jasper's main folder before committing.

4. Related to the above: \*\*Include docstrings that follow our existing format!\*\* Good documentation is a good thing.

4. \*\*Add any new Python dependencies to requirements.txt.\*\* Make sure that your additional dependencies are dependencies of `jasper-client` and not existing packages on your disk image!

5. \*\*Explain \_why\_ your change is necessary.\*\* What does it accomplish? Is this something that others will want as well?

6. Once your pull request has received some positive feedback: \*\*Submit a parallel pull request to the [documentation repository](https://github.com/jasperproject/jasperproject.github.io)\*\* to keep the docs in sync.

On the other hand, you \*\*\_should not\_\*\*:

1. \*\*Commit \_any\_ modules to the \_jasper-client\_ repository.\*\* The modules included in \_jasper-client\_ are meant as illustrative examples. Any new modules that you'd like to share should be done so through other means. If you'd like us to [list your module](http://jasperproject.github.io/documentation/modules/) on the web site, [submit a pull request here](https://github.com/jasperproject/jasperproject.github.io/blob/master/documentation/modules/index.md).

2. \*\*\_Not\_ do any of the DOs!\*\*

### TODOs

If you're looking for something to do, here are some suggestions:

1. Improve unit-testing for `jasper-client`. The Jasper modules and `brain.py` have ample testing, but other Python modules (`conversation.py`, `mic.py`, etc.) do not.

2. Come up with a better way to automate testing on the Pi. This might include spinning up some sort of VM with [Docker](http://docs.docker.io), or take a completely different approach.

3. Buff up the text-refinement functions in [`alteration.py`](https://github.com/jasperproject/jasper-client/blob/master/client/alteration.py). These are meant to convert text to a form that will sound more human-friendly when spoken by your TTS software, but are quite minimal at the moment.

4. Make Jasper more platform-independent. Currently, Jasper is only supported on Raspberry Pi and OS X.

5. Decouple Jasper from any specific Linux audio driver implementation.