# Contributing to NLTK

Hi! Thanks for your interest in contributing to [NLTK](http://www.nltk.org/).

:-) You'll be joining a [long list of contributors](https://github.com/nltk/nltk/blob/develop/AUTHORS.md).

In this document we'll try to summarize everything that you need to know to

do a good job.

## Code and Issues

We use [GitHub](https://www.github.com/) to host our code repositories and

issues. The [NLTK organization on GitHub](https://github.com/nltk) has many

repositories, so we can manage better the issues and development. The most

important are:

- [nltk/nltk](https://github.com/nltk/nltk/), the main repository with code

related to the library;

- [nltk/nltk\_data](https://github.com/nltk/nltk\_data), repository with data

related to corpora, taggers and other useful data that are not shipped by

default with the library, which can be downloaded by `nltk.downloader`;

- [nltk/nltk.github.com](https://github.com/nltk/nltk.github.com), NLTK website

with information about the library, documentation, link for downloading NLTK

Book etc.;

- [nltk/nltk\_book](https://github.com/nltk/nltk\_book), source code for the NLTK

Book.

## Development priorities

NLTK consists of the functionality that the Python/NLP community is motivated to contribute.

Some priority areas for development are listed in the [NLTK Wiki](https://github.com/nltk/nltk/wiki#development)

## Git and our Branching model

### Git

We use [Git](http://git-scm.com/) as our [version control

system](http://en.wikipedia.org/wiki/Revision\_control), so the best way to

contribute is to learn how to use it and put your changes on a Git repository.

There's a plenty of documentation about Git -- you can start with the [Pro Git

book](http://git-scm.com/book/).

### Setting up a Development Environment

To set up your local development environment for contributing to the main

repository [nltk/nltk](https://github.com/nltk/nltk/):

- Fork the [nltk/nltk](https://github.com/nltk/nltk/) repository on GitHub

to your account;

- Clone your forked repository locally

(`git clone https://github.com/<your-github-username>/nltk.git`);

- Run `cd nltk` to get to the root directory of the `nltk` code base;

- Install the dependencies (`pip install -r pip-req.txt`);

- Download the datasets for running tests

(`python -m nltk.downloader all`);

- Create a remote link from your local repository to the

upstream `nltk/nltk` on GitHub

(`git remote add upstream https://github.com/nltk/nltk.git`) --

you will need to use this `upstream` link when updating your local repository

with all the latest contributions.

### GitHub Pull requests

We use the famous

[gitflow](http://nvie.com/posts/a-successful-git-branching-model/) to manage our

branches.

Summary of our git branching model:

- Go to the `develop` branch (`git checkout develop`);

- Get all the latest work from the upstream `nltk/nltk` repository

(`git pull upstream develop`);

- Create a new branch off of `develop` with a descriptive name (for example:

`feature/portuguese-sentiment-analysis`, `hotfix/bug-on-downloader`). You can

do it switching to `develop` branch (`git checkout develop`) and then

creating a new branch (`git checkout -b name-of-the-new-branch`);

- Do many small commits on that branch locally (`git add files-changed`,

`git commit -m "Add some change"`);

- Run the tests to make sure nothing breaks

(`tox -e py36` if you are on Python 3.6);

- Add your name to the `AUTHORS.md` file as a contributor;

- Push to your fork on GitHub (with the name as your local branch:

`git push origin branch-name`);

- Create a pull request using the GitHub Web interface (asking us to pull the

changes from your new branch and add to our `develop` branch);

- Wait for comments.

### Tips

- Write [helpful commit

messages](http://robots.thoughtbot.com/5-useful-tips-for-a-better-commit-message).

- Anything in the `develop` branch should be deployable (no failing tests).

- Never use `git add .`: it can add unwanted files;

- Avoid using `git commit -a` unless you know what you're doing;

- Check every change with `git diff` before adding them to the index (stage

area) and with `git diff --cached` before commiting;

- Make sure you add your name to our [list of contributors](https://github.com/nltk/nltk/blob/develop/AUTHORS.md);

- If you have push access to the main repository, please do not commit directly

to `develop`: your access should be used only to accept pull requests; if you

want to make a new feature, you should use the same process as other

developers so you code will be reviewed.

- See [RELEASE-HOWTO.txt](RELEASE-HOWTO.txt) to see everything you

need before creating a new NLTK release.

## Code Guidelines

- Use [PEP8](http://www.python.org/dev/peps/pep-0008/);

- Write tests for your new features (please see "Tests" topic below);

- Always remember that [commented code is dead

code](http://www.codinghorror.com/blog/2008/07/coding-without-comments.html);

- Name identifiers (variables, classes, functions, module names) with readable

names (`x` is always wrong);

- When manipulating strings, use [Python's new-style

formatting](http://docs.python.org/library/string.html#format-string-syntax)

(`'{} = {}'.format(a, b)` instead of `'%s = %s' % (a, b)`);

- All `#TODO` comments should be turned into issues (use our

[GitHub issue system](https://github.com/nltk/nltk/issues));

- Run all tests before pushing (just execute `tox`) so you will know if your

changes broke something;

See also our [developer's

guide](https://github.com/nltk/nltk/wiki/Developers-Guide).

## Tests

You should write tests for every feature you add or bug you solve in the code.

Having automated tests for every line of our code let us make big changes

without worries: there will always be tests to verify if the changes introduced

bugs or lack of features. If we don't have tests we will be blind and every

change will come with some fear of possibly breaking something.

For a better design of your code, we recommend using a technique called

[test-driven development](https://en.wikipedia.org/wiki/Test-driven\_development),

where you write your tests \*\*before\*\* writing the actual code that implements

the desired feature.

## Continuous Integration

\*\*Deprecated:\*\* NLTK uses [Cloudbees](https://nltk.ci.cloudbees.com/) for continuous integration.

NLTK uses [Travis](https://travis-ci.org/nltk/nltk/) for continuous integration.

The [`.travis.yml`](https://github.com/nltk/nltk/blob/travis/.travis.yml) file configures the server:

- `matrix: include:` section

- tests against supported Python versions (3.5, 3.6, 3.7, 3.8)

- all python versions run the `py-travis` tox test environment in the [`tox.ini`](https://github.com/nltk/nltk/blob/travis/tox.ini#L105) file

- tests against Python 3.6 for third-party tools APIs

- `before\_install:` section

- checks the Java and Python version calling the `tools/travis/pre-install.sh` script

- changes the permission for `tools/travis/coverage-pylint.sh` to allow it to be executable

- changes the permission for `tools/travis/third-party.sh` to allow it to be executable

- `install` section

- the `tools/travis/install.sh` installs the `pip-req.txt` for NLTK and the necessary python packages for CI testing

- install `tox` for testing

- `py-travis` tox test environment generally

- the `extras = all` dependencies in needed to emulate `pip install nltk[all]`, see https://tox.readthedocs.io/en/latest/config.html#confval-extras=MULTI-LINE-LIST

- for the `py-travis-third-party` build, it will run `tools/travis/third-party.sh` to install third-party tools (Stanford NLP tools and CoreNLP and SENNA)

- calls `tools/travis/coverage-pylint.sh` shell script that calls the `nltk/nltk/test/runtests.py` with [`coverage`](https://pypi.org/project/coverage/) and

- calls `pylint` # Currently, disabled because there's lots to clean...

- before returning a `true` to state that the build is successful

#### To test with `tox` locally

First setup a new virtual environment, see https://docs.python-guide.org/dev/virtualenvs/

Then run `tox -e py37`.

For example, using `pipenv`:

```

git clone https://github.com/nltk/nltk.git

cd nltk

pipenv install -r pip-req.txt

pipenv install tox

tox -e py37

```

# Discussion

We have three mail lists on Google Groups:

- [nltk][nltk-announce], for announcements only;

- [nltk-users][nltk-users], for general discussion and user questions;

- [nltk-dev][nltk-dev], for people interested in NLTK development.

Please feel free to contact us through the [nltk-dev][nltk-dev] mail list if

you have any questions or suggestions. Every contribution is very welcome!

Happy hacking! (;

[nltk-announce]: https://groups.google.com/forum/#!forum/nltk

[nltk-dev]: https://groups.google.com/forum/#!forum/nltk-dev

[nltk-users]: https://groups.google.com/forum/#!forum/nltk-users