# Contributing code to Synapse

Everyone is welcome to contribute code to [matrix.org

projects](https://github.com/matrix-org), provided that they are willing to

license their contributions under the same license as the project itself. We

follow a simple 'inbound=outbound' model for contributions: the act of

submitting an 'inbound' contribution means that the contributor agrees to

license the code under the same terms as the project's overall 'outbound'

license - in our case, this is almost always Apache Software License v2 (see

[LICENSE](LICENSE)).

## How to contribute

The preferred and easiest way to contribute changes is to fork the relevant

project on github, and then [create a pull request](

https://help.github.com/articles/using-pull-requests/) to ask us to pull your

changes into our repo.

Some other points to follow:

\* Please base your changes on the `develop` branch.

\* Please follow the [code style requirements](#code-style).

\* Please include a [changelog entry](#changelog) with each PR.

\* Please [sign off](#sign-off) your contribution.

\* Please keep an eye on the pull request for feedback from the [continuous

integration system](#continuous-integration-and-testing) and try to fix any

errors that come up.

\* If you need to [update your PR](#updating-your-pull-request), just add new

commits to your branch rather than rebasing.

## Code style

Synapse's code style is documented [here](docs/code\_style.md). Please follow

it, including the conventions for the [sample configuration

file](docs/code\_style.md#configuration-file-format).

Many of the conventions are enforced by scripts which are run as part of the

[continuous integration system](#continuous-integration-and-testing). To help

check if you have followed the code style, you can run `scripts-dev/lint.sh`

locally. You'll need python 3.6 or later, and to install a number of tools:

```

# Install the dependencies

pip install -U black flake8 flake8-comprehensions isort

# Run the linter script

./scripts-dev/lint.sh

```

\*\*Note that the script does not just test/check, but also reformats code, so you

may wish to ensure any new code is committed first\*\*.

By default, this script checks all files and can take some time; if you alter

only certain files, you might wish to specify paths as arguments to reduce the

run-time:

```

./scripts-dev/lint.sh path/to/file1.py path/to/file2.py path/to/folder

```

Before pushing new changes, ensure they don't produce linting errors. Commit any

files that were corrected.

Please ensure your changes match the cosmetic style of the existing project,

and \*\*never\*\* mix cosmetic and functional changes in the same commit, as it

makes it horribly hard to review otherwise.

## Changelog

All changes, even minor ones, need a corresponding changelog / newsfragment

entry. These are managed by [Towncrier](https://github.com/hawkowl/towncrier).

To create a changelog entry, make a new file in the `changelog.d` directory named

in the format of `PRnumber.type`. The type can be one of the following:

\* `feature`

\* `bugfix`

\* `docker` (for updates to the Docker image)

\* `doc` (for updates to the documentation)

\* `removal` (also used for deprecations)

\* `misc` (for internal-only changes)

This file will become part of our [changelog](

https://github.com/matrix-org/synapse/blob/master/CHANGES.md) at the next

release, so the content of the file should be a short description of your

change in the same style as the rest of the changelog. The file can contain Markdown

formatting, and should end with a full stop (.) or an exclamation mark (!) for

consistency.

Adding credits to the changelog is encouraged, we value your

contributions and would like to have you shouted out in the release notes!

For example, a fix in PR #1234 would have its changelog entry in

`changelog.d/1234.bugfix`, and contain content like:

> The security levels of Florbs are now validated when received

> via the `/federation/florb` endpoint. Contributed by Jane Matrix.

If there are multiple pull requests involved in a single bugfix/feature/etc,

then the content for each `changelog.d` file should be the same. Towncrier will

merge the matching files together into a single changelog entry when we come to

release.

### How do I know what to call the changelog file before I create the PR?

Obviously, you don't know if you should call your newsfile

`1234.bugfix` or `5678.bugfix` until you create the PR, which leads to a

chicken-and-egg problem.

There are two options for solving this:

1. Open the PR without a changelog file, see what number you got, and \*then\*

add the changelog file to your branch (see [Updating your pull

request](#updating-your-pull-request)), or:

1. Look at the [list of all

issues/PRs](https://github.com/matrix-org/synapse/issues?q=), add one to the

highest number you see, and quickly open the PR before somebody else claims

your number.

[This

script](https://github.com/richvdh/scripts/blob/master/next\_github\_number.sh)

might be helpful if you find yourself doing this a lot.

Sorry, we know it's a bit fiddly, but it's \*really\* helpful for us when we come

to put together a release!

### Debian changelog

Changes which affect the debian packaging files (in `debian`) are an

exception to the rule that all changes require a `changelog.d` file.

In this case, you will need to add an entry to the debian changelog for the

next release. For this, run the following command:

```

dch

```

This will make up a new version number (if there isn't already an unreleased

version in flight), and open an editor where you can add a new changelog entry.

(Our release process will ensure that the version number and maintainer name is

corrected for the release.)

If your change affects both the debian packaging \*and\* files outside the debian

directory, you will need both a regular newsfragment \*and\* an entry in the

debian changelog. (Though typically such changes should be submitted as two

separate pull requests.)

## Sign off

In order to have a concrete record that your contribution is intentional

and you agree to license it under the same terms as the project's license, we've adopted the

same lightweight approach that the Linux Kernel

[submitting patches process](

https://www.kernel.org/doc/html/latest/process/submitting-patches.html#sign-your-work-the-developer-s-certificate-of-origin>),

[Docker](https://github.com/docker/docker/blob/master/CONTRIBUTING.md), and many other

projects use: the DCO (Developer Certificate of Origin:

http://developercertificate.org/). This is a simple declaration that you wrote

the contribution or otherwise have the right to contribute it to Matrix:

```

Developer Certificate of Origin

Version 1.1

Copyright (C) 2004, 2006 The Linux Foundation and its contributors.

660 York Street, Suite 102,

San Francisco, CA 94110 USA

Everyone is permitted to copy and distribute verbatim copies of this

license document, but changing it is not allowed.

Developer's Certificate of Origin 1.1

By making a contribution to this project, I certify that:

(a) The contribution was created in whole or in part by me and I

have the right to submit it under the open source license

indicated in the file; or

(b) The contribution is based upon previous work that, to the best

of my knowledge, is covered under an appropriate open source

license and I have the right under that license to submit that

work with modifications, whether created in whole or in part

by me, under the same open source license (unless I am

permitted to submit under a different license), as indicated

in the file; or

(c) The contribution was provided directly to me by some other

person who certified (a), (b) or (c) and I have not modified

it.

(d) I understand and agree that this project and the contribution

are public and that a record of the contribution (including all

personal information I submit with it, including my sign-off) is

maintained indefinitely and may be redistributed consistent with

this project or the open source license(s) involved.

```

If you agree to this for your contribution, then all that's needed is to

include the line in your commit or pull request comment:

```

Signed-off-by: Your Name <your@email.example.org>

```

We accept contributions under a legally identifiable name, such as

your name on government documentation or common-law names (names

claimed by legitimate usage or repute). Unfortunately, we cannot

accept anonymous contributions at this time.

Git allows you to add this signoff automatically when using the `-s`

flag to `git commit`, which uses the name and email set in your

`user.name` and `user.email` git configs.

## Continuous integration and testing

[Buildkite](https://buildkite.com/matrix-dot-org/synapse) will automatically

run a series of checks and tests against any PR which is opened against the

project; if your change breaks the build, this will be shown in GitHub, with

links to the build results. If your build fails, please try to fix the errors

and update your branch.

To run unit tests in a local development environment, you can use:

- ``tox -e py35`` (requires tox to be installed by ``pip install tox``)

for SQLite-backed Synapse on Python 3.5.

- ``tox -e py36`` for SQLite-backed Synapse on Python 3.6.

- ``tox -e py36-postgres`` for PostgreSQL-backed Synapse on Python 3.6

(requires a running local PostgreSQL with access to create databases).

- ``./test\_postgresql.sh`` for PostgreSQL-backed Synapse on Python 3.5

(requires Docker). Entirely self-contained, recommended if you don't want to

set up PostgreSQL yourself.

Docker images are available for running the integration tests (SyTest) locally,

see the [documentation in the SyTest repo](

https://github.com/matrix-org/sytest/blob/develop/docker/README.md) for more

information.

## Updating your pull request

If you decide to make changes to your pull request - perhaps to address issues

raised in a review, or to fix problems highlighted by [continuous

integration](#continuous-integration-and-testing) - just add new commits to your

branch, and push to GitHub. The pull request will automatically be updated.

Please \*\*avoid\*\* rebasing your branch, especially once the PR has been

reviewed: doing so makes it very difficult for a reviewer to see what has

changed since a previous review.

## Notes for maintainers on merging PRs etc

There are some notes for those with commit access to the project on how we

manage git [here](docs/dev/git.md).

## Conclusion

That's it! Matrix is a very open and collaborative project as you might expect

given our obsession with open communication. If we're going to successfully

matrix together all the fragmented communication technologies out there we are

reliant on contributions and collaboration from the community to do so. So

please get involved - and we hope you have as much fun hacking on Matrix as we

do!