# User support

The [Joplin Forum](https://discourse.joplinapp.org/) is the community driven place for user support, general discussion about Joplin, problems with installation, new features and software development questions. It is possible to login with your GitHub account. Don't use the issue tracker for support questions.

# Reporting a bug

File bugs in the [Github Issue Tracker](https://github.com/laurent22/joplin/issues?utf8=%E2%9C%93&q=is%3Aissue). Please follow these guidelines:

- Search existing issues first, make sure yours hasn't already been reported.

- Please follow the template.

- Consider [enabling debug mode](https://joplinapp.org/debugging/) so that you can provide as much details as possible when reporting the issue.

- Stay on topic, but describe the issue in detail so that others can \*\*reproduce\*\* it.

- \*\*Provide a screenshot\*\* if possible. A screenshot showing the problem is often more useful than a paragraph describing it.

- For web clipper bugs, \*\*please provide the URL causing the issue\*\*. Sometimes the clipper works in one page but not in another so it is important to know what URL has a problem.

# Feature requests

Feature requests \*\*must be opened and discussed on the [forum](https://discourse.joplinapp.org/c/features)\*\*. After they have been accepted, they can be added to the GitHub tracker.

Please check that your request has not already been posted on the forum or the [Github Issue Tracker](https://github.com/laurent22/joplin/issues?utf8=%E2%9C%93&q=is%3Aissue). If it has, \*\*up-voting the issue or topic\*\* increases the chances it'll be noticed and implemented in the future. "+1" comments are not tracked.

Avoid listing multiple requests in one topic. One topic per request makes it easier to track and discuss it.

Finally, when submitting a pull request, don't forget to [test your code](#automated-tests).

# Contributing to Joplin's translation

Joplin is available in multiple languages thanks to the help of its users. You can help translate Joplin to your language or keep it up to date. Please read the documentation about [Localisation](https://joplinapp.org/#localisation).

# Contributing to Joplin's code

If you want to start contributing to the project's code, please follow these guidelines before creating a pull request:

- Bug fixes are always welcome. Start by reviewing the [list of bugs](https://github.com/laurent22/joplin/issues?q=is%3Aissue+is%3Aopen+label%3Abug)

- A good way to easily start contributing is to pick and work on a [good first issue](https://github.com/laurent22/joplin/issues?q=is%3Aissue+is%3Aopen+label%3A%22good+first+issue%22). We try to make these issues as clear as possible and provide basic info on how the code should be changed, and if something is unclear feel free to ask for more information on the issue.

- Before adding a new feature, ask about it in the [Github Issue Tracker](https://github.com/laurent22/joplin/issues?utf8=%E2%9C%93&q=is%3Aissue) or the [Joplin Forum](https://discourse.joplinapp.org/), or check if existing discussions exist to make sure the new functionality is desired.

- \*\*Changes that will consist in more than 50 lines of code should be discussed the [Joplin Forum](https://discourse.joplinapp.org/)\*\*, so that you don't spend too much time implementing something that might not be accepted.

- All the applications share the same backend (database, synchronisation, settings, models, business logic, etc.) so if you change something in the backend in one app, makes sure it still work in the other apps. Usually it does, but keep this in mind.

Building the apps is relatively easy - please [see the build instructions](https://github.com/laurent22/joplin/blob/master/BUILD.md) for more details.

## Coding style

Coding style is enforced by a pre-commit hook that runs eslint. This hook is installed whenever running `npm install` on any of the application directory. If for some reason the pre-commit hook didn't get installed, you can manually install it by running `npm install` at the root of the repository.

For new React components, please use [React Hooks](https://reactjs.org/docs/hooks-intro.html). For new code in general, please use TypeScript (unless you are modifying a file that was originally in JavaScript).

For changes made to the Desktop client that affect the user interface, refer to `ElectronClient/app/theme.js` for all styling information. The goal is to create a consistent user interface to allow for easy navigation of Joplin's various features and improve the overall user experience.

## Automated tests

When submitting a pull request for a new feature or a bug fix, please add automated tests for your code whenever possible. Tests in Joplin are divided into \*\*unit tests\*\* and \*\*feature tests\*\*.

\* \*\*Unit tests\*\* are used to test models, services or utility classes - they are relatively low level. Unit tests should be prefixed with the type of class that is being tested - for example "models\_Folder" or "services\_SearchEngine".

\* \*\*Feature tests\*\* on the other hand are to test higher level functionalities such as interactions with the GUI and how they affect the underlying model. Often these tests would dispatch Redux actions, and inspect how the application state has been changed. The feature tests should be prefixed with "feature\_", for example "feature\_TagList". There's a good explanation on what qualifies as a feature test in [this post](https://github.com/laurent22/joplin/pull/2819#issuecomment-603502230).

The tests are under CliClient/tests. To get them running, you first need to build the CLI app:

```sh

npm install

cd CliClient

```

To run all the test units:

```sh

npm test

```

To run just one particular file:

```sh

npm test -- --filter=markdownUtils # Don't add the .js extension

```

To filter tests. For example, to run all the test units that contain "should handle conflict" in their description:

```sh

npm test -- --filter="should handle conflict"

```

## About abandoned pull requests

It happens that a pull request is started but not finished and despite our attempts to contact the contributor, we don't hear from them again.

In that case we will not merge the pull request, even if only small changes are missing. Our policy is simply to close the pull request. Why? Because an unfinished pull request essentially means giving us work and moving on. We would rather not encourage this behaviour.

Also, please note that since we have spent time reviewing the pull request and proposing solutions, we reserve the right to re-use that knowledge to create a new pull request, potentially based on your changes.

We'd much prefer that you complete the pull request though, so we'll be sure to ping you a few times before that!