# Contributing to DevDocs

Want to contribute? Great. Please review the following guidelines carefully and search for existing issues before opening a new one.

\*\*Table of Contents:\*\*

1. [Reporting bugs](#reporting-bugs)

2. [Requesting new features](#requesting-new-features)

3. [Requesting new documentations](#requesting-new-documentations)

4. [Contributing code and features](#contributing-code-and-features)

5. [Contributing new documentations](#contributing-new-documentations)

6. [Updating existing documentations](#updating-existing-documentations)

7. [Other contributions](#other-contributions)

8. [Coding conventions](#coding-conventions)

9. [Questions?](#questions)

## Reporting bugs

1. Update to the most recent master release; the bug may already be fixed.

2. Search for existing issues; it's possible someone has already encountered this bug.

3. Try to isolate the problem and include steps to reproduce it.

4. Share as much information as possible (e.g. browser/OS environment, log output, stack trace, screenshots, etc.).

## Requesting new features

1. Search for similar feature requests; someone may have already requested it.

2. Make sure your feature fits DevDocs's [vision](../README.md#vision).

3. Provide a clear and detailed explanation of the feature and why it's important to add it.

## Requesting new documentations

Please don't open issues to request new documentations.

Use the [Trello board](https://trello.com/b/6BmTulfx/devdocs-documentation) where everyone can vote.

## Contributing code and features

1. Search for existing issues; someone may already be working on a similar feature.

2. Before embarking on any significant pull request, please open an issue describing the changes you intend to make. Otherwise you risk spending a lot of time working on something we may not want to merge. This also tells other contributors that you're working on the feature.

3. Follow the [coding conventions](#coding-conventions).

4. If you're modifying the Ruby code, include tests and ensure they pass.

5. Try to keep your pull request small and simple.

6. When it makes sense, squash your commits into a single commit.

7. Describe all your changes in the commit message and/or pull request.

## Contributing new documentations

See the [`docs` folder](https://github.com/freeCodeCamp/devdocs/tree/master/docs) to learn how to add new documentations.

\*\*Important:\*\* the documentation's license must permit alteration, redistribution and commercial use, and the documented software must be released under an open source license. Feel free to get in touch if you're not sure if a documentation meets those requirements.

In addition to the [guidelines for contributing code](#contributing-code-and-features), the following guidelines apply to pull requests that add a new documentation:

\* Your documentation must come with an official icon, in both 1x and 2x resolutions (16x16 and 32x32 pixels). This is important because icons are the only thing differentiating search results in the UI.

\* DevDocs favors quality over quantity. Your documentation should only include documents that most developers may want to read semi-regularly. By reducing the number of entries, we make it easier to find other, more relevant entries.

\* Remove as much content and HTML markup as possible, particularly content not associated with any entry (e.g. introduction, changelog, etc.).

\* Names must be as short as possible and unique across the documentation.

\* The number of types (categories) should ideally be less than 100.

## Updating existing documentations

If the latest [documentation versions report](https://github.com/freeCodeCamp/devdocs/issues?utf8=%E2%9C%93&q=Documentation+versions+report+is%3Aissue+author%3Adevdocs-bot+sort%3Acreated-desc) wrongly shows a documentation to be up-to-date, please open an issue or a PR to fix it.

\*\*Important:\*\* PR's that update documentation versions that do not contain the checklist shown to you in section B of the PR template may be closed without review.

Follow the following steps to update documentations to their latest version:

1. Make version/release changes in the scraper file.

2. Check if the license is still correct. If you update `options[:attribution]`, also update the documentation's entry in the array in [`assets/javascripts/templates/pages/about\_tmpl.coffee`](../assets/javascripts/templates/pages/about\_tmpl.coffee) to match.

3. If the documentation has a custom icon, ensure the icons in <code>public/icons/\*your\_scraper\_name\*/</code> are up-to-date. If you pull the updated icon from a place different than the one specified in the `SOURCE` file, make sure to replace the old link with the new one.

4. If `self.links` is defined, check if the urls are still correct.

5. Generate the docs using `thor docs:generate <doc>`.

6. Make sure `thor docs:generate` doesn't show errors and that the documentation still works well. Verify locally that everything works and that the categorization of entries is still good. Often, updates will require code changes in the scraper or its filters to tweak some new markup in the source website or to categorize new entries.

7. Repeat steps 5 and 6 for all versions that you updated. `thor docs:generate` accepts a `--version` argument to specify which version to scrape.

8. Create a PR and make sure to fill the checklist in section B of the PR template (remove the other sections).

## Coding conventions

\* two spaces; no tabs

\* no trailing whitespace; blank lines should have no spaces; new line at end-of-file

\* use the same coding style as the rest of the codebase

## Questions?

If you have any questions, please feel free to ask them on the contributor chat room on [Gitter](https://gitter.im/FreeCodeCamp/DevDocs).