# Contributing

[![Gitter chat][gitter-image]][gitter-url]

[![Merged PRs][prs-merged-image]][prs-merged-url]

[![GitHub contributors][contributors-image]][contributors-url]

[![CLA assistant][cla-assistant-image]][cla-assistant-url]

[![license][license-image]][license-url]

[gitter-url]: https://gitter.im/tldr-pages/tldr

[gitter-image]: https://img.shields.io/badge/chat-on\_gitter-deeppink

[prs-merged-url]: https://github.com/tldr-pages/tldr/pulls?q=is:pr+is:merged

[prs-merged-image]: https://img.shields.io/github/issues-pr-closed-raw/tldr-pages/tldr.svg?label=merged+PRs&color=green

[contributors-url]: https://github.com/tldr-pages/tldr/graphs/contributors

[contributors-image]: https://img.shields.io/github/contributors/tldr-pages/tldr.svg

[cla-assistant-url]: https://cla-assistant.io/tldr-pages/tldr

[cla-assistant-image]: https://cla-assistant.io/readme/badge/tldr-pages/tldr

[license-url]: https://github.com/tldr-pages/tldr/blob/master/LICENSE.md

[license-image]: https://img.shields.io/badge/license-CC\_BY\_4.0-blue.svg

Contributions to the tldr-pages project are [most welcome](GOVERNANCE.md)!

All `tldr` pages are stored in Markdown right here on GitHub.

Just open an issue or send a pull request and we'll incorporate it as soon as possible.

To get started, please [sign](https://cla-assistant.io/tldr-pages/tldr) the

[Contributor License Agreement](https://gist.github.com/waldyrious/e50feec13683e565769fbd58ce503d4e).

\*Note\*: when submitting a new command, don't forget to check if there's already a pull request in progress for it.

## Guidelines

The basic format of a `tldr` page is a set of concrete usage examples.

Here are a few guidelines to get started:

1. Try to keep pages at around 5 examples. Pages can be longer if needed, but don't exceed 8 examples.

Remember, it's OK if the page doesn't cover everything; that's what `man` is for.

2. When in doubt, keep new command-line users in mind. Err on the side of clarity rather than terseness.

For example, commands that require `sudo` should include it directly in the examples.

3. Try to incorporate the spelled-out version of single-letter options in the example's description.

The goal is to allow people to \*understand\* the syntax of the commands, not just \*memorize\* it.

4. Introduce options gradually, starting with the simplest command invocations,

and using more complex examples progressively.

5. Focus on details specific to the command, and avoid explaining general UNIX concepts that could apply to any command

(ex: relative/absolute paths, glob patterns/wildcards, special character escaping...).

These are all guidelines, not strict rules.

Use proper judgement, keeping simplicity and user-friendliness as the top priorities.

When in doubt, have a look at a few existing pages :).

## Markdown format

As a quick reference, the format of each page should match the following template:

```

# command-name

> Short, snappy description.

> Preferably one line; two are acceptable if necessary.

- Example description:

`command -opt1 -opt2 -arg1 {{arg\_value}}`

- Example description:

`command -opt1 -opt2`

```

For more detailed page formatting guidelines,

refer to the [style guide](contributing-guides/style-guide.md).

## Subcommands

Many programs use subcommands for separating functionality, which may require their own separate pages.

For instance, `git commit` has its own page, as well as `git push` and many others.

To create a page for a subcommand, the program and subcommand need to be separated with a dash (`-`), so `git-commit.md` is shown when calling `tldr git commit`.

You should always add a base page (e.g. `git`) that describes the program and basic switches like `--version` or `help`.

See these examples for reference:

\* [git](pages/common/git.md)

\* [git-commit](pages/common/git-commit.md)

\* [aws](pages/common/aws.md)

\* [aws-s3](pages/common/aws-s3.md)

## Translations

Translation of pages can be done by simply creating the corresponding page within the appropriate language-specific directory, creating that as well if it does not already exist.

Language specific directories must follow the pattern `pages.<locale>`, where `<locale>` is a [POSIX Locale Name](https://www.gnu.org/software/gettext/manual/html\_node/Locale-Names.html#Locale-Names) in the form of `<language>[\_<country>]`, where:

- `<language>` is the shortest [ISO 639](https://en.wikipedia.org/wiki/ISO\_639) language code for the chosen language (see [here](https://en.wikipedia.org/wiki/List\_of\_ISO\_639-2\_codes) for a complete list).

- `<country>` is the two-letter [ISO 3166-1](https://en.wikipedia.org/wiki/ISO\_3166-1) country code for the chosen region (see [here](https://en.wikipedia.org/wiki/ISO\_3166-1\_alpha-2#Officially\_assigned\_code\_elements) for a complete list).

The `<country>` code is optional and should only be added when it is needed. In other words, only when there is a valid reason to distinguish between a language (`ll`) and its regional dialects (`ll\_CC1`, `ll\_CC2`, etc.). As an example, both `fr\_FR` and `fr\_BE` should fall under the same `pages.fr` directory, since there virtually is no difference in writing between standard French and Belgian French.

Some examples of valid locale tags:

- French: `fr`.

- Chinese: `zh`.

- Chinese (Singapore): `zh\_SG`.

- Portuguese (Brazil): `pt\_BR`.

### Default language for newly added pages

The default language used for pages is English (US). Pages written in English are stored in the default `pages` directory (notice the absence of a specific language tag). Although not strictly required, if you'd like to add a new page in a different language, please consider creating the English page too.

## Submitting a pull request

The easiest way to submit a change is to just edit the page directly on the GitHub interface.

Check out the step-by-step instructions (with screenshots) on

[GitHub Help](https://help.github.com/articles/editing-files-in-another-user-s-repository/).

Alternatively, you can do most of the process

[using git on the command line](contributing-guides/git-terminal.md).

### Commit message

For the commit message, use the following format:

<command>: type of change

Examples:

- For a new page addition: `ls: add page`

- For a page edit: `cat: fix typo`, `git-push: add --force example`

- For a new translation of an existing page: `cp: add Tamil translation`

- For related changes to several pages: `grep, find, locate: synchronize format of wildcards`

## Licensing

This repository is licensed under the [Creative Commons Attribution 4.0 International License](LICENSE.md).

The contents of the `scripts/` directory are licensed under the [MIT license](LICENSE.md).

Any contributions to this project are governed by the

[Contributor License Agreement](https://cla-assistant.io/tldr-pages/tldr).