# Contributing to PSReadLine

We welcome and appreciate contributions from the community.

There are many ways to become involved with PSReadLine, including

- Filing issues

- Joining in design conversations

- Writing and improving documentation

- Contributing to the code

Please read the rest of this document to ensure a smooth contribution process.

## Intro to Git and GitHub

\* Make sure you have a [GitHub account](https://github.com/signup/free).

\* Learning Git:

\* GitHub Help: [Good Resources for Learning Git and GitHub][good-git-resources]

\* [Git Basics][git-basics]: install and getting started

\* [GitHub Flow Guide](https://guides.github.com/introduction/flow/):

step-by-step instructions of GitHub Flow

## Contributor License Agreement (CLA)

To speed up the acceptance of any contribution to any PowerShell repositories,

you should sign the [Microsoft Contributor License Agreement (CLA)][cla] ahead of time.

If you've already contributed to PowerShell or Microsoft repositories in the past, congratulations!

You've already completed this step.

This a one-time requirement for the PowerShell projects.

Signing the CLA process is simple and can be done in less than a minute.

You don't have to do this up-front.

You can simply clone, fork, and submit your pull request as usual.

When your pull request is created, it is checked by the CLA bot.

If you have signed the CLA, the status check will be set to `passing`. Otherwise, it will stay at `pending`.

Once you sign a CLA, all your existing and future pull requests will have the status check automatically set at `passing`.

## Contributing to Issues

\* Check if the issue you are going to file already exists in our [GitHub issues][open-issue].

\* If you can't find your issue already,

[open a new issue](https://github.com/PowerShell/PSReadLine/issues/new/choose),

making sure to follow the directions as best you can.

\* If the issue is marked as [Up-for-Grabs][up-for-grabs],

the PSReadLine Maintainers are looking for help with the issue.

## Contributing to Documentation

The documentation is located in the [docs][psreadline-docs] folder.

The markdown files there are converted to the PowerShell `help.xml` file via [platyPS][platy-ps] during the build.

## Contributing to Code

### Forks and Pull Requests

GitHub fosters collaboration through the notion of [pull requests][using-prs].

On GitHub, anyone can [fork][fork-a-repo] an existing repository

into their own user account, where they can make private changes to their fork.

To contribute these changes back into the original repository,

a user simply creates a pull request in order to "request" that the changes be taken "upstream".

Additional references:

\* GitHub's guide on [forking](https://guides.github.com/activities/forking/)

\* GitHub's guide on [Contributing to Open Source](https://guides.github.com/activities/contributing-to-open-source/#pull-request)

\* GitHub's guide on [Understanding the GitHub Flow](https://guides.github.com/introduction/flow/)

### Bootstrap, Build and Test

To build `PSReadLine` on Windows, Linux, or macOS,

you must have the following installed:

\* .NET Core SDK 2.1.802 or [a newer version](https://www.microsoft.com/net/download)

\* The PowerShell modules `InvokeBuild` and `platyPS`

The build script `build.ps1` can be used to bootstrap, build and test the project.

\* Bootstrap: `./build.ps1 -Bootstrap`

\* Build:

\* Targeting .NET 4.6.1 (Windows only): `./build.ps1 -Configuration Debug -Framework net461`

\* Targeting .NET Core: `./build.ps1 -Configuration Debug -Framework netcoreapp2.1`

\* Test:

\* Targeting .NET 4.6.1 (Windows only): `./build.ps1 -Test -Configuration Debug -Framework net461`

\* Targeting .NET Core: `./build.ps1 -Test -Configuration Debug -Framework netcoreapp2.1`

After build, the produced artifacts can be found at `<your-local-repo-root>/bin/Debug`.

### Submitting Pull Request

\* If your change would fix a security vulnerability,

first follow the [vulnerability issue reporting policy][vuln-reporting], before submitting a PR.

\* To avoid merge conflicts, make sure your branch is rebased on the `master` branch of this repository.

\* Many code changes will require new tests,

so make sure you've added a new test if existing tests do not effectively cover the code changed.

\* If your change adds a new source file, ensure the appropriate copyright and license headers is on top.

It is standard practice to have both a copyright and license notice for each source file.

\* For `.cs` files use the copyright header with empty line after it:

```c#

// Copyright (c) Microsoft Corporation.

// Licensed under the 2-Clause BSD License.

<Add empty line here>

```

\* For `.ps1` files use the copyright header with empty line after it:

```powershell

# Copyright (c) Microsoft Corporation.

# Licensed under the 2-Clause BSD License.

<Add empty line here>

```

\* If you're contributing in a way that changes the user or developer experience, you are expected to document those changes.

\* When you create a pull request,

use a meaningful title for the PR describing what change you want to check in.

Make sure you also include a summary about your changes in the PR description.

The description is used to create change logs,

so try to have the first sentence explain the benefit to end users.

If the changes are related to an existing GitHub issue,

please reference the issue in the PR description (e.g. ```Fix #11```).

See [this][closing-via-message] for more details.

[up-for-grabs]: https://github.com/PowerShell/PSReadLine/issues?q=is%3Aopen+is%3Aissue+label%3AUp-for-Grabs

[good-git-resources]: https://help.github.com/articles/good-resources-for-learning-git-and-github/

[git-basics]: https://github.com/PowerShell/PowerShell/blob/master/docs/git/basics.md

[cla]: https://cla.microsoft.com/

[open-issue]: https://github.com/PowerShell/PSReadLine/issues

[psreadline-docs]: https://github.com/PowerShell/PSReadLine/tree/master/docs

[platy-ps]: https://www.powershellgallery.com/packages/platyPS

[using-prs]: https://help.github.com/articles/using-pull-requests/

[fork-a-repo]: https://help.github.com/articles/fork-a-repo/

[vuln-reporting]: https://github.com/PowerShell/PowerShell/blob/master/.github/SECURITY.md

[closing-via-message]: https://help.github.com/articles/closing-issues-via-commit-messages/