# Contributing

We work hard to provide a high-quality and useful command line interface, and we greatly value feedback and contributions from our community. Whether it's a new feature, correction, or additional documentation, we welcome your pull requests. Please submit any [issues](https://github.com/aws/aws-cli/issues) or [pull requests](https://github.com/aws/aws-cli/pulls) through GitHub.

This document contains guidelines for reporting issues or pull requests and contributing code.

## Reporting Issues

- Check to see if there\'s an existing issue/pull request for the bug/feature. All issues are at <https://github.com/aws/aws-cli/issues> and pull reqs are at <https://github.com/aws/aws-cli/pulls>.

- If there isn\'t an existing issue there, please file an issue. If possible, used one of the suggested issue types when creating a new issue (like a bug report of feature request). These issue types have their own template and required information. In general, the ideal report includes:

- A description of the problem/suggestion.

- The specific AWS CLI commands you are running. Please include debug logs for these commands by appending the `--debug` option to each command. Be sure to remove any sensitive information from the debug logs.

- The AWS CLI version you are using `aws --version`.

The first thing an AWS CLI developer will do is try to reproduce the issue you are seeing, so try to reduce your issue to the smallest possible set of steps that demonstrate the issue. This will lead to quicker resolution of your issue.

## Contributing Code

The list below are guidelines to use when submitting pull requests. These are the same set of guidelines that the core contributors use when submitting changes, and we ask the same of all community contributions as well:

- The SDK is released under the [Apache license](http://aws.amazon.com/apache2.0/). Any code you submit will be released under that license.

- We maintain a high percentage of code coverage in our unit tests. As a general rule of thumb, code changes should not lower the overall code coverage percentage for the project. In practice, this means that \*\*every bug fix and feature addition should include tests.\*\*

- Code should follow [pep8](https://www.python.org/dev/peps/pep-0008/), although if you are modifying an existing module, it is more important for the code to be consistent if there are any discrepancies. Using [`flake8`](https://flake8.pycqa.org/en/latest/) can assist in identifying `pep8` compliance issues.

- Code must work on `python2.7`, and `python3.4` and higher.

- The AWS CLI is cross platform and code must work on at least Linux, Windows, and Mac OS X. Avoid platform specific behavior.

- If you would like to implement support for a significant feature that is not yet available in the AWS CLI, please talk to us beforehand to avoid any duplication of effort. You can file an [issue](https://github.com/aws/aws-cli/issues) to discuss the feature request further.

## Git Commits and Workflow

When sending a pull request, please follow these guidelines:

- The PR should target the `develop` branch. If you send a PR to the `master` branch, the travis CI jobs will fail.

- Your PR branch should be based off a recent commit of the `develop` branch. Preferably the base commit for the PR should use the latest commit of `develop` at the time the PR was created. This helps to ensure there are no merge conflicts or test failures when the PR is merged back to the develop branch.

- Make separate commits for logically separate changes. Avoid commits such as \"update\", \"fix typo again\", \"more updates\". Rebase your commits before submitting your PR to ensure they represent a logical change.

- Avoid merge commits in your PRs. If you want to pull in the latest changes from the `develop` branch, rebase on top of the `develop` branch instead of merging the `develop` branch into your feature branch.

Also, ensure your commit messages match this format:

Short (50 characters or less) summary

After the 50 character summary and a blank line, you can include a body if necessary. Note that the 50 character summary does not end with any punctuation. Describe your changes in the imperative mood, e.g., "Add foo to bar", "Update foo component for bar", "Fix race condition for foo".

The body of the commit message can include:

\* an explanation of the problem and what this change tries to solve.

\* rationale behind the specific implementation

\* alternatives considered and why they were discarded, if appropriate.

Please limit the line length in the body of a commit message to 80 characters or less.

### Example Git Workflow

Below is an example of how you can use git to create a feature branch. First, make sure you've created a fork of `aws/aws-cli`. Then you can run these commands:

# Clone the repo and set up the remotes.

$ git clone git@github.com:myusername/aws-cli.git

$ cd aws-cli

$ git remote add upstream https://github.com/aws/aws-cli.git

$ git fetch upstream

$ git merge upstream/develop

# Now to create a feature branch:

$ git checkout -b my-branch-name

# Now add your commits for your features.

$ git add path/to/my/files

# Make sure our commit message matches format described in the

# previous section.

$ git commit -m "Add support for foo"

# If we want to sync with the latest upstream changes before

# sending our pull request we can run:

$ git fetch upstream

$ git rebase upstream/develop

# When you're ready to send a PR, make sure you push your commits

# to your fork:

$ git push origin my-branch-name

When you push to your remote, the output will contain a URL you can use to open a pull request.

## CLI Development Version

If you are interested in using the latest released version of the AWS CLI, please see the [Installation](README.md#installation) section in the README. This section is for anyone who wants to install the development version of the CLI. You might need to do this if:

- You are developing a feature for the CLI and plan on submitting a Pull Request.

- You want to test the latest changes of the CLI before they make it into an official release.

The latest changes to the CLI are in the `develop` branch on github. This is the default branch when you clone the git repository.

Additionally, the [`botocore`](https://github.com/boto/botocore) package is developed in lockstep with the CLI.

If you just want to install a snapshot of the latest development version of the CLI, you can use the `requirements.txt` file included in this repo. This file points to the development version of the above packages:

$ cd <path\_to\_awscli>

$ python -m pip install -r requirements.txt

$ python -m pip install -e .

However, to keep up to date, you will continually have to run the `python -m pip install -r requirements.txt` file to pull in the latest changes from the develop branch of botocore.

You can optionally clone each of those repositories and run \"python -m pip install -e .\":

$ git clone <botocore> && cd botocore/

$ python -m pip install -e . && cd ..

$ git clone <awscli> && cd aws-cli/

$ python -m pip install -e .