# Contributing to Moq

Moq welcomes your contributions! :heart:

## A quick word about the upcoming next major version of Moq (v5)

The next major version of Moq is currently in active development (over at the [moq/moq](https://github.com/moq/moq) repository). Because of this, we are somewhat hesitant about big API changes in Moq v4. The plan for v5 is to have an API very similar to Moq v4 (to make migration easy). Therefore, any radical changes in v4 might have to be reproduced in v5, which costs additional time.

For this reason, the main (but not sole!) focus at this repository (v4) is on bug fixing, improving stability, and performance. That being said, small additions stand a good chance of getting in if they seem generally useful.

## Asking usage questions

This repository's focus is on the development of Moq (i.e. fixing bugs and improving features; see above). Because the Moq team is rather small, our resources to answer usage questions is very limited. You might be better off asking usage questions over at [Stack Overflow](https://stackoverflow.com) as your question will meet with a much larger audience. Please observe SO's rules and etiquette, and tag your question with `moq`.

## Submitting a bug report (as an issue)

If you think you've found a bug, please start by searching [the changelog](https://github.com/moq/moq4/blob/master/CHANGELOG.md) and [GitHub issues](https://github.com/moq/moq4/issues) (both open and closed ones!) to see if the problem has already been addressed or documented in any way.

If you find nothing of relevance, [open a new issue](https://github.com/moq/moq4/issues/new).

\*\*What to include:\*\* Try to include the following information in it:

\* a brief summary

\* minimal, complete, and verifiable repro code (similar to what you'd have to prepare for a Stack Overflow question; see e.g. [their documentation on "How to create a Minimal, Complete, and Verifiable example"](https://stackoverflow.com/help/mcve)

\* a description of the expected (correct) outcome

\* a description of the actual (incorrect) outcome

\* Moq version used

## Submitting a pull request (PR)

Unless for very simple and straightforward changes, please open an issue first to discuss the PR you're about to submit with the Moq team.

\*\*What to include:\*\* Once you do submit a PR, please include the following:

\* Unit tests whenever you add a bug fix, new feature, or when you change an existing one.

\* All changes relevant to Moq users should be documented in the `CHANGELOG.md`. Please describe your change or bug fix in that file (more specifically under the \_Unreleased\_ heading at the beginning of it). Phrase the summary in a way that it will be easily understandable and useful to other Moq users.

```diff

# Unreleased

...

#### Fixed

+\* <Short explanation of the bug fix> (@<your username>, #<issue or PR number>)

```

Note the inclusion of your user name and a reference to the relevant PR or issue. The former is included to give you credit for your contribution. The latter allows users to trace the history of a change back to the issue where it was first reported.

\*\*Structuring your PR commits (example):\*\* One good way, but not the only one, of structuring your PR might be to follow a test-first approach:

1. Start with a commit that adds one or more failing unit tests.

2. Add one or more commits that make these unit tests pass.

3. Add a final commit with a new changelog entry.

Of course, if your changes are small, you might combine all these steps into fewer or even a single commit.

\*\*When a reviewer requests changes to your PR,\*\* we encourage (but don't require) you to keep your PR tidy using any Git facilities available: You can rewrite your PR branch's history by amending existing commits, rebasing, etc., then force-pushing the changed commits to your PR branch. If you are not familiar enough with Git to do all of this, simply adding more commits to make requested changes is fine. A Moq team member merging your PR may decide to "squash" them into a single commit to keep the repository's history more easily readable.

## Code style rules

\*\*Indentation, line endings, etc.\*\*: The project includes a [`.editorconfig`](https://editorconfig.org/) file. Please make sure your IDE can read this file so that indentation, line endings, etc. is kept consistent across the whole code base. When submitting a PR, try to keep your commit diffs free of whitespace-only changes.

\*\*Copyright notice:\*\* If you add a new file to the solution, please reproduce the short copyright notice found at the beginning of existing files.

\*\*.NET code conventions:\*\* The Moq codebase follows the usual .NET code conventions as documented e.g. in [Framework Design Guidelines](https://docs.microsoft.com/en-us/dotnet/standard/design-guidelines/); so following the existing conventions should be no big issue when adding new code.