# Contributing to lit-html

Thank you for your interest in contributing to lit-html!

There are many ways to contribute to lit-html project, and we have many different needs to be addressed. All contributions, from PRs to reports of successful usage, are appreciated and valuable.

## Code of Conduct

We have a [Code of Conduct](./CODE\_OF\_CONDUCT.md), please follow it in all interactions with project maintainers and fellow users.

## Filing Issues

Issues are one of the most important ways to contribute to lit-html.

Please search though open and closed issues to see if a similar issue already exists. If not, open an issue and try to provide a minimal reproduction if you can.

Occasionally we'll close issues if they appear stale or are too vague - please don't take this personally! Please feel free to re-open issues we've closed if there's something we've missed and they still need to be addressed.

## Pull Requests

Pull requests are greatly appreciated! To ensure a smooth review process, please follow these steps:

1. Make sure there's an open issue that the PR addresses. Add "Fixes #(issue number)" to the PR description.

2. Please discuss the general shape of the change ahead of time. This can save much time for reviewers and submitters alike. Many times there may be existing ideas on how to handle an issue that are not fully written out, and asking about it will bring out more details.

3. All PRs that change behavior or fix bugs should have new or updated tests.

4. Try to create a set of descriptive commits that each do one focused change. Avoid commits like "oops", and prefer commits like "Added method foo to Bar".

5. When addressing review comments, try to add new commits, rather than modifying previous commits. This makes it easier for reviewers to see what changed since the last review. `git commit --fixup {SHA}` is really useful for this. Obviously, requests like "Please rebase onto master" require changing commits.

6. If you [allow changes to be committed to your PR branches](https://help.github.com/articles/allowing-changes-to-a-pull-request-branch-created-from-a-fork/) we can fix some small things in the PR for you, speeding up the review process. This is especially useful if you're new to TypeScript and need help with type annotations.

7. Please run `npm run lint` and `npm run format` before submitting PRs. PRs that don't lint and aren't formatted will fail continuous integration tests.

## Code Style

We follow the [Google JavaScript Style Guide](https://google.github.io/styleguide/jsguide.html), but there are a couple of points worth emphasizing:

1. Clear is better than clever. Optimize for simple, readable code first.

2. Prefer longer, more descriptive names, over shorter names. For most variables, minification means we don't pay for extra characters in production.

3. Always err on the side of too many comments. When commenting, "why" is more important than "what".

4. If you're tempted to add a "what" comment, see if you can't restructure the code and use more descriptive names so that the comment is unnecessary.

## TypeScript

We use TypeScript on lit-html in order to automatically check the code for type errors and document the types of fields and attributes for easier reading. If you don't know TypeScript, we hope it doesn't discourage you from contributing - TypeScript is a superset of JavaScript that focuses on adding type annotations.

TypeScript is hopefully relatively easy to pick up, but if you have any problems we're more than happy to help. You can submit a pull request with type warnings and we'll either help you fix them, or if you allow commits to your PR branch, fix them for you. VS Code is a very nice IDE for TypeScript development if you care to try it.

We stick to subset of TypeScript that is more strict and closer to standard JavaScript.

1. We have strict compiler options turned on. Do not change them.

2. Prefer the `unknown` type over `any`.

3. Prefer the `object` type over `Object`

4. Prefer named type alias over complex inline types.

5. Use web-compatible, full URLs as import specifiers, including file extensions. ie, `import \* as foo from './foo.js`, not `import \* as foo from './foo`

6. Only use TypeScript for types:

1. We compile to the `esnext` target and don't use TypeScript for "downlevel" compilation. Do not change that.

2. Don't use features that are not part of the type system:

1. `namespace`

2. `enum`

3. Parameter properties (initialize class fields in the constructor parameter list)

7. Prefix private members with `\_`. (don't assume clients are using TypeScript)

## Contributor License Agreement

You might notice our friendly CLA-bot commenting on a pull request you open if you haven't yet signed our CLA. We use the same CLA for all open-source Google projects, so you only have to sign it once. Once you complete the CLA, all your pull-requests will automatically get the `cla: yes` tag.

If you've already signed a CLA but are still getting bothered by the awfully insistent CLA bot, it's possible we don't have your GitHub username or you're using a different email address. Check the [information on your CLA](https://cla.developers.google.com/clas) or see this help article on [setting the email on your git commits](https://help.github.com/articles/setting-your-email-in-git/).

[Complete the CLA](https://cla.developers.google.com/clas)