# Contributing to Polymer

There are many ways to contribute to the Polymer project! We welcome and truly appreciate contribution in all forms - issues and pull requests to the [main library](https://github.com/polymer/polymer), issues and pull requests to the [elements the Polymer team maintains](https://github.com/polymerelements), issues and pull requests to one of our many [Polymer-related tools](https://github.com/polymer), and of course we love to hear about any Polymer elements that you build to share with the community!

## Logistics

### Communicating with the Polymer team

Beyond GitHub, we try to have a variety of different lines of communication open:

\* [Blog](https://blog.polymer-project.org/)

\* [Twitter](https://twitter.com/polymer)

\* [Mailing list](https://groups.google.com/forum/#!forum/polymer-dev)

\* [Slack channel](https://bit.ly/polymerslack)

### The Polymer Repositories

Because of the component-based nature of the Polymer project, we tend to have lots of different repositories. Our main repository for the Polymer library itself is at [github.com/Polymer/polymer](https://github.com/polymer/polymer). File any issues or pull requests that have to do with the core library on that repository, and we'll take a look ASAP.

We keep all of the element "product lines" that the Polymer team maintains and distributes in the [PolymerElements](https://github.com/polymerelements) organization. For any element-specific issues or pull requests, file directly on the element's repository, such as the `paper-button` repository at [github.com/polymerelements/paper-button](https://github.com/polymerelements/paper-button). Of course, the elements built by the Polymer team are just a tiny fraction of all the Polymer-based elements out there - catalogs of other web components include [https://www.webcomponents.org/](https://github.com/webcomponents/webcomponents.org) and [component.kitchen](https://component.kitchen).

The GoogleWebComponents element product line is maintained by teams all across Google, and so is kept in a separate organization: the [GoogleWebComponents](https://github.com/googlewebcomponents) org. Feel free to file issues and PR's on those elements directly in that organization.

We also track each element product line overall in "meta-repos", named as `$PRODUCTLINE-elements`. These include [paper-elements](https://github.com/polymerelements/paper-elements), [iron-elements](https://github.com/polymerelements/iron-elements), [gold-elements](https://github.com/polymerelements/gold-elements), and more. Feel free to file issues for element requests on those meta-repos, and the README in each repo tracks a roadmap for the product line.

### 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)

## Contributing

### Contributing documentation

Docs source is in the `docs` folder. To build the site yourself, see the instructions in [docs/README.md](docs/README.md).

### Filing bugs

The Polymer team heavily uses (and loves!) GitHub for all of our software management. We use GitHub issues to track all bugs and features.

If you find an issue, please do file it on the repository. The [lit-element issues](https://github.com/Polymer/lit-element/issues) should be used only for issues on the lit-element library itself - bugs somewhere in the core codebase.

For issues with elements the team maintains, please file directly on the element's repository. If you're not sure if a bug stems from the element or the library, air toward filing it on the element and we'll move the issue if necessary.

Please file issues using the issue template provided, filling out as many fields as possible. We love examples for addressing issues - issues with a jsBin, Plunkr, jsFiddle, or glitch.me repro will be much easier for us to work on quickly. You can start with [this StackBlitz](https://stackblitz.com/edit/lit-element-example?file=index.js) which sets up the basics to demonstrate a lit-element. If you need your repro to run in IE11, you can start from [this glitch](https://glitch.com/edit/#!/hello-lit-element?path=index.html:1:0), which serves the source via polyserve for automatic transpilation, although you must sign up for a glitch.me account to ensure your code persists for more than 5 days (note the glitch.me \_editing environment\_ is not compatible with IE11, however the "live" view link of the running code should work).

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.

### Contributing Pull Requests

PR's are even better than issues. We gladly accept community pull requests. In general across the core library and all of the elements, there are a few necessary steps before we can accept a pull request:

- Open an issue describing the problem that you are looking to solve in your PR (if one is not already open), and your approach to solving it. This makes it easier to have a conversation around the best general approach for solving your problem, outside of the code itself.

- Sign the [CLA](https://cla.developers.google.com/clas), as described above.

- Fork the repo you're making the fix on to your own GitHub account.

- Code!

- Ideally, squash your commits into a single commit with a clear message of what the PR does. If it absolutely makes sense to keep multiple commits, that's OK - or perhaps consider making two separate PR's.

- \*\*Include tests that test the range of behavior that changes with your PR.\*\* If you PR fixes a bug, make sure your tests capture that bug. If your PR adds new behavior, make sure that behavior is fully tested. Every PR \*must\* include associated tests. (See [Unit tests](#unit-tests) for more.)

- Submit your PR, making sure it references the issue you created.

- If your PR fixes a bug, make sure the issue includes clear steps to reproduce the bug so we can test your fix.

If you've completed all of these steps the core team will do its best to respond to the PR as soon as possible.

#### Contributing Code to Elements

Though the aim of the Polymer library is to allow lots of flexibility and not get in your way, we work to standardize our elements to make them as toolable and easy to maintain as possible.

All elements should follow the [Polymer element style guide](https://www.polymer-project.org/3.0/docs/tools/documentation), which defines how to specify properties, documentation, and more. It's a great guide to follow when building your own elements as well, for maximum standardization and toolability. For instance, structuring elements following the style guide will ensure that they work with the [`iron-component-page`](https://github.com/polymerelements/iron-component-page) element, an incredibly easy way to turn any raw element directly into a documentation page.

#### Contributing Code to the Polymer library

We follow the most common JavaScript and HTML style guidelines for how we structure our code - in general, look at the code and you'll know how to contribute! If you'd like a bit more structure, the [Google JavaScript Styleguide](https://google.github.io/styleguide/javascriptguide.xml) is a good place to start.

Polymer also participates in Google's [Patch Rewards Program](https://www.google.com/about/appsecurity/patch-rewards/), where you can earn cold, hard cash for qualifying security patches to the Polymer library. Visit the [patch rewards page](https://www.google.com/about/appsecurity/patch-rewards/) to find out more.

## Unit tests

All Polymer projects use [`polymer-cli`](https://github.com/Polymer/tools/tree/master/packages/cli) for unit tests.

For maximum flexibility, install `polymer-cli` locally:

npm install -g polymer-cli

### Running the lit-element unit tests

To run the lit-element unit tests:

1. Clone the [lit-element repo](https://github.com/polymer/lit-element).

2. Install the dependencies:

npm install

3. Run the tests:

npm test

Or if you have `polymer-cli` installed locally:

polymer test --npm

To run individual test suites:

<code>npm test <var>path/to/suite</var></code>

Or:

<code>polymer test --npm <var>path/to/suite</var></code>

For example:

polymer test --npm test/index.html

You can also run tests in the browser:

polymer serve --npm

Navigate to:

[`http://localhost:8080/components/@polymer/lit-element/test/index.html`](http://localhost:8080/components/@polymer/lit-element/test/index.html)

### Configuring `web-component-tester`

By default, `polymer test` runs tests on all installed browsers. You can configure it

to run tests on a subset of available browsers, or to run tests remotely using Sauce Labs.

See the [`web-component-tester` README](https://github.com/Polymer/tools/tree/master/packages/web-component-tester) for

information on configuring the tool using by `polymer-cli` to run the tests.

### Viewing the source documentation locally

You can view the updates you make to the source documentation locally with the following steps.

Make sure to rerun step 1 after every change you make.

1. Run `polymer analyze > analysis.json`

1. Run `polymer serve`

1. Open `http://127.0.0.1:PORT/components/polymer/` to view the documentation