# Contributing to Panzoom

- Prettier enforces the style guide and should at least format on commit. Make sure those changes are added to the commit.

- Run unit tests with `npm test`

- Check demos with `npm start`

\*\*Supported browsers\*\*: (basically IE 11, modern desktop and mobile browsers) https://browserl.ist/?q=%3E0.35%25%2C+not+op\_mini+all

Contributions are always welcome. Before contributing please [search the issue tracker](https://github.com/timmywil/panzoom/issues); your issue

may have already been discussed or fixed in `master`. To contribute,

[fork](https://help.github.com/articles/fork-a-repo/) Panzoom, commit your changes,

& [send a pull request](https://help.github.com/articles/using-pull-requests/).

## Feature Requests

Feature requests should be submitted in the

[issue tracker](https://github.com/timmywil/panzoom/issues), with a description of

the expected behavior & use case, where they’ll remain closed until sufficient interest,

[e.g. :+1: reactions](https://help.github.com/articles/about-discussions-in-issues-and-pull-requests/),

has been [shown by the community](https://github.com/timmywil/panzoom/issues?q=label%3A%22votes+needed%22+sort%3Areactions-%2B1-desc).

Before submitting a request, please search for similar ones in the

[closed issues](https://github.com/timmywil/panzoom/issues?q=is%3Aissue+is%3Aclosed+label%3Afeature).

I got this convention from [lodash](https://github.com/lodash/lodash). It helps keep the open issues list uncluttered.

## Pull Requests

For additions or bug fixes you should only need to modify files in `src/`. Include

updated unit tests in the `test` directory or an updated/added demo in the `demo` directory as part of your pull request. Don’t worry about regenerating the built files.

## Editing documentation

### Do not edit below the "Documentation" header in the README

Edit the comments and type declarations in the `src/` folder, or the markdown files in the `tasks/` folder.

The documentation is auto-generated using a combination of [typedoc](https://typedoc.org/) and a hand-rolled script (`tasks/docs.js`). First, typedoc generates markdown from the TypeScript files into the `docs/` folder, which is ignored by git, and then `tasks/docs.js` concatenates those files and cleans them up for presentation in the README.

## Testing

Tests are written with [mocha](https://mochajs.org/) and [Node's official assert module](https://nodejs.org/api/assert.html#assert\_assert).

Here are the npm scripts that run tests:

```bash

$ npm test # Lints and runs the unit tests

$ npm run test:unit # Runs the unit tests

$ npm test:watch # Watches files and runs the unit tests on file save

```

## Building

The `dist/` folder is ignored on master and included in releases. To build, first make sure dependencies are installed (`npm i`) and run the following:

```bash

$ npm run build

```

This is usually unnecessary as `npm start` will rebuild automatically as you change files.

## Coding Guidelines

In addition to the following guidelines, please follow the conventions already

established in the code.

- \*\*Spacing\*\*:<br>

Use two spaces for indentation. No tabs.

- \*\*Naming\*\*:<br>

Keep variable & method names concise & descriptive.<br>

Variable names `index`, `array`, & `iteratee` are preferable to

`i`, `arr`, & `fn`.

- \*\*Quotes\*\*:<br>

Single-quoted strings are preferred to double-quoted strings; however,

please use a double-quoted string if the value contains a single-quote

character to avoid unnecessary escaping.

- \*\*Comments\*\*:<br>

Comments are kept to a minimum, but are encouraged to explain confusing bits of code.

- \*\*Types\*\*:<br>

Panzoom is written in TypeScript and documentation is generated from the type annotations.

Any code additions should be properly typed, with no use of `any`.

Guidelines are enforced using [TSLint](https://github.com/palantir/tslint) and [Prettier](https://github.com/prettier/prettier):

```bash

$ npm run lint

```

Some things are fixable automatically.

```bash

$ npm run lint:fix

```

This script is run on commit, which means that the commit may need amending if any changes were made as a result of the commit.

Check the working directory is clean of all changes after committing.

## Commit message guidelines

Commit messages should follow [Conventional Commits Specification(https://www.conventionalcommits.org).

This is also enforced on commit using a commit message hook.

Panzoom includes a helpful prompt for committing to guide you in the process of writing a valid commit message.

Run the following after staging files:

```bash

$ npm run commit

```

## Debugging in VS Code

First run `npm start`, then run VS Code Debug with the following config to enable breakpoints:

```json

{

"type": "chrome",

"request": "launch",

"name": "Launch Chrome",

"url": "http://localhost:8080",

"webRoot": "${workspaceFolder}",

"sourceMaps": true

}

```