# Contributing FAQs

- [How do I go about contributing ideas or new features?](#how-do-i-go-about-contributing-ideas-or-new-features?)

- [How should I report bugs?](#how-should-i-report-bugs?)

- [How do I send a pull request?](#how-do-i-send-a-pull-request?)

- [I'm submitting a PR. How do I run tests?](#i'm-submitting-a-pr-how-do-i-run-tests?)

- [How do I build Mithril?](#how-do-i-build-mithril?)

- [Is there a style guide?](#is-there-a-style-guide?)

- [How do I embed live previews in docs?](#how-do-I-embed-live-previews-in-docs?)

- [Why do tests mock the browser APIs?](#why-do-tests-mock-the-browser-apis?)

- [Why does Mithril use its own testing framework and not Mocha/Jasmine/Tape?](#why-does-mithril-use-its-own-testing-framework-and-not-mochajasminetape?)

- [Why doesn't the Mithril codebase use ES6 via Babel or Bublé? Would a PR to upgrade be welcome?](#why-doesn't-the-mithril-codebase-use-es6-via-babel-or-bublé?-would-a-pr-to-upgrade-be-welcome?)

- [Why doesn't the Mithril codebase use trailing semi-colons? Would a PR to add them be welcome?](#why-doesn't-the-mithril-codebase-use-trailing-semi-colons?-would-a-pr-to-add-them-be-welcome?)

- [Why does the Mithril codebase use a mix of `instanceof` and `typeof` checks instead of `Object.prototype.toString.call`, `Array.isArray`, etc? Would a PR to refactor those checks be welcome?](#why-does-the-mithril-codebase-use-a-mix-of-instanceof-and-typeof-checks-instead-of-objectprototypetostringcall,-arrayisarray,-etc?-would-a-pr-to-refactor-those-checks-be-welcome?)

- [What should I know in advance when attempting a performance related contribution?](#What-should-I-know-in-advance-when-attempting-a-performance-related-contribution?)

- [Do you all accept donations?](#do-you-all-accept-donations?)

## How do I go about contributing ideas or new features?

Create an [issue thread on GitHub](https://github.com/MithrilJS/mithril.js/issues/new) to suggest your idea so the community can discuss it.

If the consensus is that it's a good idea, the fastest way to get it into a release is to send a pull request. Without a PR, the time to implement the feature will depend on the bandwidth of the development team and its list of priorities.

## How should I report bugs?

Ideally, the best way to report bugs is to provide a small snippet of code where the issue can be reproduced (via jsfiddle, jsbin, a gist, etc). Even better would be to submit a pull request with a fix and tests. If you don't know how to test your fix, or lint or whatever, submit anyways, and we can help you.

## How do I send a pull request?

To send a pull request:

- fork the repo (button at the top right in GitHub)

- clone the forked repo to your computer (green button in GitHub)

- Switch to the `next` branch (run `git checkout next`)

- create a feature branch (run `git checkout -b the-feature-branch-name`)

- make your changes

- run the tests (run `npm test`)

- push your changes to your fork

- submit a pull request (go to the pull requests tab in GitHub, click the green button and select your feature branch)

## I'm submitting a PR. How do I run tests?

Assuming you have forked this repo, you can open the `index.html` file in a module's `tests` folder and look at console output to see only tests for that module, or you can run `ospec/bin/ospec` from the command line to run all tests.

While testing, you can modify a test to use `o.only(description, test)` instead of `o(description, test)` if you wish to run only a specific test to speed up your debugging experience. Don't forget to remove the `.only` after you're done!

There is no need to `npm install` anything in order to run the test suite, however NodeJS is required to run the test suite from the command line. You do need to `npm install` if you want to lint or get a code coverage report though.

## How do I build Mithril?

If all you're trying to do is run examples in the codebase, you don't need to build Mithril, you can just open the various html files and things should just work.

To generate the bundled file for testing, run `npm run dev` from the command line. To generate the minified file, run `npm run build`. There is no need to `npm install` anything, but NodeJS is required to run the build scripts.

## Is there a style guide?

Yes, there's an `eslint` configuration, but it's not strict about formatting at all. If your contribution passes `npm run lint`, it's good enough for a PR (and it can still be accepted even if it doesn't pass).

Spacing and formatting inconsistencies may be fixed after the fact, and we don't want that kind of stuff getting in the way of contributing.

## How do I embed live previews in docs?

Any code tag marked as `js` and not `javascript` will automatically be wrapped in a live Flems preview.

## Why do tests mock the browser APIs?

Most notoriously, because it's impossible to test the router and some side effects properly otherwise. Also, mocks allow the tests to run under Node.js without requiring heavy dependencies like PhantomJS/ChromeDriver/JSDOM.

Another important reason is that it allows us to document browser API quirks via code, through the tests for the mocks.

## Why does Mithril use its own testing framework and not Mocha/Jasmine/Tape?

Mainly to avoid requiring dependencies. `ospec` is customized to provide only essential information for common testing workflows (namely, no spamming ok's on pass, and accurate noiseless errors on failure)

## Why doesn't the Mithril codebase use ES6 via Babel or Bublé? Would a PR to upgrade be welcome?

Being able to run Mithril's raw source code in all supported browsers is a requirement for all browser-related modules in this repo. In addition, transpiled code is generally much bulkier.

## Why doesn't the Mithril codebase use trailing semi-colons? Would a PR to add them be welcome?

I don't use them. Adding them means the semi-colon usage in the codebase will eventually become inconsistent. Besides, [we aren't the only one who've decided to drop the semicolon](https://standardjs.com/#who-uses-javascript-standard-style). (We don't use Standard, though.)

## Why does the Mithril codebase use a mix of `instanceof` and `typeof` checks instead of `Object.prototype.toString.call`, `Array.isArray`, etc? Would a PR to refactor those checks be welcome?

Mithril avoids peeking at objects' [[class]] string for performance considerations. Many type checks are seemingly inconsistent, weird or convoluted because those specific constructs demonstrated the best performance profile in benchmarks compared to alternatives.

Type checks are generally already irreducible expressions and having micro-modules for type checking subroutines would add maintenance overhead.

## What should I know in advance when attempting a performance related contribution?

You should be trying to reduce the number of DOM operations or reduce algorithmic complexity in a hot spot. Anything else is likely a waste of time. Specifically, micro-optimizations like caching array lengths, caching object property values and inlining functions won't have any positive impact in modern JavaScript engines.

Keep object properties consistent (i.e. ensure the data objects always have the same properties and that properties are always in the same order) to allow the engine to keep using JIT'ed structs instead of hashmaps. Always place null checks first in compound type checking expressions to allow the JavaScript engine to optimize to type-specific code paths. Prefer for loops over Array methods and try to pull conditionals out of loops if possible.

## Do you all accept donations?

Yes, we do, over at [our OpenCollective page](https://opencollective.com/mithriljs). We don't actively seek donations, but they are very much appreciated and are used to support development and related expenses. Both one-time and recurring donations are accepted.