# Contributing

Thank you for contributing!

<a href="https://github.com/FormidableLabs/spectacle#maintenance-status">

<img alt="Maintenance Status" src="https://img.shields.io/badge/maintenance-active-green.svg" />

</a>

Spectacle is actively maintained by @[carlos-kelly][] and @[kale-stew][]

from within [@FormidableLabs][formidable-github].

## Development

### Installing dependencies

We prefer to use [`yarn`][yarn-docs].

Install all dependencies by running:

```bash

$ yarn

```

### Examples

#### Overview

Our examples are spread out across multiple projects depending on where the core technology lies. We publish most of these to `npm` for use in `spectacle-cli` project to either use with the CLI (`spectacle`) or generate a fresh project boilerplate (`spectacle-boilerplate`).

- `spectacle`

- [`examples/js`](https://github.com/FormidableLabs/spectacle/tree/main/examples/js)

- [`examples/md`](https://github.com/FormidableLabs/spectacle/tree/main/examples/md)

- [`examples/one-page`](https://github.com/FormidableLabs/spectacle/tree/main/examples/one-page.html)

- `spectacle-mdx-loader`

- [`examples/mdx`](https://github.com/FormidableLabs/spectacle-mdx-loader/tree/main/examples/mdx)

- `spectacle-cli`

- [`examples/cli-mdx-babel`](https://github.com/FormidableLabs/spectacle-mdx-loader/tree/main/examples/cli-mdx-babel): \_Not published\_

#### This repository

We have various deck scenarios in `examples` in this repository that are part of the development process.

We follow the convention of `start:NAME` to run an in-memory dev server for a specific

example, but we also have a `yarn build-examples` script task to make sure we're actually

producing non-broken sample presentations as a CI / assurance test.

#### `examples/js`

A basic deck with JSX and JavaScript:

```bash

# start the dev server

$ yarn start:js

# open the browser

$ open http://localhost:3000/

```

\*\*Note\*\*: The files `index.{js,html}`, `slides.md` are published and used by `spectacle-cli`.

#### `examples/md`

A basic deck written in markdown:

```bash

# In one terminal open dev server

$ yarn start:md

# In another open a browser to 3100

$ open http://localhost:3100/

```

\*\*Note\*\*: The files `index.{js,html}`, `slides.md` are published and used by `spectacle-cli`.

#### `examples/one-page`

A self-contained single web page that uses Spectacle, React, and `htm` for a "no build" presentation!

```bash

# [optional] build the library -

# comment out the unpkg dependency in

# one-page.html and use the local dist/

$ yarn build

# open the browser

$ open examples/one-page.html

```

\_or\_ use the single line:

```bash

$ yarn start:one-page

```

\*\*Note\*\*: This file is published and used by `spectacle-cli`.

\*\*Development Note\*\*: This JS code portion of this file is programmatically updated from the source in `examples/js/index.js` directly into `one-page.html`. Rather than editing directly, please run `yarn build-one-page` and verify changes look good.

### Examples integration with `spectacle-cli`

`spectacle-cli` uses our `js,md,one-page` examples in the CLI and boilerplate tools. To check that changes to these files don't break `spectacle-cli` upstream, check with something like the following:

```bash

# In `spectacle` repo

$ yarn link

# In `spectacle-cli` repo

$ yarn link spectacle

# Check all MDX, MD examples per https://github.com/FormidableLabs/spectacle-cli/blob/main/CONTRIBUTING.md#examples

$ yarn start:examples

# (In another shell) Check mdx:5000, mdx+babel:5001, md:5100

$ open http://localhost:5000/ http://localhost:5001/ http://localhost:5100/

# Check all JS, MDX, MD boilerplates per https://github.com/FormidableLabs/spectacle-cli/blob/main/CONTRIBUTING.md#boilerplate

$ yarn clean:boilerplate

$ yarn boilerplate:generate

$ yarn boilerplate:install

$ yarn start:boilerplate

# (In another shell) Check mdx:6300, md:6100, js:6200

#

# \*\*Note\*\*: These `yarn install` internally so will use latest published

# `spectacle`, so results may be not entirely accurage. You may need to manually

# update the installed contents in generated project `node\_modules`.

$ open http://localhost:6300/ http://localhost:6100/ http://localhost:6200/

```

### Testing

To run all tests:

```bash

$ yarn test

```

### Linting and Formatting

To check (and fix) code:

```bash

$ yarn lint

$ yarn lint-fix

```

To check (and fix) formatting of MD, JSON, \_and\_ code:

```bash

$ yarn prettier-check

$ yarn prettier-fix

```

We also have a simple one-liner for running both of these fix-checks back-to-back:

```bash

$ yarn format

```

Note that there is duplication for JS code in `prettier` doing the same style changes,

but both should be harmonious and run together.

### Before submitting a PR

Thanks for taking the time to help us make Spectacle even better! Before you go

ahead and submit a PR, make sure that you have done the following:

- Run all checks using `yarn check-ci`.

- Run `yarn build-one-page` and check + commit changes to `examples/one-page.html`

- Check that both the core library and \_all\_ examples build: `yarn build && yarn build-examples`.

- Update the [type definitions](./index.d.ts) for anything that modifies the Spectacle API,

like breaking changes or new features.

- Everything else included in our [pull request checklist](.github/PULL\_REQUEST\_TEMPLATE.md).

### Releasing a new version to NPM

<details>

<summary>

<i>Only for project administrators</i>

</summary>

```sh

# (1) Update the Changelog, following format from previous versions

# and commit those changes independently of other updates

$ git add CHANGELOG.md && git commit -m "Changes for v<version>"

# (2) Run tests, lint, build published dir, update package.json

$ npm version [patch|minor|major|<version>]

# (3) If all is well, publish the new version to the npm registry

$ npm publish

# (4) Then, update github with the associated tag

$ git push --tags && git push

```

Then, go and manually draft a release for your recently pushed tag with notes in

the [Github UI](https://github.com/FormidableLabs/spectacle/releases/new).

</details>

## Contributor Covenant Code of Conduct

### Our Pledge

In the interest of fostering an open and welcoming environment, we as

contributors and maintainers pledge to making participation in our project and

our community a harassment-free experience for everyone, regardless of age, body

size, disability, ethnicity, gender identity and expression, level of experience,

nationality, personal appearance, race, religion, or sexual identity and

orientation.

### Our Standards

Examples of behavior that contributes to creating a positive environment

include:

- Using welcoming and inclusive language

- Being respectful of differing viewpoints and experiences

- Gracefully accepting constructive criticism

- Focusing on what is best for the community

- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or

advances

- Trolling, insulting/derogatory comments, and personal or political attacks

- Public or private harassment

- Publishing others' private information, such as a physical or electronic

address, without explicit permission

- Other conduct which could reasonably be considered inappropriate in a

professional setting

### Our Responsibilities

Project maintainers are responsible for clarifying the standards of acceptable

behavior and are expected to take appropriate and fair corrective action in

response to any instances of unacceptable behavior.

Project maintainers have the right and responsibility to remove, edit, or

reject comments, commits, code, wiki edits, issues, and other contributions

that are not aligned to this Code of Conduct, or to ban temporarily or

permanently any contributor for other behaviors that they deem inappropriate,

threatening, offensive, or harmful.

### Scope

This Code of Conduct applies both within project spaces and in public spaces

when an individual is representing the project or its community. Examples of

representing a project or community include using an official project e-mail

address, posting via an official social media account, or acting as an appointed

representative at an online or offline event. Representation of a project may be

further defined and clarified by project maintainers.

### Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be

reported by contacting the project team at coc@formidable.com. All

complaints will be reviewed and investigated and will result in a response that

is deemed necessary and appropriate to the circumstances. The project team is

obligated to maintain confidentiality with regard to the reporter of an incident.

Further details of specific enforcement policies may be posted separately.

Project maintainers who do not follow or enforce the Code of Conduct in good

faith may face temporary or permanent repercussions as determined by other

members of the project's leadership.

### Attribution

This Code of Conduct is adapted from the [Contributor Covenant][cc-homepage], version 2.0,

available at [https://www.contributor-covenant.org/version/2/0][cc-latest-version]

<!-- Links -->

[carlos-kelly]: https://www.github.com/carlos-kelly

[cc-homepage]: http://contributor-covenant.org

[cc-latest-version]: https://www.contributor-covenant.org/version/2/0/code\_of\_conduct

[formidable-github]: https://www.github.com/FormidableLabs

[kale-stew]: https://www.github.com/kale-stew

[mdx]: https://mdxjs.com/

[spectacle-cli]: https://www.github.com/FormidableLabs/spectacle-cli

[yarn-docs]: https://yarnpkg.com/en/docs/getting-started