Version: 6.x

React Navigation contributor guide

Want to help improve React Navigation? Your help would be greatly appreciated!

Here are some of the ways to contribute to the project:

- Contributing
  - Reporting Bugs
  - Improving the Documentation
  - Responding to Issues
  - Bug Fixes
  - Suggesting a Feature
  - Big Pull Requests
- Information
  - Issue Template
  - Pull Request Template
  - Forking the Repository
  - Code Review Guidelines
  - Run the Example App
  - Run Tests

And here are a few helpful resources to aid in getting started:

- Contributing
  - Reporting Bugs
  - Improving the Documentation
  - Responding to Issues
  - Bug Fixes
  - Suggesting a Feature
  - Big Pull Requests
- Information
  - Issue Template

- Pull Request Template
- Forking the Repository
- Code Review Guidelines
- Run the Example App
- Run Tests

Contributing

Reporting Bugs

You can't write code without writing the occasional bug. Especially as React Navigation is moving quickly, bugs happen. When you think you've found one here's what to do:

1. Search the existing issues for one like what you're seeing. If you see one, add a 👍 reaction (please no +1 comments). Read through the comments and see if you can provide any more valuable information to the thread
2. If there are no other issues like yours then create a new one. Be sure to follow the issue template.

Creating a high quality reproduction is critical. Without it we likely can't fix the bug and, in an ideal situation, you'll find out that it's not actually a bug of the library but simply done incorrectly in your project. Instant bug fix!

Improving the Documentation

Any successful projects needs quality documentation and React Navigation is no different.

Read more about the documentation on the react-navigation/react-navigation.github.io repository.

Responding to Issues

Another great way to contribute to React Navigation is by responding to issues. Maybe it's answering someone's question, pointing out a small typo in their code, or helping them put together a reproduction. If you're interested in a more active role in React Navigation start with responding to issues - not only is it helpful but it demonstrates your commitment and knowledge of the code!

Bug Fixes

Find a bug, fix it up, all day long you'll have good luck! Like it was mentioned earlier, bugs happen. If you find a bug do the following:

1. Check if a pull request already exists addressing that bug. If it does give it a review and leave your comments
2. If there isn't already a pull request then figure out the fix! If it's relatively small go ahead and fix it and submit a pull request. If it's a decent number of changes file an issue first so we can discuss it (see the Big Pull Requests section)
3. If there is an issue related to that bug leave a comment on it, linking to your pull request, so others know it's been addressed.

Check out the help wanted and good first issue tags to see where you can start helping out!

Suggesting a Feature

Is there something you want to see from React Navigation? Please create a feature request on Canny.

Big Pull Requests

For any changes that will add/remove/modify multiple files in the project (new features or bug fixes) hold off on writing code right away. There's a few reasons for that

1. Big pull requests take a lot of time to review and it's sometimes hard to pick up the context
2. Often you may not have to make as big of a change as you expect

With that in mind, here's the suggestion

1. Open an issue and clearly define what it is you want to accomplish and how you intend to accomplish it
2. Discuss that solution with the community and maintainers. Provide context, establish edge cases, and figure out the design
3. Decide on a plan of action
4. Write the code and submit the PR
5. Review the PR. This can take some time but, if you followed the steps above, hopefully it won't take too much time.

The reason we want to do this is to save everyone time. Maybe that feature already exists but isn't documented? Or maybe it doesn't fit with the library. Regardless, by discussing a major change up front you're saving your time and others time as well.

Information

Issue Template

Before submitting an issue, please take a look at the issue template and follow it. This is in place to help everyone better understand the issue you're having and reduce the back and forth to get the necessary information.

Yes, it takes time and effort to complete the issue template. But that's the only way to ask high quality questions that actually get responses.

Would you rather take 1 minute to create an incomplete issue report and wait months to get any sort of response? Or would you rather take 20 minutes to fill out a high quality issue report, with all the necessary elements, and get a response in days? It's also a respectful thing to do for anyone willing to take the time to review your issue.

Pull Request Template

Much like the issue template, the pull request template lays out instructions to ensure your pull request gets reviewed in a timely manner and reduces the back and forth. Make sure to look it over before you start writing any code.

Forking the Repository

- Fork the `repo` on GitHub
- Run these commands in the terminal to download locally and install it:

```
git clone https://github.com/<USERNAME>/navigation-ex.git
cd navigation-ex
git remote add upstream https://github.com/react-navigation/react-navigation.git
yarn
```

The project uses a monorepo structure for the packages managed by yarn workspaces and lerna. All of the packages are under the packages/ directory.

Code Review Guidelines

Look around. Match the style of the rest of the codebase. This project uses ESLint to ensure consistency throughout the project. You can check your project by running:

```
yarn lint
```

If any errors occur you'll either have to manually fix them or you can attempt to automatically fix them by running:

```
yarn lint --fix
```

The codebase is written in TypeScript, and must pass typecheck. To typecheck files, run:

```
yarn typescript
```

It's useful to run typechecking in watch mode when working on the project. To do it, run:

```
yarn typescript --watch
```

Run the Example App

The example app includes a variety of patterns and is used as a simple way for contributors to manually integration test changes.

While developing, you can run the example app with Expo to test your changes:

```
yarn example start
```

Run Tests

React Navigation has tests implemented in Jest. To run either of these, from the React Navigation directory, run either of the following commands (after installing the `node_modules`) to run tests or type-checking.

```
yarn test
```

It's useful to run tests in watch mode when working on the project. To do it, run:

```
yarn test --watch
```

These commands will be run by our CI and are required to pass before any contributions are merged.

Edit this page