Next Steps

What Wasn't Covered

This course covered a lot of material and techniques, but there's a lot of things we *didn't* cover. Some examples:

- Dealing with side effects, including use of advanced side effect libraries like `redux-saga` or `redux-observable`
- Creating new entities on the client
- Working with more complex nested/relational data
- Optimizing performance
- Sorting / filtering / pagination
- Code-splitting the application, including dynamically adding reducers
- Tracing data flow and debugging techniques
- Using higher-order functions to generate action creators and reducers for similar data types

I hope to write about some of these topics on my dev blog in the future as I continue the "Practical Redux" tutorial series this course is based on. If you're interested, I keep a gist listing some of the topics I hope to write about, which also includes several other areas of interest. Keep an eye on my blog for more posts!

Further Learning Resources

Until then, I'd encourage you to check out my React/Redux links list, which points to many articles on topics like Redux Side Effects, Redux Reducers and Selectors, React/Redux Performance, and much more.

The official Redux docs are a great resource, including the Redux FAQ and the Structuring Reducers section.

If you're looking for some other "real-world" React+Redux examples, I recommend Sophie DeBenedetto's 8-part Build a Simple CRUD App with React and Redux series. If you want to look at code from existing applications, check out the "Real World" React+Redux example app, and my Redux addons catalog has a section that lists many open-source React+Redux apps, including both purpose-built examples and real applications. My links list also has a section specifically listing "Project-Based" Redux Tutorials.

I'd specifically suggest that you read some articles that go into more detail on Redux best practices and the philosophy behind Redux. First, there's my own "Idiomatic Redux" series, where I've written about what I think is the "right" way to use Redux:

- Idiomatic Redux: Why Use Action Creators? looks at reasons why I think consistent use of action creator functions is a good idea
- Idiomatic Redux: Thoughts on Thunks, Sagas, Abstraction, and Reusability lists several concerns about use of thunks and sagas, and has my responses on why I still recommend those as useful tools
- The 2-part post The Tao of Redux, Part 1 - Implementation and Intent and The Tao of Redux, Part 2 - Practice and Philosophy is a detailed look at the history and intent behind Redux's design, how it's *meant* to be used, why common usage patterns exist, and my thoughts on other ways that people sometimes use Redux.

I also highly recommend:

- Dan Abramov's classic post You Might Not Need Redux, which discusses the tradeoffs that Redux asks you to make, and the benefits you get in return
- Justin Falcone's post What's So Great About Redux?, which looks at how Redux compares with OOP behavior, and how its design can be both a strength and a weakness. Some very insightful thoughts.

Finally, come by the Reactiflux chat channels on Discord - they're a great place to hang out, ask questions, and learn.

Contact Info

I'm @acemarke on Twitter as well as Reddit and HN, and I'm @markerikson on Github and Stack Overflow.

on Github and Stack Overflow.

I'm also @acemarke in Reactiflux, and usually online there most evenings US time.

If you bought this course, I'd love to hear your feedback! Please get in touch and let me know what you thought, and anything I can do to improve things.

Thanks for choosing my Practical Redux course. Now go out there and build something awesome!