React + Redux
Managing Your React App State with Redux
Keith Orlando
Agenda

1. What is Redux?
2. Why Redux?
3. Vocabulary
4. Principles
5. The Redux lifecycle
6. Workshop: catbook integration
7. Good practices
8. Resources
9. Homework
What is Redux?
Redux is a predictable state container for JavaScript apps.
Why Redux?
“Lift up state” to App.js and pass it down as props.

GET /api/whoami
“Lift up state” to the Redux Store
Redux Vocabulary

• **Store**: the global app state
• **Actions**: objects that describe “what happened”
• **Reducers**: functions that return the next state based on an action
• **Dispatching**: triggering a state update by sending an action
Redux Principles
1) Your app state (Redux store) is the single source of truth.
2) Like in React, the state is read-only. Never mutate your state directly.
3) State changes are made with **pure functions** (no side effects).

E.g. Don’t make an API call inside of a reducer.
Redux Lifecycle

Reducer

Dispatch the action

Create a new state

Store

Subscribe to the store (you choose what data is available as props)

Action Creator

Create an action

Component (View)
Actions (objects)

describe what happened

{  
  type: string, \textcolor{red}{\textbf{REQUIRED}}  
  ...,  
}

additional data
Reducers (pure functions)

function(state, action) {
    switch (action.type) {
        case UPDATE_USER_ID:
            return Object.assign({}, state, {
                userId: action.userId,
            });
        ...
    }
}
```json
{"user: {
  userId: string,
  user: object,
},

story: {
  stories: [object],
},
}
```
Workshop: Integrating Redux into Catbook

If you plan on following along:

1. Control + C (quit) any currently running instances of catbook/react hot loader
2. Open a terminal window/tab and type:

   
   $ cd ~  (or wherever you want to clone the project)
   $ git clone https://github.com/korlando/catbook-redux
   $ cd catbook-redux
   $ npm install
   $ npm start
   $ (open the project in another terminal) npm run hotloader

Redux Good Practices
(when not to use Redux?)

• Rule of thumb: put data in the Redux store when it’s shared between two or more components.
  • E.g. You probably don’t need to put the state of an input textbox in the redux store.

• Redux and React are not highly opinionated frameworks. Pick a convention and stick to it for consistency.
  • Applies to naming conventions
  • Applies to making API calls (I prefer making an API call in componentDidMount and then dispatching an action)
Additional Resources

• Redux Docs
  https://redux.js.org

• The redux-workshop branch of my fork
  $ git reset -hard
  $ git fetch
  $ git checkout redux-workshop

• Redux Sagas for advanced asynchronous handling
  https://redux-saga.js.org

• Redux Sauce for quicker setup
  https://github.com/jkeam/reduxsauce
{ 
  user: { 
    userId: string, 
    user: object, 
  }, 
  story: { 
    stories: [object], 
  }, 
}
Thanks!

Keith Orlando

korlando@nextjump.com