33 Redux Middleware

Objectives

- Students will be able to create middleware for redux
- Students will be able to add third party middleware to redux

Logging

Wouldn't it be nice if we logged every action that happens in the app, together with the state computed after it? When something goes wrong, we can look back at our log, and figure out which action corrupted the state.

Attaching Middleware

```
import { createStore, combineReducers, applyMiddleware } from 'redux'

const todoApp = combineReducers(reducers)
const store = createStore(
  todoApp,
  applyMiddleware(logger, crashReporter)
)
```

Example Logger

```
const logger = store => next => action => {
  console.log('dispatching', action)
  let result = next(action)
  console.log('next state', store.getState())
  return result
}
```

Example Crash Reporter

```
const crashReporter = store => next => action => {
  try {
    return next(action)
  } catch (err) {
    console.error('Caught an exception!', err)
    Raven.captureException(err, {
      extra: {
        action,
        state: store.getState()
      }
    })
    throw err
  }
}
```

Example Timeout Scheduler

- Schedules actions with { meta: { delay: N } } to be delayed by N milliseconds.
- Makes `dispatch` return a function to cancel the timeout in this case.

```
const timeoutScheduler = store => next => action => {
  if (!action.meta || !action.meta.delay) {
    return next(action)
  }

  const timeoutId = setTimeout(
    () => next(action),
    action.meta.delay
  )

  return function cancel() {
    clearTimeout(timeoutId)
  }
}
```

Example Thunk

```
const thunk = store => next => action =>
  typeof action === 'function'
    ? action(store.dispatch, store.getState)
    : next(action)
```