Testing Reducers

We'll test Redux reducers in our Weather app that's built with React and Redux.

The reducer is, again, a pure function! It's quite easy to see what we need to validate actually, basically every `case` of our `switch` needs to have a test:

```
export default function mainReducer(state = initialState, action) {
  switch (action.type) {
    case 'CHANGE_LOCATION':
      return state.set('location', action.location);
    case 'SET_DATA':
      return state.set('data', fromJS(action.data));
    case 'SET_DATES':
      return state.set('dates', fromJS(action.dates));
    case 'SET_TEMPS':
      return state.set('temps', fromJS(action.temps));
    case 'SET_SELECTED_DATE':
      return state.setIn(['selected', 'date'], action.date);
    case 'SET_SELECTED_TEMP':
      return state.setIn(['selected', 'temp'], action.temp);
    default:
      return state;
  }
}
```

Let's showcase this on the `'CHANGE_LOCATION'` case, first create a `reducer.test.js` file in the `__tests__ /` directory, import the reducer and add the basic structure:

```
// __tests__/reducer.test.js
import mainReducer from '../reducer';

describe('mainReducer', function() {

});
```

The first branch of the switch statement we'll test is the `default` one – if we

don't pass any state and an empty action in it should return the initial state.

The thing is that the `initialState` is an immutable object, so we'll need to import `fromJS` too:

```
// __tests__/reducer.test.js
import mainReducer from '../reducer';
import { fromJS } from 'immutable';

describe('mainReducer', function() {
  it('should return the initial state', function() {
    expect(mainReducer(undefined, {})).toEqual(fromJS({
            location: '',
          data: {},
          dates: [],
          temps: [],
          selected: {
            date: '',
            temp: null
          }
        }));
  });
});
```

You should now see this output:

```
PASS  src/__tests__/actions.test.js (0.365s)
PASS  src/__tests__/reducer.test.js (0.215s)
13 tests passed (13 total in 2 test suites, run time 0.519s)
```

Brilliant! Let's showcase how we can test specific actions, again using our beloved `'CHANGE_LOCATION'` one.

First, add a new `it` explaining what the reducer should do:

```
// __tests__/reducer.test.js
import mainReducer from '../reducer';
import { fromJS } from 'immutable';

describe('mainReducer', function() {
  it('should return the initial state', function() {/* … */});

  it('should react to an action with the type CHANGE_LOCATION', function
() {
```

```
  });
});
```

Then, validate that the reducer changes the `location` field in the state correctly:

```
it('should react to an action with the type CHANGE_LOCATION', function() {
  var location = 'Vienna, Austria';
  expect(mainReducer(undefined, {
    type: 'CHANGE_LOCATION',
    location: location
  })).toEqual(fromJS({
    location: location,
    data: {},
    dates: [],
    temps: [],
    selected: {
      date: '',
      temp: null
    }
  }));
});
```

Now we know that our action returns an object with a `type` of `"CHANGE_LOCATION"` and that our reducer changes the `location` field in the state correctly in response to an object with a `type` of `"CHANGE_LOCATION"`! Brilliant!

Let's do the same thing for our `'SET_DATES'` case, first add the `it`:

```
// __tests__/reducer.test.js
import mainReducer from '../reducer';
import { fromJS } from 'immutable';

describe('mainReducer', function() {
  it('should return the initial state', function() {/* … */});

  it('should react to an action with the type CHANGE_LOCATION', function
() {/* … */});

  it('should react to an action with the type SET_DATES', function() {
```

```
  });
});
```

Then make sure our reducer acts accordingly:

```
it('should react to an action with the type SET_DATES', function() {
  var dates = ['2016-01-01', '2016-02-02'];
  expect(mainReducer(undefined, {
    type: 'SET_DATES',
    dates: dates
  })).toEqual(fromJS({
    location: '',
    data: {},
    dates: dates,
    temps: [],
    selected: {
      date: '',
      temp: null
    }
  }));
});
```

Not too hard, eh? That's the power of redux!

Now that we have showcased how it works with those two examples, go ahead and test the other cases too!

Done? This is what your terminal output should look like when running `npm run test -- --verbose`:

```
PASS  src/__tests__/actions.test.js (0.365s)
 actions
   changeLocation
     ✓ it should have a type of CHANGE_LOCATION (5ms)
     ✓ it should pass on the location we pass in (1ms)
   setSelectedDate
     ✓ it should have a type of SET_SELECTED_DATE (1ms)
     ✓ it should pass on the date we pass in
   setSelectedTemp
     ✓ it should have a type of SET_SELECTED_TEMP (1ms)
     ✓ it should pass on the temp we pass in
   setData
     ✓ it should have a type of SET_DATA (1ms)
     ✓ it should pass on the data we pass in
   setDates
```

```
    ✓ it should have a type of SET_DATES
    ✓ it should pass on the dates we pass in (1ms)
  setTemps
    ✓ it should have a type of SET_TEMPS
    ✓ it should pass on the temps we pass in (1ms)

PASS  src/__tests__/reducer.test.js (0.515s)
 mainReducer
  ✓ it should return the initial state (7ms)
  ✓ it should react to an action with the type 'CHANGE_LOCATION' (1ms)
  ✓ it should react to an action with the type 'SET_DATA' (3ms)
  ✓ it should react to an action with the type 'SET_DATES' (4ms)
  ✓ it should react to an action with the type 'SET_TEMPS' (2ms)
  ✓ it should react to an action with the type 'SET_SELECTED_DATE' (1ms)
  ✓ it should react to an action with the type 'SET_SELECTED_TEMP'

19 tests passed (19 total in 2 test suites, run time 0.819s)
```

If you do not have all the 7 cases in your reducer tested, go back and try to do them all! It'll strengthen your testing muscle and help you get used to thinking this way!

When your output looks like the output above, you're done! This is what your `reducer.test.js` file should look like:

```
import mainReducer from '../reducer';
import { fromJS } from 'immutable';

describe('mainReducer', function() {
    it('should return the initial state', function() {
        expect(mainReducer(undefined, {})).toEqual(fromJS({
            location: '',
          data: {},
          dates: [],
          temps: [],
          selected: {
            date: '',
            temp: null
          }
        }));
    });

    it("should react to an action with the type 'CHANGE_LOCATION'", functi
on() {
        var location = 'Vienna, Austria';
```

```
    expect(mainReducer(undefined, {
        type: 'CHANGE_LOCATION',
        location: location
    })).toEqual(fromJS({
        location: location,
      data: {},
      dates: [],
      temps: [],
      selected: {
        date: '',
        temp: null
      }
    }));
});

it("should react to an action with the type 'SET_DATA'", function() {
    var data = { some: 'data' };
    expect(mainReducer(undefined, {
        type: 'SET_DATA',
        data: data
    })).toEqual(fromJS({
        location: '',
      data: data,
      dates: [],
      temps: [],
      selected: {
        date: '',
        temp: null
      }
    }));
});

it("should react to an action with the type 'SET_DATES'", function() {
    var dates = ['2016-01-01', '2016-02-02'];
    expect(mainReducer(undefined, {
        type: 'SET_DATES',
        dates: dates
    })).toEqual(fromJS({
        location: '',
      data: {},
      dates: dates,
      temps: [],
      selected: {
        date: '',
        temp: null
      }
```

```
        }));
    });


    it("should react to an action with the type 'SET_TEMPS'", function() {
        var temps = ['31', '32'];
        expect(mainReducer(undefined, {
            type: 'SET_TEMPS',
            temps: temps
        })).toEqual(fromJS({
            location: '',
          data: {},
          dates: [],
          temps: temps,
          selected: {
            date: '',
            temp: null
          }
        }));
    });

    it("should react to an action with the type 'SET_SELECTED_DATE'", func
tion() {
        var date = '2016-02-01'
        expect(mainReducer(undefined, {
            type: 'SET_SELECTED_DATE',
            date: date
        })).toEqual(fromJS({
            location: '',
          data: {},
          dates: [],
          temps: [],
          selected: {
            date: date,
            temp: null
          }
        }));
    });

    it("should react to an action with the type 'SET_SELECTED_TEMP'", func
tion() {
        var temp = '31';
        expect(mainReducer(undefined, {
            type: 'SET_SELECTED_TEMP',
            temp: temp
        })).toEqual(fromJS({
            location: '',
```

```
          data: {},
          dates: [],

          temps: [],
          selected: {
            date: '',
            temp: temp
          }
        }));
    });
});
```

Onwards to testing our components!