Creating the store

Now that we've gotten a taste for React*, it's time to get rolling with Redux. Let's use some starters I'll provide you.

1. Create the store folder under src.
2. Add a file called store.js to it.
3. In that file create an initial state like this:

```
const initialState = {
  cart: {seats:[], food:[]},
  currentDate: new Date().setHours(0,0,0,0),
  films: [],
  reservations: [],
  showings: [],
  theaters: [],
  user: {},
};
```

4. Create the world's simplest reducer like this:

```
const reducer = (state, action) => state;
```

Next we want to create our Redux store so that it can be used in a React application.

5. At the top of store.js, import createStore from the redux library.
6. Call createStore passing in your reducer and your initialState. It will return an instance of the store. Save that to a const and export it.
7. This store will be needed in our React app, so import it at the top of App.js. If there are errors in your store.js, this is where they'll show up. Go ahead and fix any problems that show up here.
8. Let's just make sure that we've set it up properly so far. Put this code at the top of App.js:

```
import { useEffect } from 'react';
```

9. And put this just inside the App function:

```
const state = store.getState();
useEffect(() => {
  console.log("State is", state);
}, []);
```

Don't worry about the useEffect because it's unrelated to Redux. It's just a React thing that will be explained elsewhere in the course. For now, just know that this useEffect statement will run some things only the first time the component is rendered.

10. Run and test. Take a look in the browser's console and examine the state. It's pretty simple, right? Do you see some familiar things in there?

We have a store and state. It know, it's pretty lame so far but in the next few labs we'll be adding lots of capability to it. Stand by for that.

---

* Note: This lab was designed to run after the create-react-app lab. Please make sure that lab is complete before you start to work on this one.