Ajax with Redux

In this lab we're going to learn the pattern of fetching data from a server. Ajax calls deal with data manipulation and state, so they properly belong in Redux rather than in a UI like a React component. But Ajax calls are asynchronous so they must be implemented in middleware.

1. Edit App.js. Remove the raw, manual fetch of the films and replace it with a simple dispatch:

```
store.dispatch(actions.fetchFilms());
```

2. We know this won't work yet because we don't have an action for fetchFilms. Let's fix that. Add an actionType of "FETCH_FILMS". Also add an action creator of fetchFilms. (Hint: this will return a simple object with only a type. No payload needed).
3. Run and test. Your app will transpile just fine but nothing is handling an action of FETCH_FILMS so you're missing your list of films. We'll put that in a middleware.

Writing our fetch films middleware

4. Write fetchFilmsMiddleware in a new file in the middleware folder. Something along these lines will work:

```
export const fetchFilmsMiddleware = ({ getState, dispatch }) => next => action => {
  if (action.type === actionTypes.FETCH_FILMS) {
    const url = `/api/films`;
    fetch(url)
      .then(res => res.json())
      .then(films => dispatch(actions.setFilms(films)))
      .catch(err => console.error(`Error fetching films`, err));
  }
  next(action);
}
```

   (Hint: Don't forget to import actions and actionTypes).
5. Add fetchFilmsMiddleware to the exported middleware array in middleware.js.
6. Run and test. Your films should be back.

Adding some more fetches

We're fetching the films but our app will eventually need some more initial data.

7. What we did to fetch the films, do the same thing to fetch theaters.
8. Run and test. Do you see your theaters? (Hint: You may want to turn on debugging logging).
9. Now do the same for showings. Make sure you can see them all.

Awesome. We're getting lots of data but we have an opportunity for a little efficiency. Let's combine some middleware.

10. Create an actionType of "FETCH_INITIAL_DATA" and an action creator of fetchInitialData().
11. Add a fetchInitialDataMiddleware that simply dispatches actions to fetchFilms, fetchShowings, and fetchTheaters if the actionType is FETCH_INITIAL_DATA.
12. In App.js, replace your three dispatches with one simpler dispatch to fetchInitialData.
13. Run and test.

Once you're getting all the data expected you can be finished.