This app is going to be a bit plain

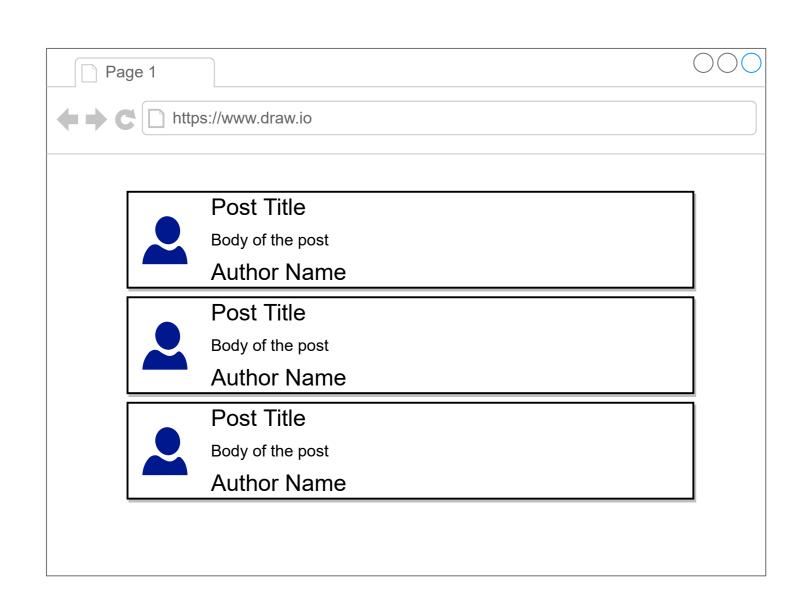
...for a very good reason!

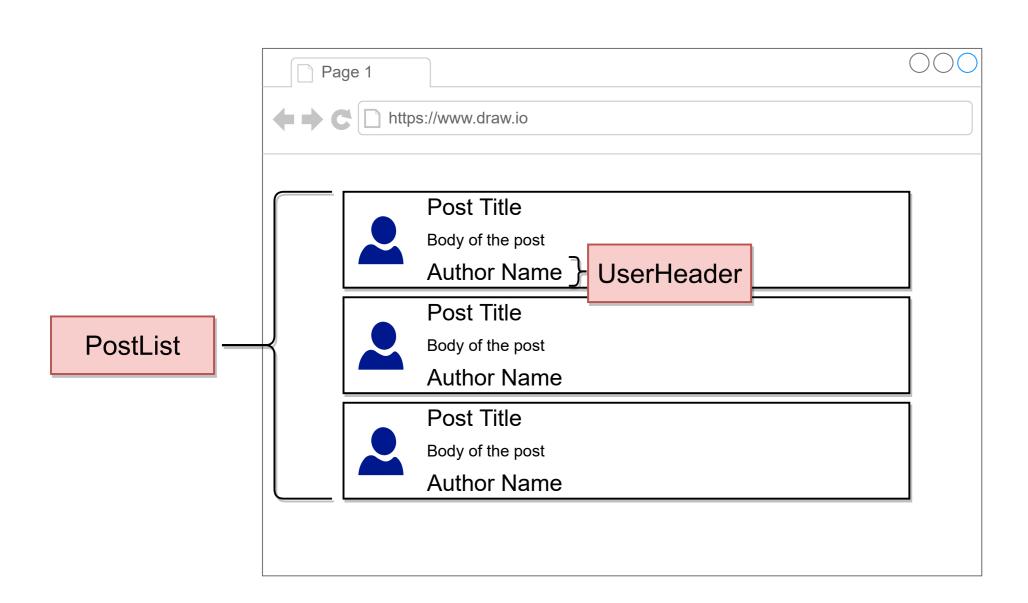
App Goals

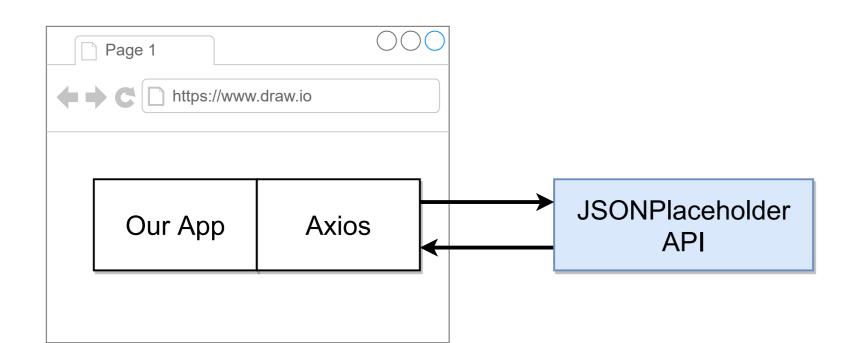
Absolutely understand the purpose of reducers

Absolutely understand making API requests with Redux

Absolutely understand the purpose of 'redux-thunk'

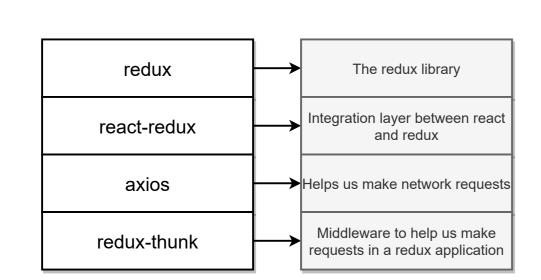






JSON Placeholder API

jsonplaceholder.typicode.com/



General Data Loading with Redux

Component gets rendered onto the screen

Component's 'componentDidMount' lifecycle method gets called

We call action creator from 'componentDidMount'

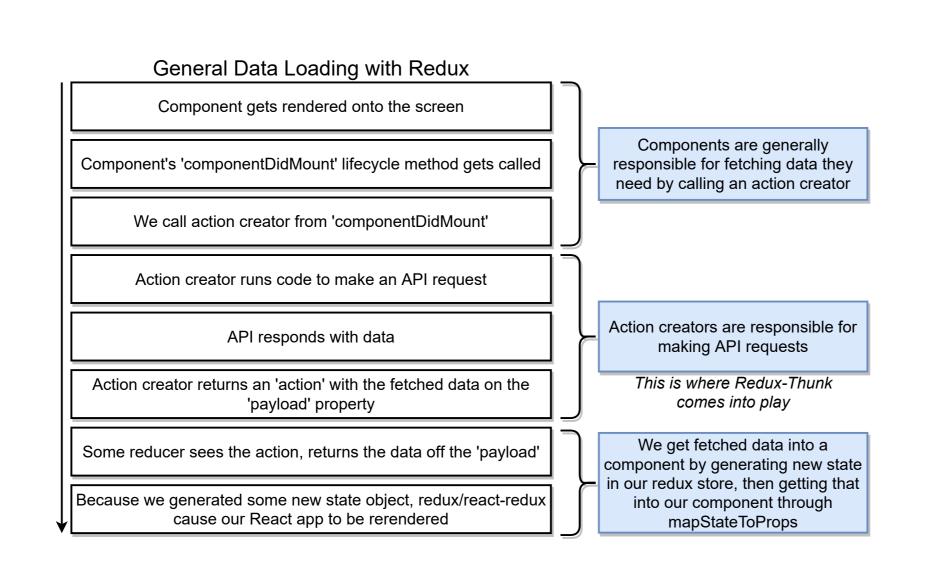
Action creator runs code to make an API request

API responds with data

Action creator returns an 'action' with the fetched data on the 'payload' property

Some reducer sees the action, returns the data off the 'payload'

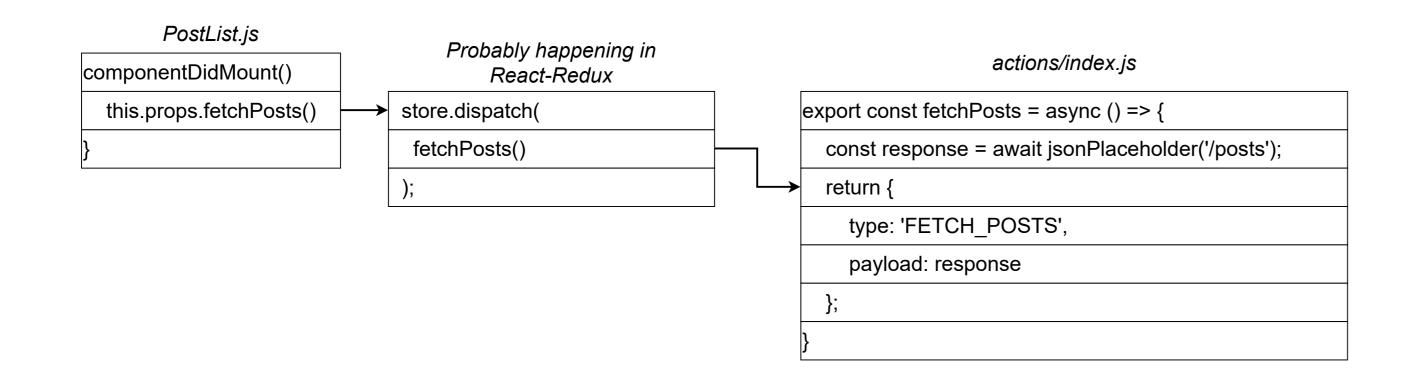
Because we generated some new state object, redux/react-redux cause our React app to be rerendered

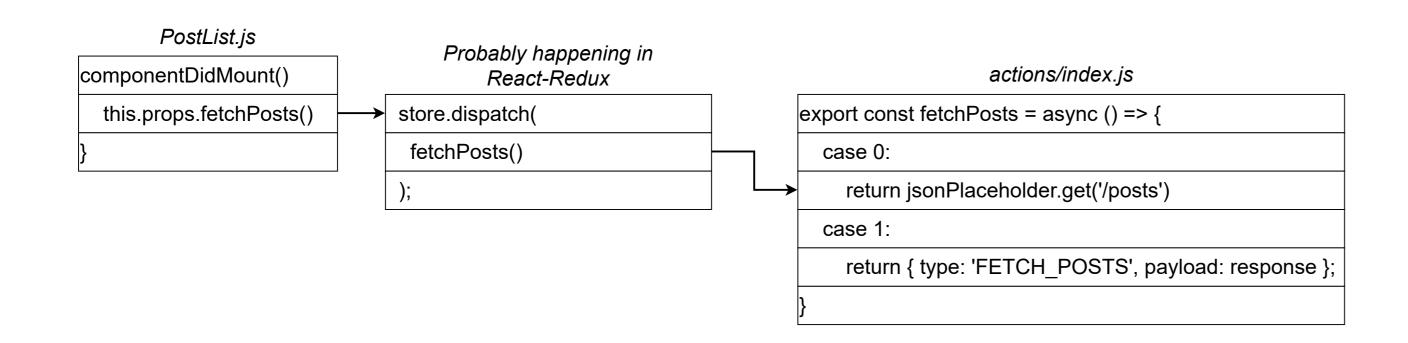


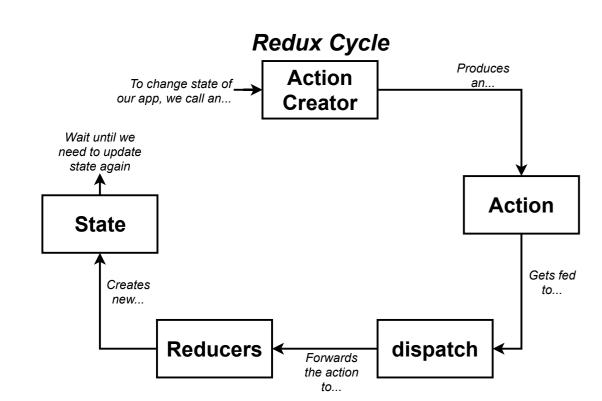
What's wrong with 'fetchPosts'?

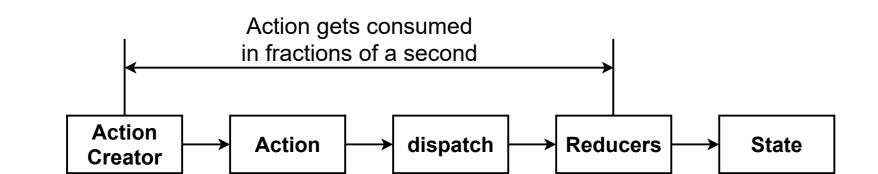
Action creators must return plain JS objects with a type property - we are not!

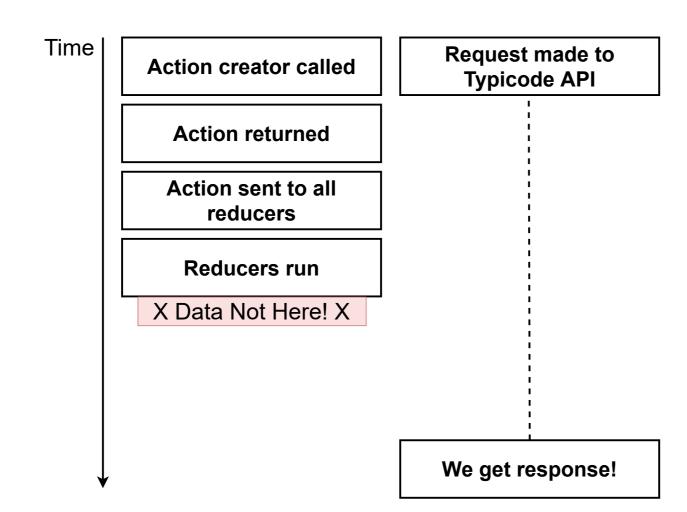
By the time our action gets to a reducer, we won't have fetched our data!









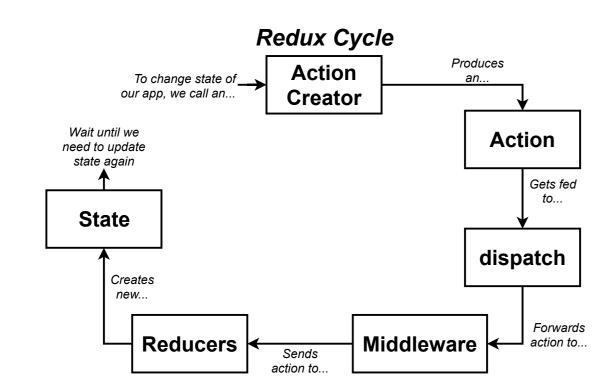


Synchronous action creator

Asynchronous action creator

Instantly returns an action with data ready to go

Takes some amount of time for it to get its data ready to go



Middleware in Redux

Function that gets called with every action we dispatch

Has the ability to **STOP, MODIFY**, or otherwise mess around with actions

Tons of open source middleware exist

Most popular use of middleware is for dealing with async actions

We are going to use a middleware called 'Redux-Thunk' to solve our async issues

Action Creators must return action objects

Actions must have a type property

Actions can optionally have a 'payload'

Action Creators can return action objects

Or

Action Creators can return functions!

If an action object gets returned, it must have a type

If an action object gets returned, it can optionally have a 'payload'

