Speak React Native

React Native course by U+_

Practical DevOps with Django

http://events.u.plus/techtea-16-practical-devops-with-django

Your homework

Overview

- Flow
- Redux

Flow

What is Flow?

- A tool for static typing - variables, functions, etc.
- Missing feature of JavaScript
- Why should we use it?
  - Error prevention
  - Code hints in IDE
  - Refactoring does not need to break everything :)

Flow basic example

```
// completely normal in JavaScript
let variable = 'string'
variable = 123
variable = {
  variableType: "object",
}

// no errors, everything's running
```

```
// @flow

// in flow, this gives an error
let variable: string = "string"
variable = 123

[flow] Cannot assign `123` to
`variable` because number [1] is
incompatible with string [2].
```

Flow basic example

```
// completely normal in JavaScript
let variable = 'string'
variable = 123
variable = {
  variableType: "object",
}

// no errors, everything's running
```

```
// @flow

// in flow, this gives an error
let variable: string = "string"
variable = 123

[flow] Cannot assign `123` to
`variable` because number [1] is
incompatible with string [2].
```

Code without Flow

```
class CustomButton extends React.PureComponent {
  render() {
    return <Button onPress={this.props.onPress}>{this.props.text}</Button>
  }
}
```

Added Flow types

```
// @flow
type Props = {
  onPress: () => void,
  text: string,
}

class CustomButton extends React.PureComponent<Props> {
  render() {
    return <Button onPress={this.props.onPress}>{this.props.text}</Button>
  }
}
```

Added Flow types

```
// @flow
type Props = {
  onPress: () => void,
  text: string,
}

class CustomButton extends React.PureComponent<Props> {
  render() {
    return <Button onPress={this.props.onPress}>{this.props.text}</Button>
  }
}
```

Added Flow types

```
// @flow
type Props = {
  onPress: () => void,
  text: string,
}

class CustomButton extends React.PureComponent<Props, State> {
  render() {
    return <Button onPress={this.props.onPress}>{this.props.text}</Button>
  }
}
```

Redux

What is Redux

- Library for holding app state
- Single source of truth for our app
- Makes it much easier to manage state, pass data between screens etc.
- Something like “database” + its management

Get skeleton with basic redux

- Branch add-redux in skeleton repo
- https://github.com/jvaclavik/speak-react-native-skeleton/tree/add-redux
- (Don’t forget to run yarn)

Or install it yourself

```
$ yarn add redux react-redux redux-logger
```

Or install it yourself

```
$ yarn add redux react-redux redux-logger


# the same as
$ yarn add redux
$ yarn add react-redux
$ yarn add redux-logger
```

Application state

Application state

- Data for your application
- Similar to component state
- Simply a big object containing our data and other state needed for the app
- App state is changing in time

Application state example

App state: Starting the app

```
{}
```

App state: Close tutorial

```
{
  app: {
    showTutorial: false,
  },
}
```

App state: Download movie list

```
{
  app: {
    showTutorial: false,
  },
  movies: {
    items: [
      { id: 1, title: "Bohemian Rhapsody" },
      { id: 2, title: "Godfather" }
    ],
  },
}
```

App state: Mark movie 1 as favorite

```
{
  app: {
    showTutorial: false,
  },
  movies: {
    items: [
      { id: 1, title: "Bohemian Rhapsody" },
      { id: 2, title: "Godfather" }
    ],
    favorites: [1],
  },
}
```

Redux basics

Reducers

- Function which takes current state, modifies it and returns a new state
- The only way to change state

```
case "ON_GET_MOVIES":
  return {
    ...state,
    items: action.movies,
  }
```

Redux actions

- Event with a name and optional payload (arguments)
- Can trigger corresponding reducer to change state
- Any number of actions can be defined in an app

```
export const onGetMovies = movies => ({
 type: "ON_GET_MOVIES",
 movies,
})
```

Implementation

Implementation

- Implement “toggle favorite” button in `Detail.js`
- Implement new action `onToggleFavorite`
- Implement reducer (case `ON_TOGGLE_FAVORITE`)

Initial State

```
// src/redux/MoviesRedux.js
export const initialState = {
  items: [],
  favorites: [],
}
```

Action

```
// src/redux/MoviesRedux.js
export const onToggleFavorite = movieId => ({
  type: "ON_TOGGLE_FAVORITE",
  movieId,
})
```

Reducer

```
// src/redux/MoviesRedux.js
case "ON_TOGGLE_FAVORITE":
  return {
    ...state,
    favorites:
      state.favorites.indexOf(action.movieId) > -1
        ? state.favorites.filter(favorite => favorite !== action.movieId)
        : [...state.favorites, action.movieId],
  }
```

Render function

```
// src/containers/Detail.js
render() {
   const { movie, onToggleFavorite, isMovieFavorite } = this.props
   return (
     <SafeAreaView style={styles.container}>
       <Text>{movie.title}</Text>
       <RoundedButton onPress={() => onToggleFavorite(movie.id)}>
         {isMovieFavorite ? "Remove" : "Add"}
       </RoundedButton>
     </SafeAreaView>
   )
 }
```

Connect function

- Connects your screens with redux
  - Access to redux state and actions in our components and screens

- We define which parts of the state we want to use and connect gives them to the component as props

Connect Detail to redux

```
// src/containers/Detail.js

import { connect } from "react-redux"
...
export default connect(
  mapStateToProps,
  mapDispatchToProps,
)(Detail)
```

Get data from redux

```
// src/containers/Detail.js
const mapStateToProps = (state, props) => {
  const { movieId } = props.navigation.state.params
  return {
    movie: getMovieById(state, movieId),
    isMovieFavorite: state.movies.favorites.indexOf(movieId) > -1,
  }
}
```

Pass action from redux

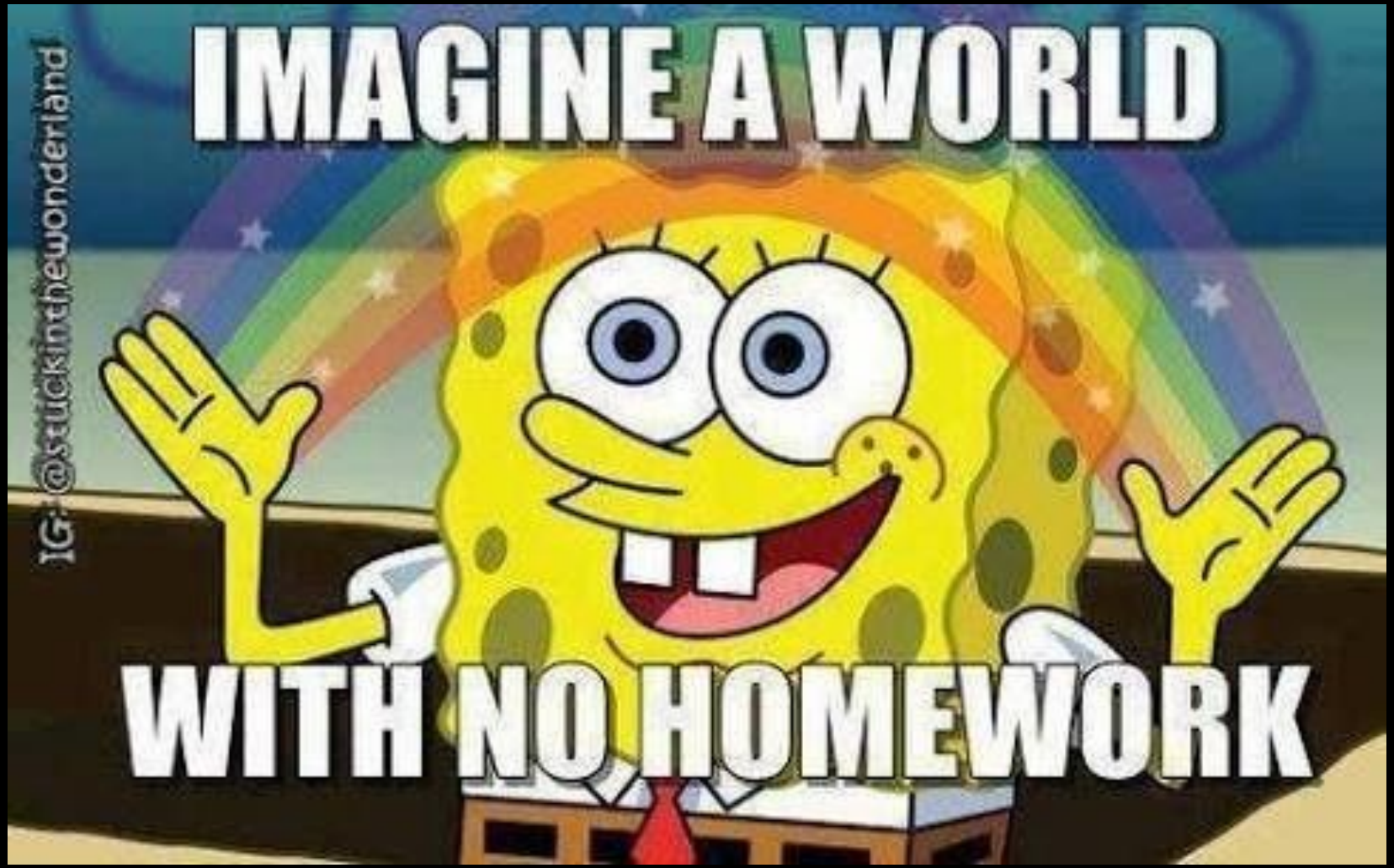
```
// src/containers/Detail.js
import { onToggleFavorite } from "../redux/MoviesRedux"

const mapDispatchToProps = {
  onToggleFavorite,
}
```

Get movie by ID

```
// src/containers/Detail.js

// For illustration purposes only (we should use selector)
const getMovieById = (state, movieId) =>
  state.movies.items.filter(movie => movie.id === movieId)[0]
```


IMAGINE A WORLD
WITH NO HOMEWORK
IG: @stuckinthewonderland

Projects

Questions?