React & Redux Course Overview

Overview

React is a JavaScript library for building user interfaces. It is an open source library and is developed by Facebook. It can be used to create both web and mobile apps. React apps are fast, flexible, modular and scalable. React has a robust developer community that's rapidly growing.

Redux is a JavaScript library used to manage an application's front-end state. As single-page apps become more complex, the need for proper state management has become more important than ever. The main benefits of using Redux come in when your app involves shared state. It enables two or more components to share same piece(s) of application state. Redux not only allows for an organized way to store data, it also gives us the ability to quickly obtain that data anywhere within the app.

The goal of this course is to equip you with the skills and experience you'll need to become a professional React developer. It will teach the core knowledge you need to deeply understand and build React components and structure applications with Redux.

We'll start by mastering the fundamentals of React, including JSX, props, state, and events. After an introduction to React, we'll dive right in to Redux, covering topics like reducers, actions, and the state tree.

Participants' Profile

For this course, you need to have experience with building front-end web applications with:

- HTML & CSS
- JavaScript
- Bootstrap (is a plus)

Benefits

At the end of this course, the participant will:

- Build amazing single page applications with React & Redux
- Learn and apply fundamental concepts behind structuring Redux applications
- Realize and appreciate the power of building composable components
- Understand and use NPM, Babel and ES6 JavaScript syntax

Topics Covered

1. ES6
    1.1. Introduction
    1.2. Array helper methods
    1.3. Let and Const
    1.4. Template literals
    1.5. Arrow functions

1.6. Default function parameters
1.7. Rest and Spread
1.8. Destructuring
1.9. Classes
1.10. Promises
1.11. Fetch

2. Fundamentals of React
   2.1. Why React?
   2.2. Rendering UI
   2.3. Components
   2.4. Component Lifecycle Events
   2.5. Forms
   2.6. Component Interaction
   2.7. React Router
3. Introduction to Redux
   3.1. Why Redux?
   3.2. Core Concepts of Redux
4. React & Redux
   4.1. The react-redux Node package
   4.2. Provider component
   4.3. Connecting React components with Redux store
   4.4. Reducer Composition
   4.5. Normalization – Points to keep in mind when designing a Redux store
   4.6. Redux Middleware

Software Requirements

1) Node.js (https://nodejs.org/en/)
   a) Required for installing JSON Server and Create React App CLI mentioned below
2) Create React App (https://github.com/facebookincubator/create-react-app)
   a) A command line interface for creating and managing React applications
3) JSON Server (https://www.npmjs.com/package/json-server)
   a) Allows us to expose JSON data as REST API
   b) This is required for demonstrating client-server communication
   c) Install it globally using "npm install -g json-server" command
   d) Check the URL for more information
4) Code Editor (any one)
   a) Visual Studio Code (https://code.visualstudio.com/)
   b) Sublime Text (https://www.sublimetext.com/)
   c) Brackets (http://brackets.io/)
   d) Atom (https://atom.io/)
5) Browser - Google Chrome

a) Preferred because of easier debugging

Detailed Content

1. ES6
   - 1.1. Introduction
   - 1.2. Array helper methods
   - 1.3. Let and Const
   - 1.4. Template literals
   - 1.5. Destructuring
   - 1.6. Rest and Spread operator
   - 1.7. Arrow functions
   - 1.8. Default function parameters
   - 1.9. Classes
   - 1.10. Promises and Fetch
2. Fundamentals of React
   - 2.1. Why React?
     - 2.1.1. Introduction
     - 2.1.2. Composition
     - 2.1.3. Declarative programming
     - 2.1.4. Unidirectional data flow
     - 2.1.5. React is just JavaScript
       - 2.1.5.1. Array's map() method
       - 2.1.5.2. Array's filter() method
   - 2.2. Rendering UI
     - 2.2.1. Setting up React – Create React App
     - 2.2.2. Creating elements
     - 2.2.3. Introducing JSX
   - 2.3. Components
     - 2.3.1. Functional components
     - 2.3.2. Class components
     - 2.3.3. Passing data with props
     - 2.3.4. Rendering a component
     - 2.3.5. Composing components
     - 2.3.6. Managing State
       - 2.3.6.1. Adding state to a component
       - 2.3.6.2. Update state with setState()
   - 2.4. Component Lifecycle Events
     - 2.4.1. componentWillMount()
     - 2.4.2. componentDidMount()
     - 2.4.3. componentWillUnmount()
     - 2.4.4. componentWillReceiveProps()