Course Outline

React and Redux

Course Summary

Description

React makes creation of beautiful and capable web applications possible with much less work. It allows software developers to write ultra-modern web-based programs as part of large teams. Redux makes the management of web data extremely controllable. The combination of the two is powerful!

Using short, to-the-point TED-style lectures, this course leads you through both of these JavaScript libraries from scratch. We'll lay a foundation of the tooling for both libraries. Then we launch into an understanding of Redux from the ground-up, moving all the way into advanced topics like middleware and asynchronous Ajax calls through dispatchers. Finally, we'll learn how React works and how to integrate Redux into it.

We will get you up and running rapidly, preparing you for real-world application development with a deep understanding of React components and well-structured applications. If you are new to React and Redux, or if you've been working with it but not quite getting how it really works, this is the course for you!

Objectives

After taking this course, students will be able to:

- Create a new React application from scratch.
- Use the tooling to run and debug that application
- Simplify complex state management with the Redux library
- Use best practices for reducers, dispatchers, and action creators
- Explain and leverage Redux reducer composition
- Understand when, how, and why to use Redux middleware
- Handle asynchronous operations with Redux -- including Ajax
- Understand React's custom markup language, JSX
- Design and create custom React components using the best practices
- Know the best way to apply React hooks
- Work with React props and state
- Use forms with React
- Configure and use the React Router to create robust navigation schemes

Topics

- Course Overview
- React Introduction
- create-react-app
- Redux intro
- Creating the store
- State and Subscriptions
- Actions and Reducers 101
- Advanced Actions
- Reducer composition
- Redux Middleware
- Ajax with Redux
- Redux-thunk (Time permitting)
- Redux-saga (Time permitting)
- How to create components
- How to display HTML
- How to display images
- How to handle events
- How to style with CSS
- How to work with JS libraries
- How to display data
- How to style with inline styles
- How to use state
- How to run tasks around render
- How to conditionally display
- How to display arrays
- How to compose and decompose
- How to pass data down
- How to lift state up
- How to run functions in JSX
- How to handle forms
- How to do simple routing
- How to read route parameters
- How to create hyperlinks
- How to navigate imperatively
- How to use inclusive routing
- TypeScript in React (Time permitting)

React and Redux

Course Summary (cont'd)

Audience

Experienced web developers who want to be equipped to handle large-scale web applications.

Prerequisites

A very strong grasp of advanced JavaScript. Please ask about our 5-day JavaScript course which will prepare you for the high-level of JavaScript fluency needed for this course. Very solid HTML5 and CSS knowledge is helpful

Duration

Five Days

Course Outline

React and Redux

Course Outline

I. Course Overview

II. React Introduction
- A. What is React?
- B. Its origins WRT Angular
- C. React's 3 design principles
- D. Composition of components
- E. How is React so darned fast?
- F. The virtual DOM
- G. One-way data flow
- H. How a React app works - a peak under the covers
- I. Transpiling, bundling, minifying with webpack

III. create-react-app
- A. The need for CRA - webpack, npm, babel, JSX, eslint, es2015, jest, etc
- B. Using npx
- C. The only build dependency needed
- D. Watch mode
- E. Linting code
- F. Ejecting
- G. Running unit tests

IV. Redux intro
- A. The single responsibility principle
- B. Why Redux?
- C. Why Redux with React?
- D. The 4 concepts of Redux
- E. Composition
- F. Immutability
- G. The parts of Redux
- H. The big picture of Redux

V. Creating the store
- A. Installing and including Redux
- B. The createStore() method
- C. The simplest possible store
- D. Redux ducks

VI. State and Subscriptions
- A. Single source of truth
- B. Initializing state
- C. What goes in state and what should not
- D. Why subscriptions
- E. How to subscribe in Redux

VII. Actions and Reducers 101
- A. Actions are objects
- B. The shape of actions
- C. Type and payload
- D. Reducers are functions
- E. Why we have them
- F. The shape of a reducer
- G. Avoiding the worst Redux rookie mistakes
- H. Actions and Reducers
- I. Action Creators in a React Application
- J. Dispatching Actions
- K. Mapping Actions to prop Names

VIII. Advanced Actions
- A. Action constants
- B. Action type enumerations
- C. Action creators
- D. Action creator enumerations

IX. Reducer composition
- A. The problem: complex state => complex reducers
- B. The solution: Create state slices
- C. Creating reducers to handle slices
- D. Combining reducers with Redux's built-in combineReducers
- E. Doing it manually
- F. Why manually is the right way

X. Redux Middleware
- A. The Open-Closed Principle
- B. Introduction to middleware
- C. The next() function
- D. The required shape of middleware
- E. Middleware's super-powers
- F. Recipes and examples
- G. Why you must register middleware and how to do it

XI. Ajax with Redux
- A. The problem with async calls in Redux
- B. The trick to making a good middleware function
- C. Registering the middleware
- D. Dispatching an Ajax call
- E. Making RESTful API calls
- F. How to process them into redux and then into React

React and Redux

Course Outline (cont'd)

XII. Redux-thunk (Time permitting)
- A. The need for a thunk
- B. How redux-thunk fits that need
- C. Installing and using redux-thunk properly
- D. Dispatching Ajax calls with redux-thunk

XIII. Redux-saga (Time permitting)
- A. Why Redux-saga?
- B. Saga vs. Thunk
- C. Handling side effects with saga

XIV. How to create components
- A. 3 simple steps to creating a component
- B. Hosting a component

XV. How to display HTML
- A. A gentle intro to JSX
- B. The 7 rules of JSX
- C. When JSX deviates from the W3C standards

XVI. How to display images
- A. Dynamic images for flexibility
- B. Bundling images for performance

XVII. How to handle events
- A. React's synthetic events
- B. Passing values to the handler
- C. Creating your own custom events

XVIII. How to style with CSS
- A. Styling with CSS
- B. Bundling CSS files for speed
- C. Using npm libraries

XIX. How to work with JS libraries
- A. How JS module loading works
- B. The two ways of importing JS libraries - when to use each

XX. How to display data
- A. Demystifying expressions - how to think about them
- B. Nesting JSX in expressions and vice-vers

XXI. How to style with inline styles
- A. Specifying inline styles
- B. Importing external styles

XXII. How to use state
- A. Defining state in React
- B. React state != Redux state
- C. the useState() hook
- D. Single values vs. Objects
- E. Lazy initial state
- F. Using spread to append

XXIII. How to run tasks around render
- A. The useEffect() hook
- B. Simulating lifecycle events
- C. Component loading, rendering, and unloading events

XXIV. How to conditionally display
- A. Why we need conditionals
- B. Using short-circuiting
- C. Using && and ||
- D. Ternaries

XXV. How to display arrays
- A. Iterating collections
- B. Array.prototype methods
- C. .map()
- D. .filter()
- E. What does key do for me

XXVI. How to compose and decompose
- A. Composition deep dive
- B. The reason for composition

XXVII. How to pass data down
- A. Data flowing down - props
- B. Why not 2-way binding?
- C. State vs props
- D. The useContext() hook

XXVIII. How to lift state up
- A. Why lift state up? What it is.
- B. Examples of lifting state up
- C. Data flowing up by emitting an event
- D. Passing data between sibling and cousin components
- E. All the ways to communicate and the cleanest solution

React and Redux

Course Outline (cont'd)

XXIX. How to run functions in JSX
- A. Rules for calling functions in JSX
- B. The right time to run a function

XXX. How to handle forms
- A. Forms in React
- B. Controlled vs. Uncontrolled forms
- C. How to solve the update problem
- D. The tricks handling <select>s and <textarea>s.

XXXI. How to do simple routing
- A. How to create a SPA in React
- B. Where React Router came from
- C. 4 easy steps to routing
- D. How to define the domain of a router
- E. How to create route

XXXII. How to read route parameters
- A. The useParams() hook
- B. Creating route parameters
- C. Reading parameters
- D. How to make them optional
- E. How to process querystrings

XXXIII. How to create hyperlinks
- A. How to route users via a link
- B. Why not us <a>?
- C. The <Link> component

XXXIV. How to navigate imperatively
- A. How to route users via JavaScript
- B. The useHistory() hook
- C. The push() and replace() methods
- D. Using the <Redirect> component

XXXV. How to use inclusive routing
- A. The power of inclusive routing
- B. Switch component
- C. exact parameter

XXXVI. TypeScript in React (Time permitting)
- A. Why TypeScript? Why not?
- B. Creating a React app with TS
- C. Upgrading an existing app with TS