the shortcut

React Tutorial
Lesson 7

©2020 The Shortcut. All rights reserved

©2020 THE SHORTCUT

Today's lesson will cover

- Integration Testing: Component
- Deploy React App
- React Router Introduction
- React Router Basics
- Nested Routing
- Protecting Routes

Intro to React

©2020 The Shortcut. All rights reserved

©2020 THE SHORTCUT

Integration Testing: Component

- React Testing Library adheres to a single core philosophy: instead of testing implementation details of React components, it tests how users interact with the application and if it works as expected. This becomes especially powerful for integration tests.
- There may be some confusion about when to use getBy or the queryBy search variants. As a rule of thumb, use getBy for single elements, and getAllBy for multiple elements. If you are checking for elements that aren't present, use queryBy (or queryAllBy).
- React Testing Library with Jest is the most popular library combination for React testing. RTL provides relevant testing tools, while Jest has a general testing framework for test suites, test cases, assertions, and mocking capabilities. If you need an alternative to RTL, consider trying Enzyme by Airbnb.

©2020 THE SHORTCUT

Deploy React App

- During production stage by hosting it on a remote server is called deployment
- Optimizations and packaging, also called bundling, comes with the build tools in create-react-app. Before deploying to your host of choice, you would run the following command from command line:
    - `npm run build`
- Running the build command creates a build folder containing files that are hosted the remote server to deploy on live website
- You can use your app to follow along or clone this repository: https://github.com/itistheshortcut/react-app-deploy

©2020 THE SHORTCUT

React Router Introduction

- React Router is the de facto standard third-party routing library for React.
- Routing is the process of keeping the browser URL in sync with what's being rendered on the page. React Router lets you handle routing *declaratively*. The declarative routing approach allows you to control the data flow in your application, by saying "the route should look like this":
  - `<Route path="/about" component={About} />`
- You can place your `<Route>` component anywhere you want your route to be rendered. Since `<Route>`, `<Link>` and all the other React Router APIs that we'll be dealing with are just components, you can easily get used to routing in React.
- The React Router library comprises three packages: `react-router`, `react-router-dom`, and `react-router-native`. `react-router` is the core package for the router, whereas the other two are environment specific. You should use `react-router-dom` if you're building a website, and `react-router-native` if you're on a mobile app development environment using React Native.
- Use npm to install `react-router-dom`:
  - `npm install --save react-router-dom`

©2020 THE SHORTCUT

React Router Basics

- You need a router component and several route components to set up a basic route.
- Commonly for building a browser-based application, we can use two types of routers from the React Router API:
  - `<BrowserRouter> :` `http://example.com/about`
  - `<HashRouter>:` `http://example.com/#/about`
- The `<BrowserRouter>` is more popular amongst the two because it uses the HTML5 History API to keep track of your router history.
- `history` is a JavaScript library that lets you easily manage session history anywhere JavaScript runs. Each router component creates a history object that keeps track of the current location (`history.location`).
- The `<Route>` component is the most important component in React router. It renders some UI if the current location matches the route's path.
- The `<Link>` component, on the other hand, is used to navigate between pages. It's comparable to the HTML anchor element.

©2020 THE SHORTCUT

Nested Routing

- To create nested routes, we need to have a better understanding of the three props that `<Route>` use to define what gets rendered:
  - `component`: When the URL is matched, the router creates a React element from the given component using `React.createElement`.
  - `render`: This is handy for inline rendering.
  - `children`: The children prop is similar to render in that it expects a function that returns a React element.
- The path is used to identify the portion of the URL that the router should match. If the router's path and the location are successfully matched, an object is created and we call it the match object. The match object carries more information about the URL and the path which is accessible through its properties:
  - `match.url`. A string that returns the matched portion of the URL. This is particularly useful for building nested `<Link>`s
  - `match.path`. A string that returns the route's path string — that is, `<Route path="">`. We'll be using this to build nested `<Route>`s.
  - `match.isExact`. A boolean that returns true if the match was exact (without any trailing characters).
  - `match.params`. An object containing key/value pairs from the URL parsed by the Path-to-RegExp package.
- `<Switch>` component comes in handy when multiple `<Route>`s are used together, all the routes that match are rendered inclusively.

©2020 THE SHORTCUT

Protecting Routes

- `<Redirect>` will replace the current location in the history stack with a new location. The new location is specified by the `to` prop. So, if someone for instance tries to access the `/admin` while logged out, they'll be redirected to the `/login` route.
- A custom route is a fancy word for a route nested inside a component. If we need to make a decision whether a route should be rendered or not, writing a custom route is the way to go.

the shortcut

Self Study Assignments

©2020 The Shortcut. All rights reserved

©2020 THE SHORTCUT

To Dos

- Unless you have your own project to work on to practice what we have learned, complete the Pokemon API app with React from here: https://learn.chrisoncode.io/webinars/javascript-to-react
- Start working on React or Fullstack JavaScript project, OR complete FCC Frontend Library Certification : Bootstrap, React, Redux, React & Redux, projects; APIs and Microservices Certification : https://www.freecodecamp.org/learn
- Refer to Reactjs official documentation as the best place to learn underlying main concepts of React with examples: https://reactjs.org/docs/hello-world.html