React Button Components

Zack Snyder
2/10/17

To View

Requirements**:**

- Node.js must be installed globally on your machine

To start server:

- Clone this repo to your machine
- Navigate to the cloned directory via the command line
- Install dependencies via 'npm install'
- Start the server via 'npm start'
- View in browser at localhost:3000 (I included dummy data in index.jsx)

Technologies Employed

- Basic Server: Node/Express
- Frontend Framework: React
- Transpiler/Bundler: Webpack (with Babel)

Design Choices

In general, my goal was to create clean and reusable components with propType validation while maintaining separation of concerns and a DRY codebase. I employed Webpack Dev Middleware to increase productivity; though if I were shipping this product I would take additional steps to ensure the code and environment were production-ready.

Given a group of radio buttons and a group of check buttons employ very different logic, I decided to add logic to the ButtonGroup component that will render either a RadioButtonGroup component or a CheckButonGroup component depending on the value of the 'multiple' prop. I added additional logic to ButtonGroup in order to introduce the 'All' and/or 'None' option(s) to the options prop to avoid adding duplicate logic in the RadioButtonGroup and CheckButtonGroup components. I assumed the 'All' and 'None' buttons should have the same name attribute as the other options.

I made the assumption that a RadioButton or CheckButton will never be rendered outside of their button group. As such, I designed RadioButtonGroup and CheckButtonGroup as class components while RadioButton and CheckButton are stateless functional components. By doing so I can handle all the logic to determine which button(s) should be checked at a given point in time in the onChange method of the RadioButtonGroup and CheckButtonGroup components. I added logic to the constructor methods of RadioButtonGroup and CheckButtonGroup to determine

which buttons should be checked upon initial render, if any. For both types of button groups, the 'All' option takes precedence over the 'None' option. The state objects of the two button group components contain a set of key/value pairs where each key is the 'value' attribute of an option and the associated value is a boolean representing whether or not that option is currently checked. While robust, I admit the code in the onChange method of CheckButtonGroup isn't the cleanest. I also made sure to use the 'value' attribute of each option as the key prop for each button instance to take advantage of React's reconciliation structure.

With the exception of the 'type' prop, the RadioButton and CheckButton components are identical. Thus, I created an additional component called ButtonWithLabel that accepts a 'type' prop and renders the appropriate input element. By doing so, I created a reusable button component and avoided writing duplicate logic in the RadioButton and CheckButton components. Note the input tag in ButtonWithLabel is wrapped inside the label tag to avoid adding an unnecessary container around the two.