Designing a Context Menu System

Now that we've seen how to build a modal dialog system, we can apply the same principles to context menus. **A context menu is really just another modal that's probably absolutely positioned on screen, doesn't have a dimmer overlay behind it, and contains just a menu instead of a titlebar, content area, and action buttons.**

This might be a good time to go back and review a couple of the concepts around how modals in React can actually get shown on screen.

React Portals

We start our React applications by calling `ReactDOM.render(<App />, rootElement)`. All of the nested HTML elements created by our React components are appended inside of that root element, in a single render tree. However, this can make showing modals a little bit awkward. especially if a very deeply nested child component wants to show a modal. That nested component can render a `<Modal open={true} />`, but now the HTML generated by the `Modal` component is going to be appended inside of the nested component. That means it probably won't show up correctly on top of the rest of the UI.

Now, sure, we can do some funky CSS stuff and make those elements pop out somehow, but there's a specific technique that's commonly used to make modals in React show up overlaid on the page contents. That technique is called a "portal". **A "portal" is when a React component uses its lifecycle methods to start a second React render tree, usually appended to the page body. That way a nested component can render a `<Modal>`, but the modal content pops out on top of the page.**

React 15 had support for this with a semi-official method called `ReactDOM.unstable_renderSubtreeIntoContainer`. However, React 16 made this official with a new method called `ReactDOM.createPortal()`. See the React docs

official with a new method called `ReactDOM.createPortal()`. See the React docs section on portals for more info on how that works.

There's a very useful library called react-portal that helps wrap up this capability to make it easier to use. Version 3 supports through React 15, and it looks like version 4 is a rewrite to support React 16 and clean up other aspects of the implementation.

The Semantic-UI-React `Modal` component makes use of React's portal capabilities as well, and it's abstracted out into a `<Portal>` component. (At the time of writing, it looks like SUI-React 0.76 is compatible with React 16 in general, although the `<Portal>` component still uses `unstable_renderSubtreeIntoContainer`. A look at React 16's source shows that function still exists, although it will likely be removed at some point.)

While we could use SUI-React's `<Portal>` component for our context menus, we'll use `react-portal` instead (largely because that's how I already had this code written.)