React Fundamentals Short recap - day #1

Peter Kassenaar –
info@kassenaar.com

Day #1

- Introduction – overview of the front-end landscape
- React tooling – installation
  - Node.js
  - Create React App
- The structure and architecture of React apps.
  - General structure, adding components, loading components, adding functionality
- Class based vs. Function based components
- state and props in components

Front-end Frameworks – the big four

React

3 Basic principles in every React App

1. Components
2. Reactive Updates
3. Virtual DOM in memory

Important Files

- `Package.json`
- `Index.js`
- `App.js`

Starting your app

`npm start` (`react-scripts start`) does a lot of stuff for you under the covers:

- Kicks of Babel for transpiling JSX
- Compile SASS to CSS
- Webpack for bundling files
- Sets up webserver at http://localhost:3000/
- Sets up Live Reload
- …

Principles we covered:

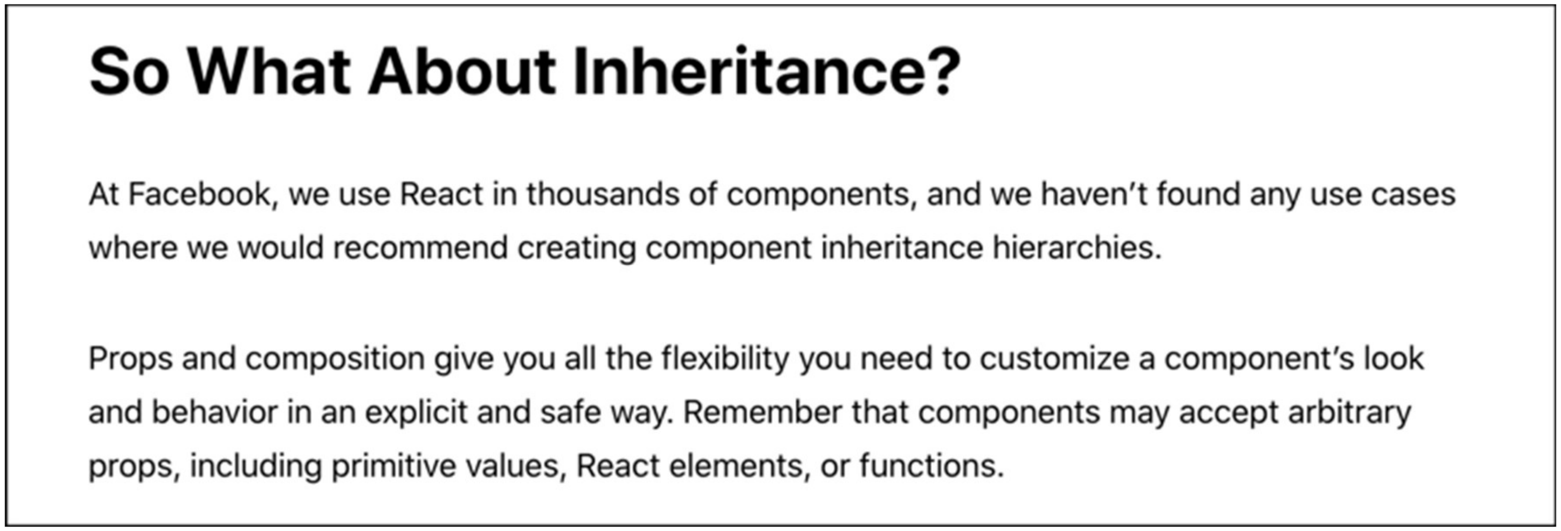
- JSX (looks like HTML, but isn't)
- State
  - `const [someVar, setSomeVar]` in functions
  - `state={…} and setState()` in classes
- "Lifting state up"
- Passing props: variables and functions
  - Inline functions to trigger state changes (`<button onClick="() => someFunction()">`)

Yesterday:

“Using class components if inheritance is needed?”

Facebook: *“use composition over inheritance”*

https://reactjs.org/docs/composition-vs-inheritance.html



So What About Inheritance?

At Facebook, we use React in thousands of components, and we haven’t found any use cases where we would recommend creating component inheritance hierarchies.

Props and composition give you all the flexibility you need to customize a component’s look and behavior in an explicit and safe way. Remember that components may accept arbitrary props, including primitive values, React elements, or functions.

Using bootstrap.js?

- We also need `popper.js` and `jquery.js`
  - `npm install popper.js jquery.js`
- After that, just import in `main.js`
- Use Bootstrap-examples as described…
- Beware! We now have *two* libraries updating the DOM!

```
<div className="dropdown">
   <button className="btn btn-secondary dropdown-toggle" type="button" id="dropdownMenuButton"
         data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
      Dropdown button
   </button>
   <div className="dropdown-menu" aria-labelledby="dropdownMenuButton">
      <a className="dropdown-item" href="#">Action</a>
      <a className="dropdown-item" href="#">Another action</a>
      <a className="dropdown-item" href="#">Something else here</a>
   </div>
</div>
```

Other

Questions?

Today:

- Loading external data and looping over data collections with `.map()`
- Binding to images
- Conditional rendering
- Lifecycle hooks
- Styling components
- Forms and handling user input

Tomorrow

- Wrapping up Forms (submitting)
- Using Http and external API’s – complete Apps
- React Router
    - Basic routing
    - Routing parameters
- useEffect() hook
- Creating your own hooks
- Redux - introduction
- Deployment