**React notes 2023K**

**CRA**

Development Server, Babel, Webpack

(npm run eject – would remove the CRA tools hooking and make a normal npm project where you would need to do all changes to tool config etc.)

**Babel – JSX – JavaScript**

Babel transpiles JSX markup to React.createElement etc. JavaScript code. But for understanding that is actually not that helpful. But explains why e.g. import React from 'react'; is needed.

**Arrow functions**

'this' inside functions. Fix. this bound to the 'window' object

arrow functions do not rebind the 'this', they e.g. inherit it in the case of class member

"fat arrow" => "fat arrow functions"

Arrow functions should not be used as:

- event-handlers (if they would need the arguments or target properties)

- constructors

- object methods

predicate function (function passed as argument to e.g. array.filter)

**Webpack**

With webpack own imports do not need the .js extension.

**Material-UI or Bootstrap**

Bootstrap also handy, but select just one. Bootstrap has its template to be downloaded.

**VS Code**

Paint code, press Alt down, move the code with arrows.

Ctrl+D to do multi-cursor editing on similar code lines

F2 to rename a symbol

.jsx extension to get better IntelliSense suggestions

prettier plugin/add-on

**React**

<React.Fragment> or <>

{this.props.children} // Component asking its (given) children to be rendered. Possible but not always used.

**React Developer Tools**

React Developer Tools (in dev tools as separate tab for Chrome). Select component to see: Key, Props, State

Console: $r (contents of the selected component) $r.render( )

(And similarly if selecting from the DOM with the Elements tab, Console: $0 or e.g. $0.click()

Try also the search feature

**State and Props**

State – internal inside the component only.

Props – set from outside by the parent, and read-only

E.g. 'Controlled component' gets props from parent

Stateless "Functional component" = Not functional=having activity for user, but functional = written with function way.  
"Presentational component" = Gets props from parent, just shows data.

**Three ways to share information**

(In addition to the original 0. state – props – props -tree = "props drilling"). In historical order:

1. Redux (its own library, concepts etc. was still used in 50% projects years time ago)
2. Context
   1. React.createContext(), UserContext.Producer/Consumer, contextType, .context
   2. useContext hook.

We can use the context to pass both data state and functions to change that data

1. Routing context. BrowserRouter provides a React context.

**Routing**

* npm i react-router-dom,
* <BrowserRouter> …whole app here… </BrowserRouter>
* <Switch> start with most specific Route, then to most generic. Only first matching is displayed
  + Though this might be old info and nowadays React router v6 is intelligently matching to correct route.
* <Link> (<NavLink> to have SPA plumbing instead of full page reload requests
* Class: Look at the React Dev tools (History, Location: e.g. query string params , Match: e.g. route params)
* Hooks: Look at the React Dev tools (History, Location: e.g. query string params , Match: e.g. route params)
* <Route path="\*" … matches with the rest!
* <Redirect from="" to="" />
* Different react-router-dom versions! E.g. 4->5->6
* (MemoryRouter is good for tests without needing Browser)
* (StaticRouter for server-side rendering)
* (NativeRouter for React Native)
* element=<MemberLayout /> for layout shared by nested routes and inside that layout component <Outlet /> for displaying each nested routes own component content.
* <Outlet context={ name: {objProperty:value1, objProperty:value2} }
* Navigate or useNavigate

Routing introduction, v6

<https://www.youtube.com/watch?v=Ul3y1LXxzdU>

* Until 16:47 the basics
* until 23:53 "basics" about nested routes
* until 28:53 some advanced topics about nested routes.
* From 32:30 or so NavLink with isActive
* 36:30 useNavigate hook
* 43:17 state={}, useLocation, location object w. e.g. "search", "hash", and "state"

**Let's look at the case for example**

* react-router-dom version? 6
* routing architecture/ navigational model?
* are routing contexts used? are react contexts used? (Redux is not used)

**React w. TypeScript notes 2023K**

How to make an existing React app or Node app to support TypeScript? Generic steps:

<https://legacy.reactjs.org/docs/static-type-checking.html#adding-typescript-to-a-project>

Starting with TypeScript fresh:

React: Either

npx create-react-app my-front-app --template typescript

or

npm create vite@latest my-react-ts-app -- --template react-ts