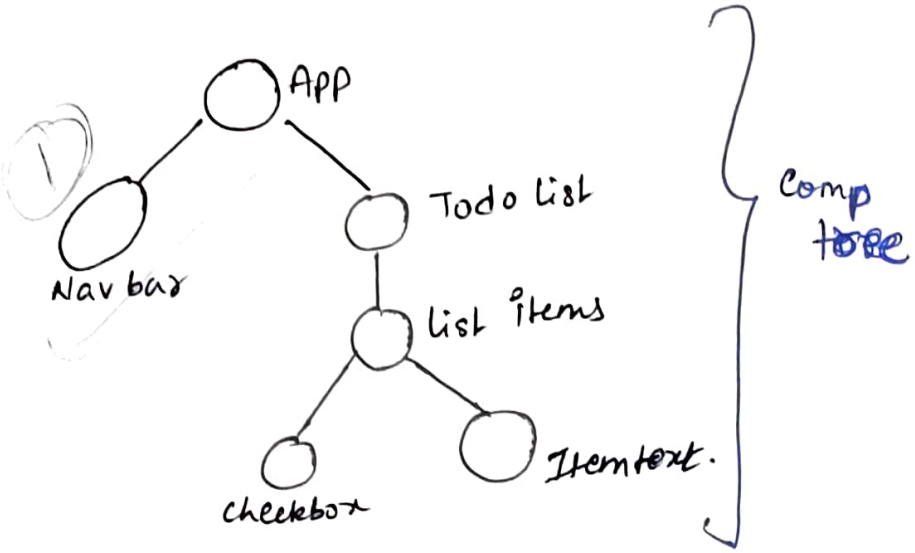
Section 33 → React.JS

① What is React? → A JS front end library to build user interfaces.

→ Breakdown things to build into component tree.



→ More clean code and simpler to debug. [Eg: of Navbar given].

→ We had refrained from the exercise of mixing html, css & JS code in the past but react benefits from this.

i.e it concentrates more on individual component than code.

and each component has it's own html, css & JS and can be seperate from other things that load & take time.

→ Compares old vs New code & compares & only changes that code, making it faster.

(3) Code Sandbox & Structure of Learning → Online editor for react is code Sandbox.

Skill → Practice → Project
Theory → editor → React.

∴ we will be using online editor initially before going to atom.

---

(4) JSX and Babel → <body>
  } ① Std in all react applications.

→ we will rarely do work in index.html, after typing std boiler plate and ①. Add React & ReactDom dependencies & require them in index.js. [** var React = require("react"); Simi for dom.]

→ Now we use render func. (Takes only single element) - ∴ div used.

* ReactDOM.render( ) ③ → three inputs (what to show, where, optional callback);

∴ ReactDOM.render (<h1> Hello world! </h1>, document.getElementById("root"));

→ we wrote something on our page without even touching html, this is possible by of JSX. & dependency (React) which has Babel (A JS compiler which converts html to JS before exciting it into web page).

→ ∴ we can use next gen code & Babel will convert it to std JS for browsers to execute easily (even for Internet expl).

→ ** Instead of that the new method is ④ [import React from "react"; import ReactDOM from "react-dom";]

⑤ JSX Code Practice

→ A basic task given to complete using React.js on what learned in prev video class.

→ If the tag is empty `<li></li>` (empty) it will turn (Sandbox) it into a self closing tag → `<li />` that does work...

⑥ JS Expressions in JSX & ES6 Template Literals →

→ JSX not only lets you insert JS and type HTML in it. It further allows you to insert more JS inside that HTML

→ You can insert JS in html by placing that code b/n { }.

But only expressions can work, not statements.

(3+4, const, var) (gives a value)

(if, else, logic) (makes comp do work.)

→ ES6 Template Literals (optional, just for reference). → Injecting strings in piece of JS → {`${fName} ${Name}`}

⑦ JSX Practice in JS inside HTML

→ Another basic task related to above content.

const currentdate = new Date()

⑧ JSX Attributes and React Styling.

- for adding styling (CSS) to HTML inside JS code we have to use 'className' not 'class' like in normal HTML. Though syntax is same.

- <script ..... text/Javascript"></script>
  ↓ change it to JSX.

- All attributes that you specify (like id, class, lang, title, etc) in JSX though might be of same name as HTML, they follow camel casing.

- Kebab casing followed in case of CSS. (img-fried-02)

- Press Alt & select 3 lines → You can type in all three lines simultaneously.

Import - - - -

const var = "Tyagi" → Even url can be given (or) something dynamic as data can be given.

ReactDOM.render(

<h1> My Name is {var} </h1>

document.getElementById("root")

);

Std procedure.

JSX is. Very strict on syntax.

---

⑨ Inline styling for React Elements.

- External styling is always recommended, but you should go through all types of stylings. (to understand others code (or) to get the feel of the loop).

» `<h1 style={{color: "red"}}>Hello!</h1>`

CSS name ↙ — ↳ in JS format — ↳ 2 brackets.

» (or) you can, ↳ use commas inst of ; (as it is JS). (gene used in CSS)

» `const customStyle = {`

Define color, etc, props here

`}`

Then write → `{customStyle}` → here

(10) **CSS Styling Practice.**

» Getting time & using if, else to say Good morning, aftern, evening. Inline CSS used to change color wrt time.

---

(11) **React Components.** ★

» We discussed component tree in the start of this module.

» Let's see what components are & how to (you) use them.

» For this we create fn (whose name should be in capital). and this fn is cut/paste into a new file made specifically for it. (The fn = component). [Eg:- Heading.jsx] ↳ we have to import react here also.

We write everything return

» Then we use import, export to connect the new file to our original file.

These 2 names need not be same but good practice.

[import Heading from "./Heading";] } both in diff files.

[export default Heading]