module-12_1

[module-12: look behind the scenes of react & optimization techniques]
a component is re-rendered only when props/state/context is changed.

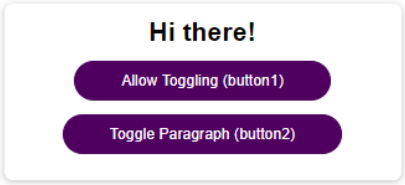
re-evaluating component !== re-rendering the DOM

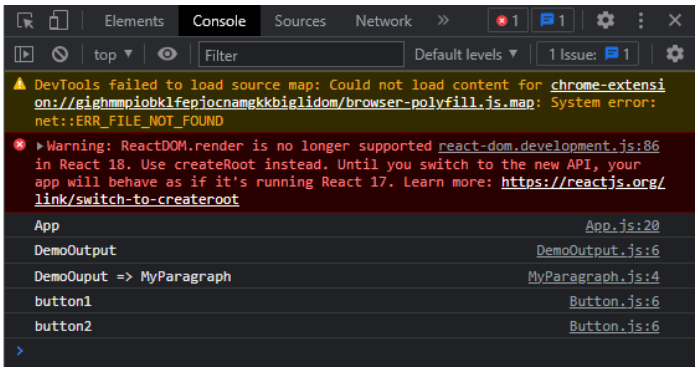
array, objects, function are non-primptive value in javascript.
string, number, boolean are primptive value in javascript.

closure??

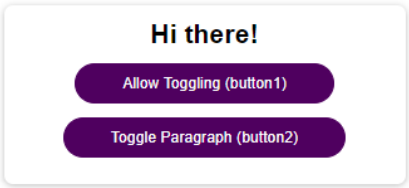
React.memo denies the re-evaluation of a component if there's no change in the component via props.
useCallback hook saves a function in a component, which optimizes the code, it also takes a dependancy.
React.memo & useCallback hook increases the optimization of our code.

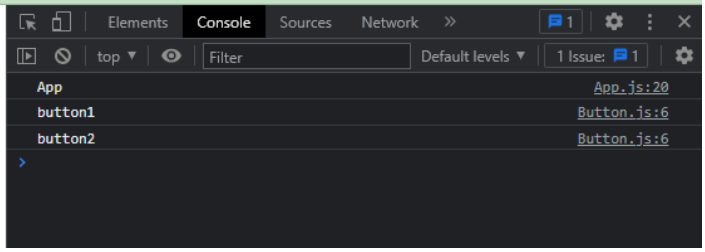
Here, React.memo(componentName) is used in Button.js
When code is executed for first time, all components are rendered.





Clicking on Allow Toggling (button 1) changes allowToggle state in App.js, so, App.js re-renders & no DemoOutput in the console.





Clicking on Toggle Paragraph (button 2) changes showParagraph state in App.js, so, App.js re-renders & shows DemoOutput, Paragraph & button1 in the console. “button1” is shown because we didn’t use any useCallback for allowToggleHandler function in App.js