React Learnings

1. Hooks
   1. useState()

useState is creates a const variable, to change its value we create a function at the time of its initialisation.

SYNTAX:

const [var,setVar] = useState(<initialValue>);

* 1. useEffect()

it helps to re-render the screen. It has many options.

1. useEffect({})

in this type, useEffect will execute on every render.

1. useEffect({}, [])

in this type, useEffect will only execute once.

1. useEffect({}, [<variable>])

in this type, useEffect will execute every time when value of <variable> is changed.

* 1. [useReducer()](https://github.com/SajalKumar2002/redux_reducer/blob/master/src/screens/UseReducerCounter.jsx)

we use useReducer to create a variable and a function to change its value just like a useState(), but the difference is that the useReducer function is created outside the component and can take values as parameters such as (state, action), the function is called dispatch and the variable is called state.

To see the example check out the link. [Click](https://github.com/SajalKumar2002/redux_reducer/blob/master/src/screens/UseReducerCounter.jsx)

To learn more about it. [Click](https://react.dev/reference/react/useReducer)

* 1. useRef()

useRef also creates a variable with data type let, it is used to give initial value to the component. For example, a clock need to be start from 00:00 seconds or click me counter will start from 0 every time page get rendered.

Note: It doesn’t cause re-renders which makes it different from useState().

SYNTAX:

const clock = useRef(0);

* 1. useContext()

To use useContext() hook we also need to 2 other hooks, useProvider() and createContext().

1. create context and export the context at last.
2. create a const provider function and a reducer function.
3. Now use useReducer() hook to create a reducer and initial value.