

ReactJS

- to run react we need that node JS must be pre installed.
- use `npm create-react-app ./` command in terminal to install reactJS
- then use "npm start" to start the local host -

Note : React JS don't Manipulate the Actual DOM instead it Manipulates the Virtual Dome which makes it Much Faster .

React Basis

=> This down below is the default script in which we are going to write our code Which are kept inside the App cosnt function

```
import './App.css';

const App = () => {
  return (
    <div className="App">
      <h1>Hello, World!!</h1>
    </div>
  );
}

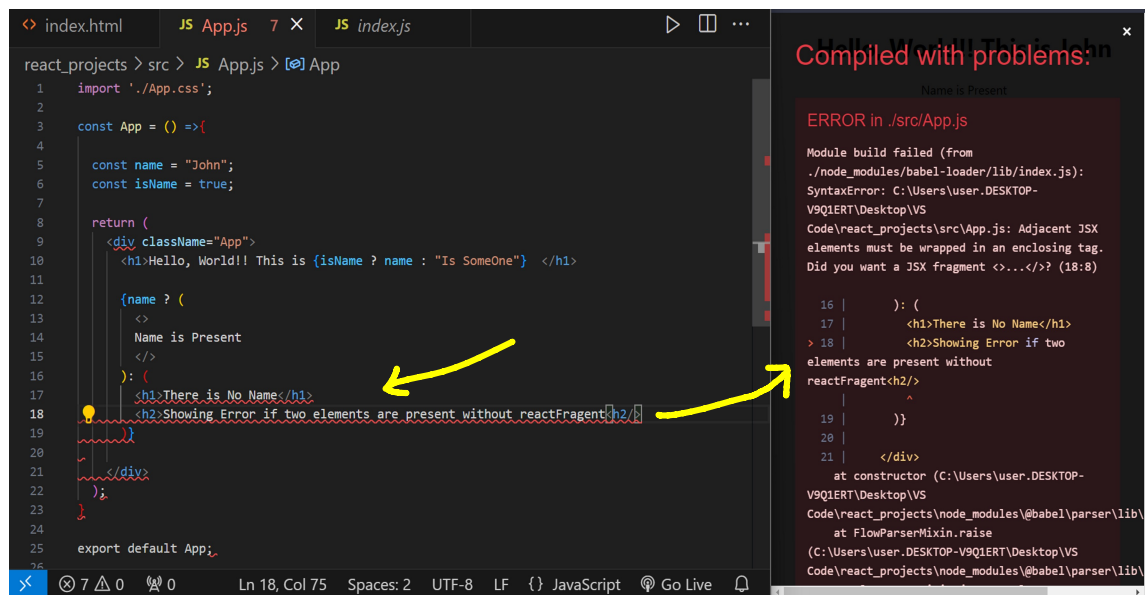
export default App;
```

=> In React we can directly inject javascript in html and also can use ternary operations.

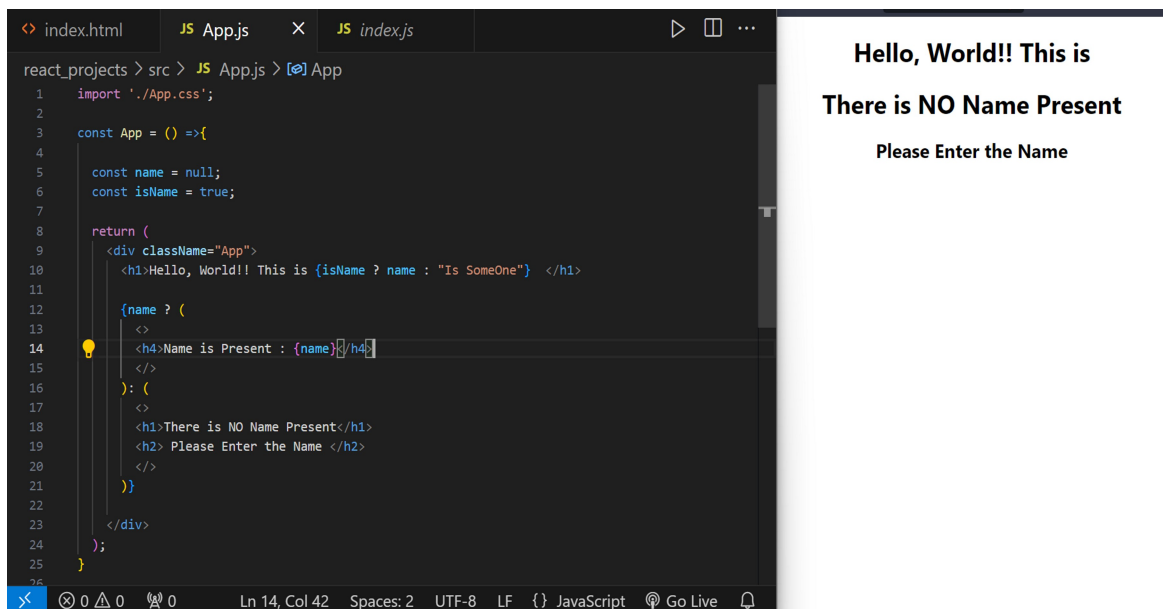
```
react_projects > src > JS App.js > App
1  import './App.css';
2
3  const App = () => {
4
5    const name = "John";
6    const isName = true;
7
8    return (
9      <div className="App">
10     | <h1>Hello, World!! This is {isName ? name : "Is SomeOne"} </h1>
11     | </div>
12
13   );
14 }
15
16 export default App;
17
```

Direct use of JS in HTML which makes it Dynamic

=> In react we have reactFragment which is like empty div in which we can write some code . It is important as if we want to run two diff different elements we need to warp them in a react Fragment

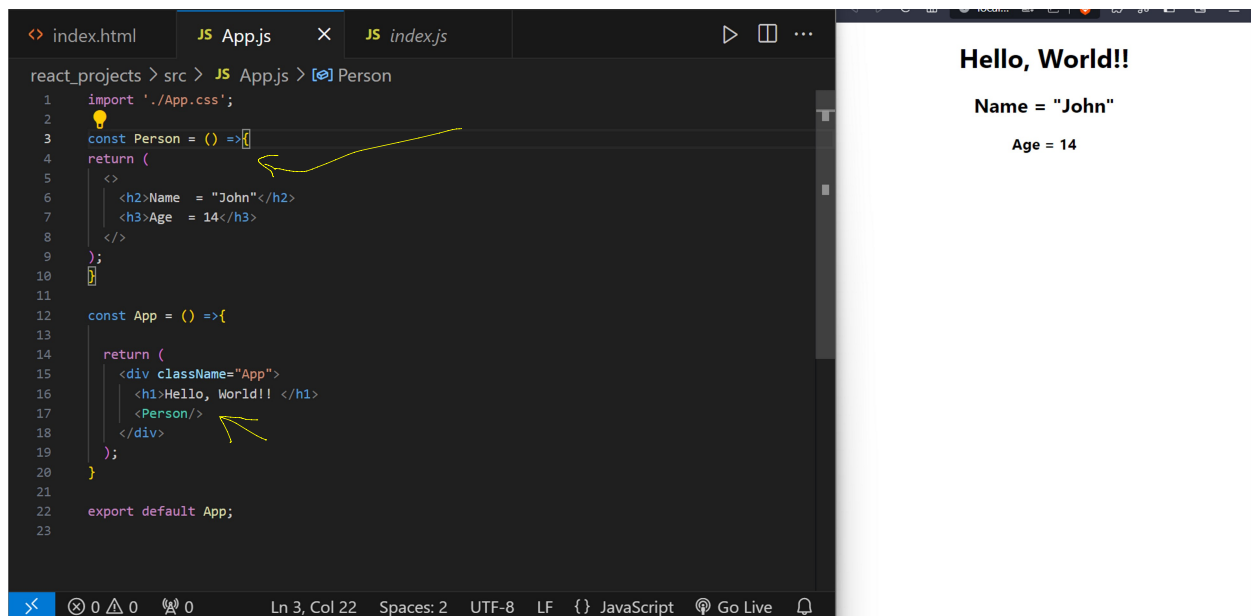


So if we just use the reactFragemt we can use both the elements



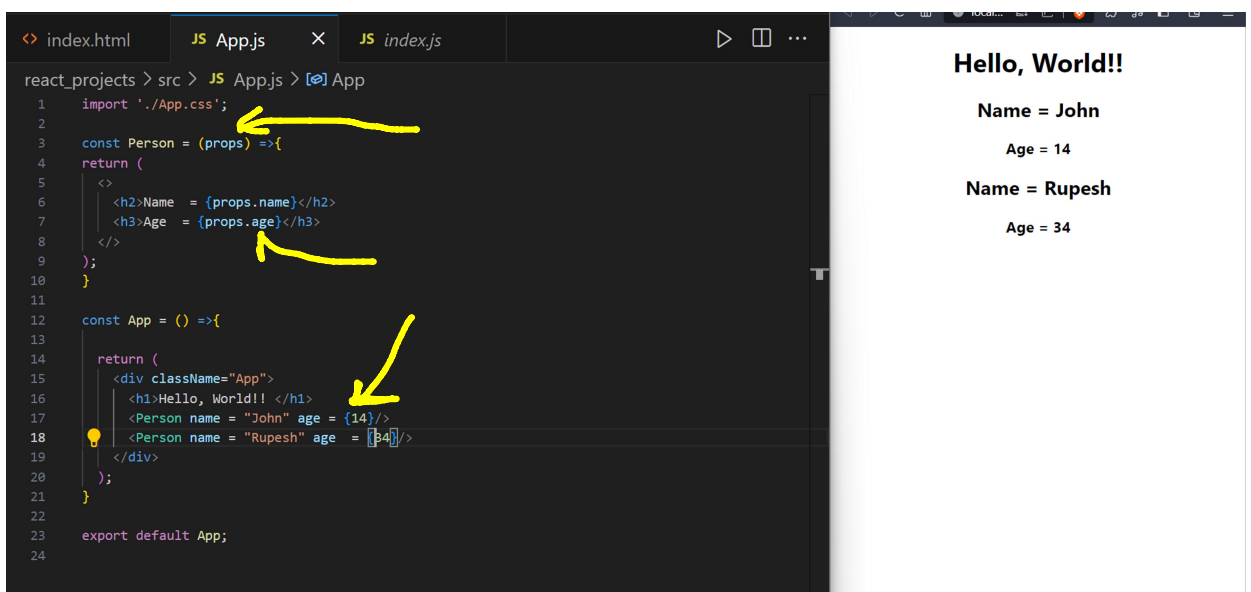
=> Components in ReactJS

We can create as many small components and can call them in the main component (like functions)



Props In ReactJS

Props allows u to pass dynamic data in components they are like arguments allowing to pass the dynamic data.



=> React State

In react state react is a plain JavaScript Object used by react to represent a piece of information about the components current situation, and completely managed by the component itself

In order to do so we have to import the react usestate hook i.e. `import { useState } from 'react';`

Hook : whenever a function is called and it start with 'use' its called a hook

setter
func()

setterFunction = set + functionName
(state Function)

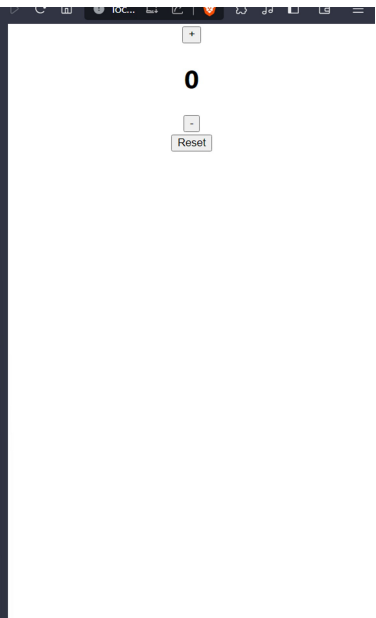
name
of
state

(State Variable)

```

index.html  App.js
react_projects > src > App.js > (x) App
1
2 import { useState } from 'react';
3 import './App.css';
4
5
6
7 const App = () =>{
8
9   const [counter, setCounter] = useState(0);
10
11
12   return (
13     <div className="App">
14       <button onClick={() => setCounter((prevCount) => prevCount + 1)}>+</button>
15       <h1>{counter}</h1>
16       <button onClick={() => setCounter((prevCount) => prevCount - 1)}>-</button>
17       <br/>
18       <button onClick={() => setCounter(0)}>Reset</button>
19     </div>
20   );
21 }
22
23 export default App;
24

```



Note : the important part is that we are changing the state without reloading the window

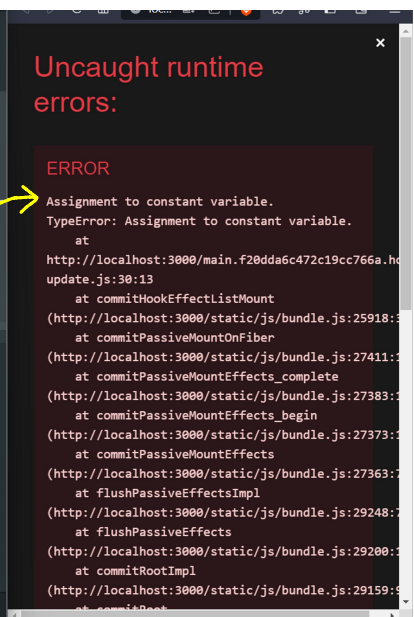
React has three main hooks and some additional hooks : 1> useState (), use Effect(), useContext;

Note : Never change the state variable itself or make it static always try to use the state function to do so.

```

index.html  App.js
react_projects > src > App.js > (x) App > fx useEffect() callback
1
2 import { useState , useEffect } from 'react';
3 import './App.css';
4
5
6
7 const App = () =>{
8
9   const [counter, setCounter] = useState(0);
10
11
12   useEffect(()=>{
13     counter = 100;
14   });
15
16   return (
17     <div className="App">
18       <button onClick={() => setCounter((prevCount) => prevCount + 1)}>+</button>
19       <h1>{counter}</h1>
20       <button onClick={() => setCounter((prevCount) => prevCount - 1)}>-</button>
21       <br/>
22       <button onClick={() => setCounter(0)}>Reset</button>
23     </div>
24   );
25 }
26

```



```
index.html 0: App.js x
react_projects > src > 0: App.js > (x) App > fx useEffect() callback
1
2 import { useState, useEffect } from 'react';
3 import './App.css';
4
5
6
7 const App = () =>{
8
9   const [counter, setCounter] = useState(0);
10
11   useEffect(()=>{
12     setCounter(100);
13   });
14
15   return (
16     <div className="App">
17       <button onClick={() => setCounter((prevCount) => prevCount + 1)}></button>
18       <h1>{counter}</h1>
19       <button onClick={() => setCounter((prevCount) => prevCount - 1)}></button>
20       <br/>
21       <button onClick={() => setCounter(0)}>Reset</button>
22     </div>
23   );
24 }
```

But even in this as soon we try to increase the counter it reloads itself to 100 so the second parameter of useEffect is the dependency array

```
useEffect(()=>{
  setCounter(100);
}, []);
```