React JS

Learning ReactJS

PRAVEEN NAIR

What is React?

ReactJS is a popular JavaScript library

It is used for building user interfaces for web applications.

It was developed and maintained by Facebook

It is great for SPA which load once and later everything happens thru JavaScript without reloading the page.

React was release on 29th May 2013.

Similar products available in market are Angular and Veujs.

Installing React

Download and Install Node Js

node –version

npm – version (node package manager)

if nodem error try **npm install -g npm**

npx create-vite@latest myapp

Select react and then javascript

npm install, code . and then npm run dev

Vscode extention vscode es7 react/reduc/react-native/js snippets

React JSX (JavaScriptXML)

JSX allows us to write HTML elements in JavaScript and place them in the DOM

```
export default function App() {
  return (<h1>Hello World</h1>);
}
```

.....

Tags must be closed

One top Element / Fragment

```
<div>
  <p>Paragraph 1</p>
  <p>Paragraph 2</p>
</div>


<>
  <p>Paragraph 1</p>
  <p>Paragraph 2</p>
</>
```

React Components (jsx)

```
export default function Header() {
  return <h1>Company Name</h1>;
}

export default function Content() {
  return <h1>Display content here</h1>;
}

export default function Footer() {
  return <h1>Footer Section</h1>;
}
```

Component Props & Expression

....User.js.....

```
export default function User(props) {
  return (
    <div>Hello {props.name}</div>
  )
}
```

.... App.js

```
<div>
<User name="John"/>
    </div>
```

React Styling – Inline Styling

```
const Header = () => {
  return (
    <>
      <h1 style={{backgroundColor: “blue"}}>Hello World!</h1>
      <p>Add a little style!</p>
    </>
  );
}
```

.....

camelCase property name – used backgroundColor instead of background-color

React Styling – Importing directly

```
import ‘./Styles.css’;
const Student = () => {
  return <h1 className=“txtblue”>Hello Student!</h1>;
}
export default Student;
.....
Style.css
.txtblue {
  color: Blue;
  padding: 20px;
  font-family: Sans-Serif;
  text-align: center;
}
```

If Statement with props

```
function Result(props) {
  const r = props.r;
  if (r) {
    return <Pass/>;
  }
  return <Fail/>;
}
```

Ternary Operator

```
.....
function Result(props) {
  const r = props.r;
  return (
    <>
      { r ? <Pass/> : <Fail/> }
    </>
  );
}
```

Events in React

```
function Customer() {
  const greet = () => {
    alert("Hello!");
  }
  return (
    <button onClick={greet}>Greet</button>
  );
}
```

Events (Passing Arguments)

```
function Customer() {
  const greet = (a) => {
    alert(a);
  }
  return (
    <button onClick={() => greet("Hi Ajay")}>Greet</button>
  );
}
```

State Management (useState hook)

The data is stored in the React Component's state object.

When the state object changes, the component will re-render itself.

```
export default function App6() {
  const [count, setCount] = useState(1); //cannot be inside loops, if or functions
  console.log(Date())  //runs everytime count changes
  return (
    <>
      <button onClick={() => setCount((prevState) => prevState - 1)}>-</button>
      <span>{count}</span>
      <button onClick={() => setCount((prevState) => prevState + 1)}>+</button>
   </>
  );
}
```

Spread Operator Arrays & Objects (...)

```
let score = [1,4,5];
 score = [...score, 9];
 console.log(score);
```

.....

```
let student = { name: "John",subject:"Maths" };
 student = {...student, marks:90};
 console.log(student);
```

React Router

npm i -D react-router-dom

- <Router> //Browserrouter as router
  <Link to="/add">Add</Link> |
  <Link to="/view">View</Link>
  <Routes>
  <Route index element={<Add />} />
  <Route path="add" element={<Add />} />
  <Route path="view" element={<View />} />
  </Routes>
  </ Router >

useNavigate (react-router-dom)

```
import React from 'react’
import { useNavigate } from 'react-router-dom’
export default function Post() {
  const Navigate = useNavigate()
  const goToFeeds = () => {
    Navigate('/’)
  }
  return (
    <div>Post
      <button onClick={goToFeeds}>Go to Feeds</button>
    </div>

  )
}
```

Side Effects (useEffect hook)

```
export default function App6() {
  const [runs, setRuns] = useState(0);
  const [wickets, setWickets] = useState(0);
  //   console.log("Component loaded");
  useEffect(() => {
    if (wickets > 0) console.log("Better Luck Next Time!");
  }, [wickets]);
  useEffect(() => {
    if (runs > 0) console.log("Good Job!");
  }, [runs]);
  return (
    <>
      <button onClick={() => setRuns((prevState) => prevState + 1)}>
        Runs({runs})
      </button>
      <button onClick={() => setWickets((prevState) => prevState + 1)}>
        Wickets({wickets})
      </button>
    </>
  );
}
```

Components Life Cycle

Mounting, unmounting and updating
comment strictmode to prevent rerender twice

```
useEffect(() => {
    console.log("Component: Mounting");
    return () => {
      console.log("Component: Unmounting");
    };
  },[cnt,result]);
.....
function Result(props) {
  const r = props.r;
  if (r) {
    return <Pass/>;
  }
  return <Fail/>;
}
```

useContext(child can access var)

```
import { useState, createContext, useContext } from "react“
import Child from "./Child";
export const UserContext = createContext();
export default function App13() {
const [user, setUser] = useState("John");
  const [email, setEmail] = useState("john@gmail.com")
  return (
    <>
    <UserContext.Provider value={{user, email}}>
      <h2>Hello {user} from App13 component</h2>
      <Child />
    </UserContext.Provider>
    </>
  );
}
.....
import {UserContext} from "./App13“
import { useContext } from "react";
export default function Child() {
  const {user, email} = useContext(UserContext);
  return (
    <>
      <h2>Hello {user} from Child component</h2>
    </>
  );
}
```

React Class vs Func Components

```
class Bus extends React.Component {
  render() {
    return <h2>This is a bus.</h2>;
  }
}
.....................................................

function Bus() {  //function based component is popular
  return <h2>This is a bus.</h2>;
}
.....................................................
root.render(<Bus />);
```

Map function to display array

```
function Cart() {
  const arr = ["Rice", "Wheat", "Sugar"];
  return (
      <ul>
        {arr.map((item) => (
          <li>{item}</li>
        ))}
      </ul>
  );
}
```

List and Keys

Keys are used by React to track individual items in a list, enabling it to manage changes like reordering, addition, or removal of elements. Without keys, React may not update the list correctly, leading to inefficient re-renders and potential UI bugs.

When rendering a list in React, each child in the list needs to have a unique key. Keys help React keep track of which list items correspond to which DOM elements. If the list is reordered or new items are inserted, React uses the keys to figure out which items stayed the same and which ones changed.

CRUD API (Axios)

```
 const fetchUsers = async () => {
    const response = await axios.get(url);
    setUsers(response.data);
  };

  const updateUser = async () => {
    const data = {
      name: name,
      email: email,
    };
    await axios.put(url + userId, data);
  };

  const deleteUser = async () => {
    await axios.delete(url + userId);
  };

const addUser = async () => {
    const data = {
      name: name,
      email: email,
    };
    await axios.post(url, data);
  };
```

Ecomm shopping cart

Add to cart

Increase and decrease quantity

Thank You

- PRAVEEN NAIR