



React and Vite

Web Application

Suriya Natsupakpong, PhD

Institute of Field Robotics (FIBO)

King Mongkut's University of Technology Thonburi (KMUTT)

React History

- React was first created by Jordan Walke, a software engineer at Facebook.
- It was incorporated into Facebook's newsfeed in 2011 and later on Instagram when it was acquired by Facebook in 2012.
- At JSConf 2013, React was made open source, and it joined the crowded category of UI libraries like jQuery, Angular, Dojo, Meteor, and others.
- React components acted as the view layer or the user interface for your JavaScript applications.

Why Use React?

React is a JavaScript library for building user interfaces that helps us build single-page web and mobile applications. In single-page applications, the page is loaded only once. When the user interacts with the page, say, they click a button, the application will make a request to the server. When the page receives data from the server, it will only update the page partially without entirely reloading the page.



Ref: Sofela, Oluwatobi, React Explained Clearly All You Need to Build Great React.js Apps

What is JSX?

- JSX (JavaScript and XML) is a syntax extension to JavaScript that allows you to build React elements with HTML-like syntax right inside your JavaScript code.
- React element created and rendered with JSX syntax:

```
function MyBio(props) {  
  return <h1>My name is FIBO KMUTT.</h1>;  
}  
const root = ReactDOM.createRoot(document.getElementById("root"));  
root.render(<MyBio />);
```

- React element created and rendered with regular JavaScript syntax:

```
function MyBio(props) {  
  return React.createElement("h1", null, "My name is FIBO KMUTT.");  
}  
const root = ReactDOM.createRoot(document.getElementById("root"));  
root.render(React.createElement(MyBio));
```

JSX

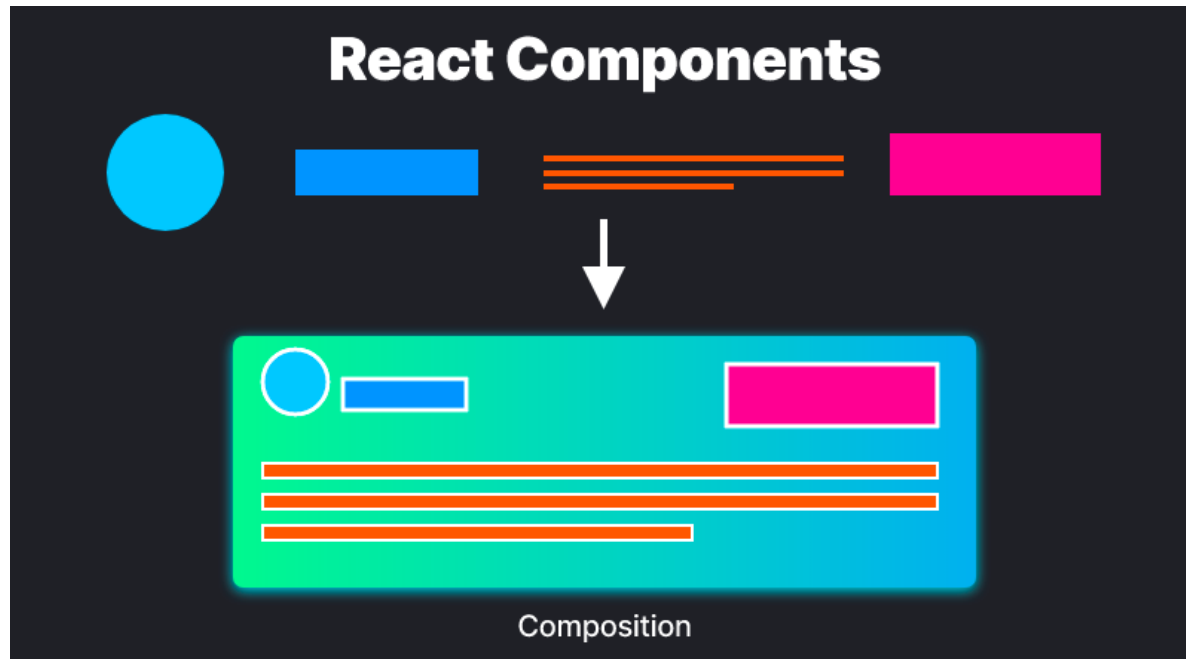
- JSX is a tag-based language and very similar to HTML
- Not indent-meaning, but keep the code clean and understanding
- Render can only contain one element which has many children, you can use a fragment `<> </>`.
- No auto closing
- Reserve keywords: `class`, `for`; can not use it as tag or attributes, change to `className`, `htmlFor`

```
root.render(  
  <div>  
    <h1 className='title'>Hello React</h1>  
    <p>Some <br/>content</p>  
    <input type="checkbox" id="my-checkbox" />  
    <label htmlFor="my-checkbox">my checkbox</label>  
  </div>  
)
```

- Single quotes and double quotes
- Inject variables and execute JavaScript between `{` and `}`

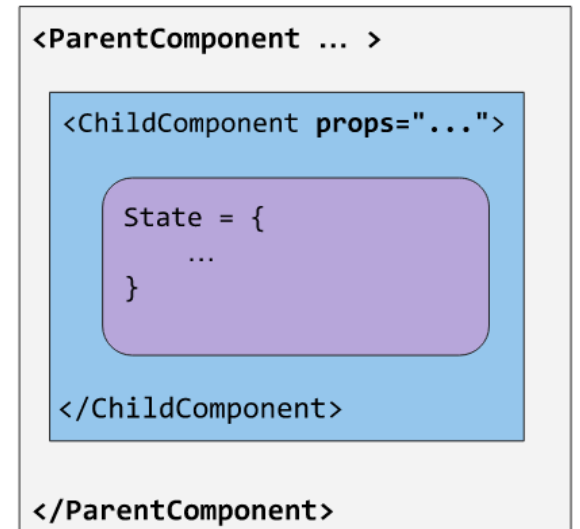
React Components

- The heart of all React applications are components.
- A component is a self-contained module that renders some output.



State and Props

- **props** are variables passed to it by its parent component.
- **State** on the other hand is still variables, but directly initialized and managed by the component.

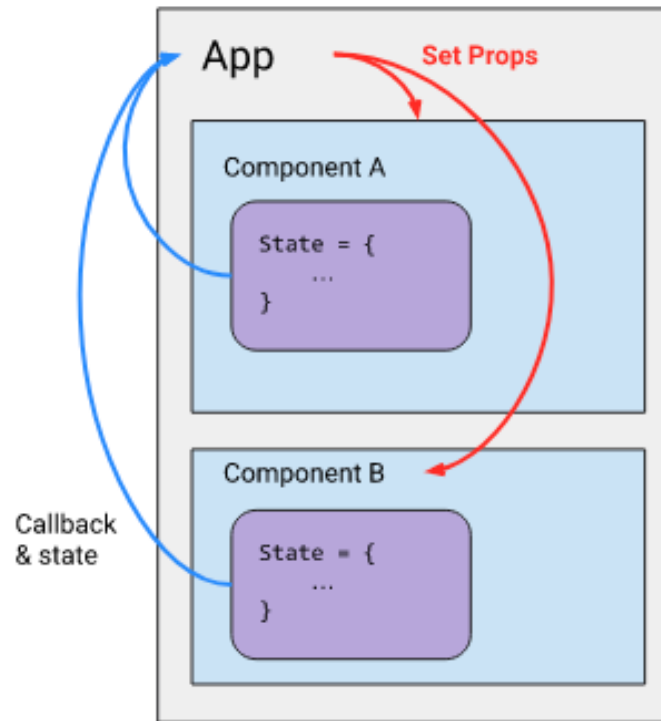


Props	State
The data remains unchanged from component to component.	Data is the current snapshot of data stored in a component's Props. It changes over the lifecycle of the component.
The data is read-only	The data can be asynchronous
The data in props cannot be modified	The data in state can be modified using <i>this.setState</i>
Props are what is passed on to the component	State is managed within the component

State Management

Without Redux:

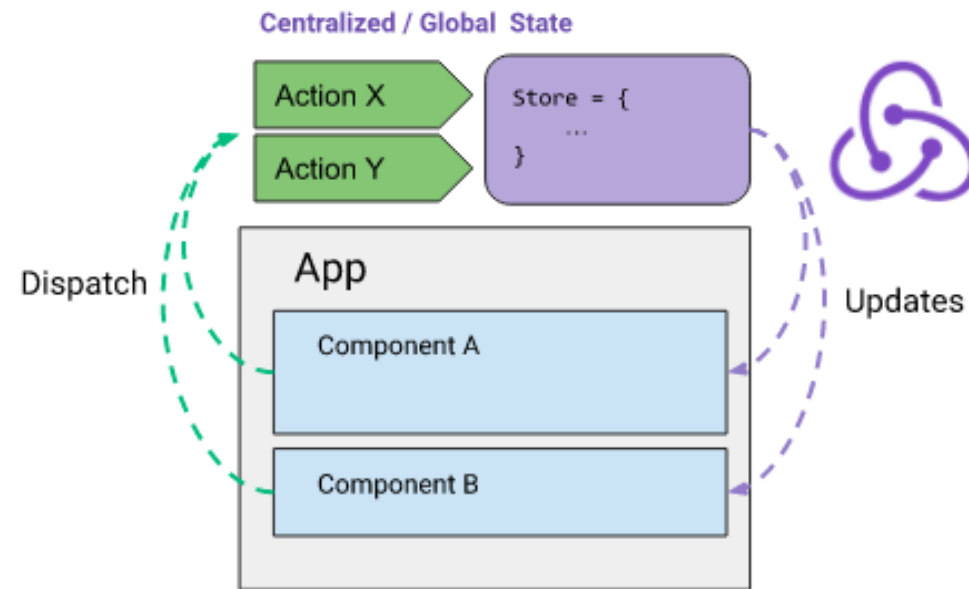
Each component has its own state, and requires extra logic for passing data outside of the component.



Redux is a JS library for predictable and maintainable global state management.

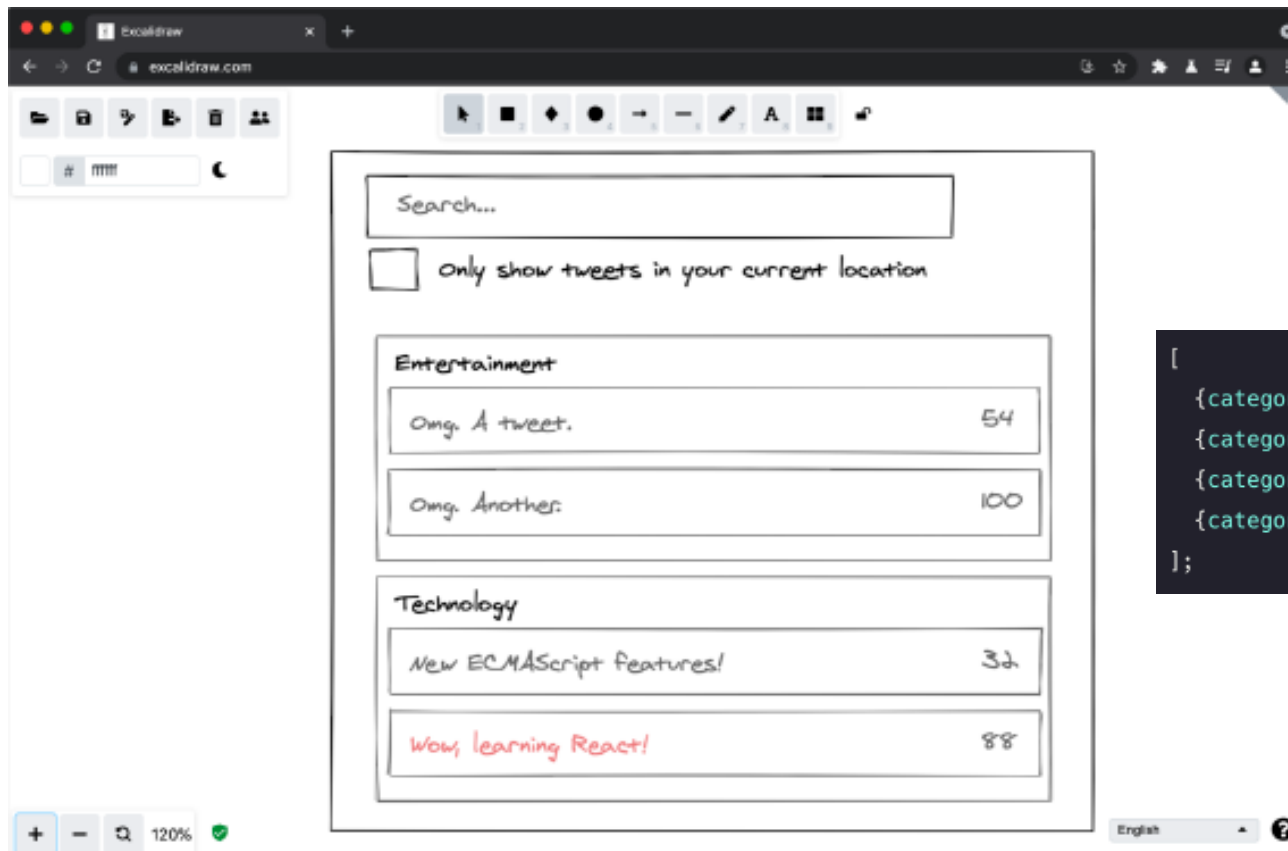
With Redux (Flux pattern):

All components refer to the centralized state. Each component only handles its presentation based on a specific state, but not the logic of data handling. (Similar to the View in an MVC framework)



Mock up UI with Excalidraw

- <https://excalidraw.com/>



```
[  
  {category: "Entertainment", retweets: "54", isLocal: false, text: "Omg. A tweet."},  
  {category: "Entertainment", retweets: "100", isLocal: false, text: "Omg. Another."},  
  {category: "Technology", retweets: "32", isLocal: false, text: "New ECMAScript features!"},  
  {category: "Technology", retweets: "88", isLocal: true, text: "Wow, learnin React!"}  
];
```

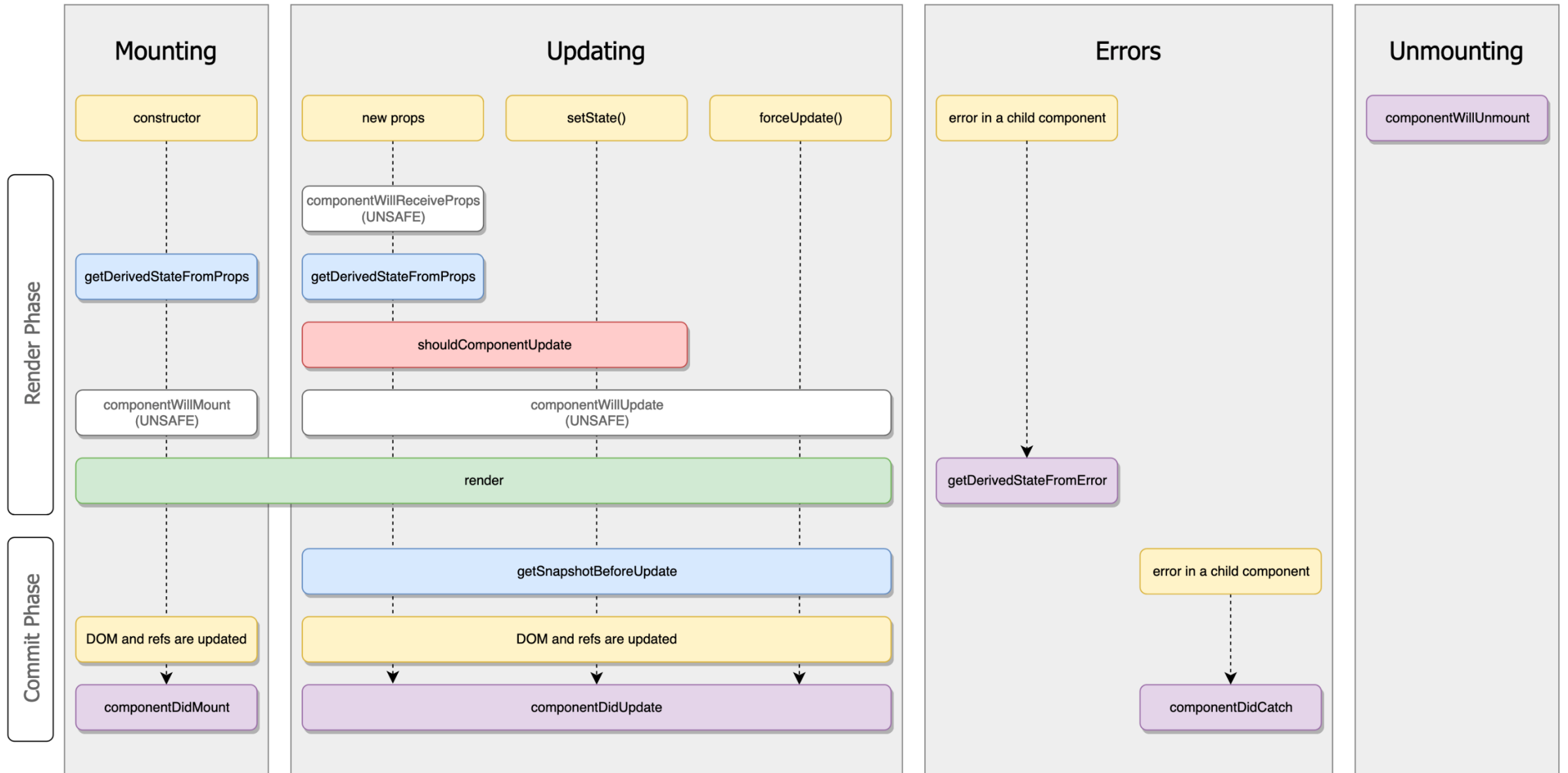

Break the UI into a Hierarchy Component

- TweetSearchResults (orange): container for the full component
- SearchBar (blue): user input for what to search for
- TweetList (green): displays and filters tweets based on user input
- TweetCategory (turquoise): displays a heading for each category
- TweetRow (red): displays a row for each tweet

- TweetSearchResults
 - SearchBar
 - TweetList
 - TweetCategory
 - TweetRow



React Component Lifecycle



React Component Lifecycle

App.js

```
import React, { Component } from 'react';
class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { hello: "JavaTpoint" };
    this.changeState = this.changeState.bind(this);
  }
  render() {
    return (
      <div>
        <h1>ReactJS component's Lifecycle</h1>
        <h3>Hello {this.state.hello}</h3>
        <button onClick={this.changeState}>Click Here!</button>
      </div>
    );
  }
  componentDidMount() {
    console.log("Component Did MOUNT!");
  }
  changeState() {
    this.setState({ hello: "All!!- Its a great reactjs tutorial." });
  }
  shouldComponentUpdate(newProps, newState) {
    return true;
  }
  componentDidUpdate(prevProps, prevState) {
    console.log("Component Did UPDATE!");
  }
  componentWillUnmount() {
    console.log("Component Will UNMOUNT!");
  }
}
export default App;
```

index.js

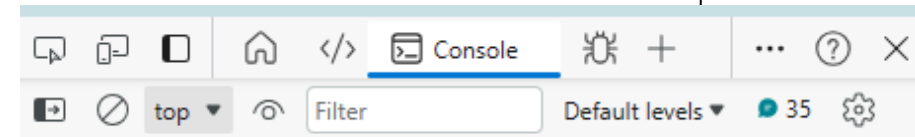
```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

ReactJS component's Lifecycle

Hello JavaTpoint

Click Here!



react refresh:6
Download the React DevTools for a better development experience: <https://reactjs.org/link/react-devtools>

Component Did MOUNT!	App.js:20
Component Will UNMOUNT!	App.js:34
Component Did MOUNT!	App.js:20

> |

HTML with React

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="css/app.css" />
    <title>React Example</title>
  </head>
  <body>
    <!-- Target container -->
    <div id="root"></div>
    <!-- ... other HTML ... -->

    <!-- Load React. -->
    <!-- Note: when deploying, replace "development.js" with "production.min.js". -->
    <script src="https://unpkg.com/react@18/umd/react.development.js" crossorigin></script>
    <script src="https://unpkg.com/react-dom@18/umd/react-dom.development.js" crossorigin></script>

    <!-- Load our React component. -->
    <script src="app.js"></script>

  </body>
</html>
```

First React Project

- Create app folder, go inside app folder and run the following command to create package.json file and install react related libraries

```
npm init -y
npm install react@18 react-dom@18.2 react-scripts@5.0
```

react is the core of React.

react-dom is a DOM renderer that will convert what we write in React to DOM elements for the web.

react-scripts contains common scripts that we need like running the dev server or building for production.

- Create folder name: `/public/` and create file: `index.html`
- Change title name in `<title>` tag as you want and add this in `<body>` tag:

```
<div id="root"></div>
```

- In `index.html` file type: `html:5` in vs code, it will generate template file.
- Create folder name: `/src/` and create file: `index.js`

`index.js`

```
import { createRoot } from "react-dom/client"

const root = createRoot(document.querySelector('#root'))
root.render(
  <h1>Hello React</h1>
)
```

Modified in `package.json`

```
{
  "name": "reactapp",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "dev": "react-scripts start",
    "build": "react-scripts build"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "react": "^18.3.1",
    "react-dom": "18.2",
    "react-scripts": "5.0"
  }
}
```

Create React App

- npx is installed with Node.js and NPM. It allows us to run NPM packages without having to create a project and install the dependencies as we did with npm.
- Run an NPM package: `create-react-app`, that will create a default React project.

How Does It Work?

- React operates on a virtual DOM, not directly on the browser's DOM.
- React resolves changes in its virtual DOM, after that React intelligently determines what changes to make to the actual DOM

Create React App with create-react-app

- With NPM

```
npx create-react-app my-project
```

- Or with Yarn

```
yarn create react-app my-project
```

- Go inside your project directory

```
cd my-project
```

- Start your application

```
npm start
```

 or

```
yarn start
```

- Open `http://localhost:3000` in your browser to view your app running live!



- If the errors occur, do the following:

```
npm uninstall react react-dom  
  
npm install react@18 react-dom@18  
  
npm install web-vitals  
  
npm start
```

Useful Commands

- Run your project in development mode.

```
npm start (or yarn start)
```

- Run React's test runner in watch mode

```
npm test (or yarn test)
```

- Create a minified bundle of your application to makes your app ready for deployment.

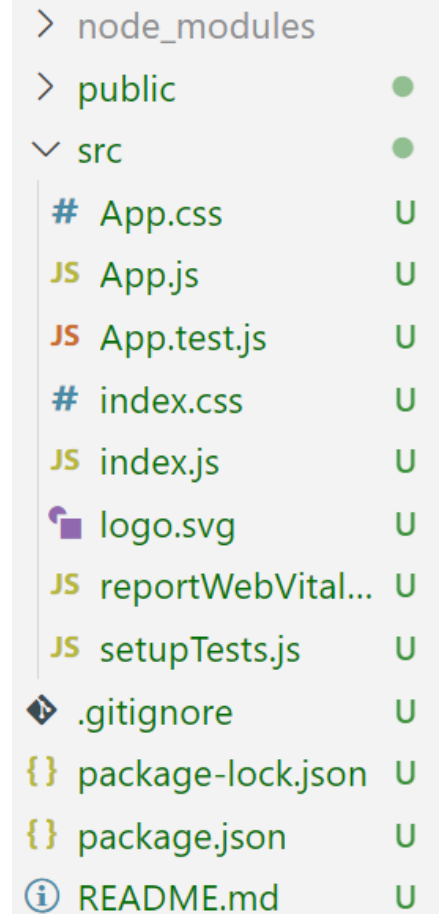
```
npm run build (or yarn build)
```

- Use a template to create your react app

```
npx create-react-app my-project --template your-template-name
```

- update your project's Create React App package to a newer version

```
npm install react-scripts@latest
```



Hello World

- Go inside the src directory of your newly configured React project and delete all the files in it.
- Create an index.js file.

```
// index.js
import * as React from "react";
import ReactDOM from "react-dom/client";

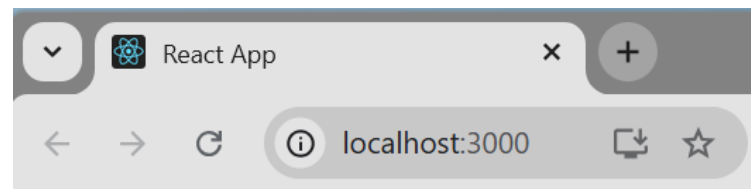
const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(<h1>Hello World!</h1>);
```

- Run your application

```
npm start
```

- Build your application

```
npm run build
```



Hello World!

```
build
├── static
│   ├── css
│   ├── js
│   └── media
├── asset-manifest.json
├── favicon.ico
├── index.html
├── logo192.png
├── logo512.png
├── manifest.json
└── robots.txt
```

What is a Component?

- React component is a JavaScript function (or class) that can accept arbitrary inputs (called “props”) and returns an element (UI).
- Types of components:

Function component

A function component is a regular JavaScript function that can accept a single properties object (props) as its parameter and emits a React element as its return value.

```
function MyBio(props) {  
  return <h1>My name is FIBO KMUTT.</h1>;  
}
```

Class component

A class component is a plain JavaScript Class that extends a user-defined class to the builtin React.Component class located inside the React library.

```
class MyBio extends React.Component {  
  constructor(props) {  
    super(props);  
  }  
  render() {  
    return <h1>My name is FIBO KMUTT.</h1>;  
  }  
}
```

How To Invoke React Components

- Invoke a component without passing any property (props)

```
function MyBio(props) {  
  return <h1>My name is FIBO KMUTT.</h1>;  
}  
<MyBio/>;
```

- Invoke a component with some properties (props)

```
function MyBio(props) {  
  return <h1>My name is  
    {props.firstName}</h1>;  
}  
<MyBio firstName="FIBO KMUTT" />;
```

- Invoke a component with the dot notation

```
// Define two components inside an object:  
const MyBio = {  
  FirstName: function FirstName(props) {  
    return <h1>My first name is  
      {props.firstName}</h1>;  
  }  
};  
  
// Invoke the FirstName component:  
<MyBio.FirstName firstName="FIBO KMUTT" />;
```

Example: List of Elements

```
// index.js
import React from "react";
import ReactDOM from "react-dom/client";
const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(
  <ol>
    <li>Blue</li>
    <li>White</li>
    <li>Peru</li>
  </ol>
);
```

```
// index.js
import React from "react";
import ReactDOM from "react-dom/client";

// Define the bestColors array:
const bestColors = ["Blue", "White", "Peru"];
// Use the bestColors array to create a list of React elements:
const bestColorsElements = bestColors.map((color) => (
  <li key={color.toString()}>{color}</li>
));
// Render the element array to the root DOM:
const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(<ul>{bestColorsElements}</ul>);
```

Example React Component

```
// index.js
import * as React from "react";
import ReactDOM from "react-dom/client";
import App from "./App";

const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(<App />);
```

```
// App.js
import * as React from "react";

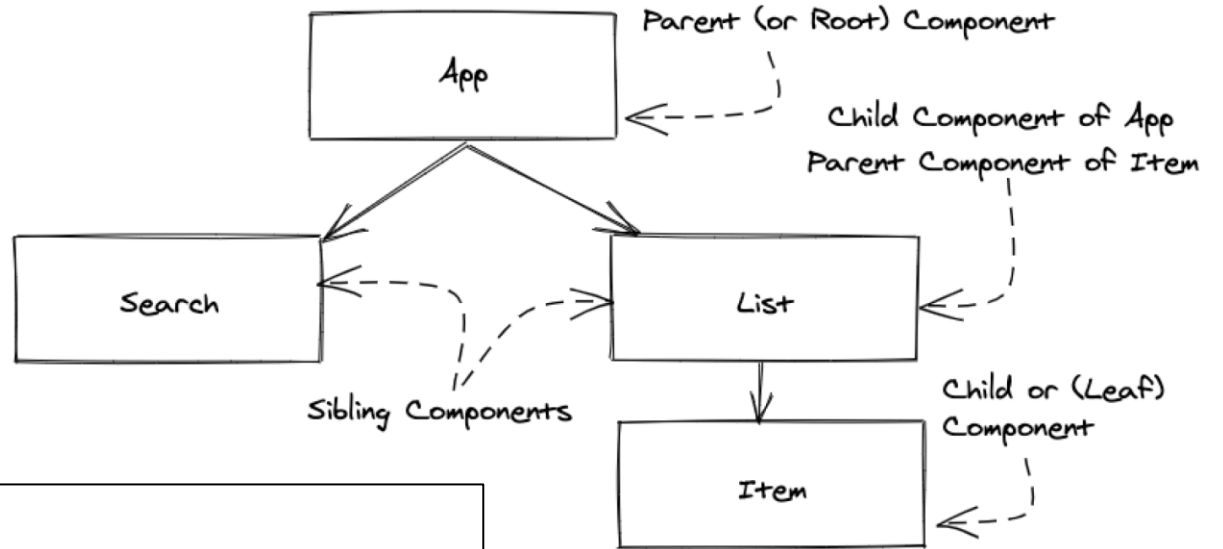
const title = "Web Programming";
const list = [
  {
    title: "React",
    url: "https://reactjs.org/",
    author: "Jordan Walke",
    num_comments: 3,
    points: 4,
    objectID: 0,
  },
  {
    title: "Redux",
    url: "https://redux.js.org/",
    author: "Dan Abramov, Andrew Clark",
    num_comments: 2,
    points: 5,
    objectID: 1,
  },
];
```

```
function List() {
  return (
    <ul>
      {list.map(function (item) {
        return (
          <li key={item.objectID}>
            <span>
              <a href={item.url}>{item.title}</a>
            </span>
            <span> {item.author}</span>
            <span> {item.num_comments}</span>
            <span> {item.points}</span>
          </li>
        );
      })}
    </ul>
  );
}

const App = () => {
  const Search = () => {
    const List = () => {
      (item) => {
```

```
function Search() {
  return (
    <div>
      <label htmlFor="search">Search: </label>
      <input id="search" type="text" />
    </div>
  );
}
```

```
export default function App() {
  return (
    <div>
      <h1>Hello {title}</h1>
      <hr />
      <Search />
      <List />
    </div>
  );
}
```



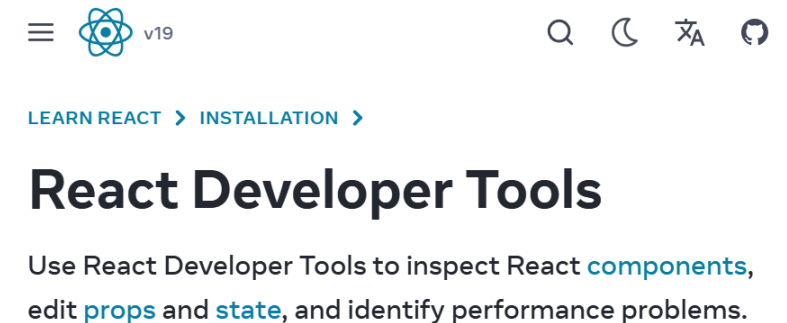
Example with Props State and Events

- `import React from 'react'` imports the 'react' library.
- The components must return a single root element.
- `export default Products` makes this component available for other files in our application to import it.
- Use by `import Products from './Products'`;
- `<Products />` here acts as a custom tag which allows us to extend or control our virtual DOM.

```
import React from "react";
function Products() {
  return (
    <div>
      <h2>Products</h2>
      <h2>Description</h2>
    </div>
  );
}
export default Products;
```

```
import React from "react";

function Products() {
  const products = ["HTML", "CSS", "JavaScript", "Node.js", "React"];
  const listProducts = products.map((product) => (
    <li key={product.toString()}>{product}</li>
  ));
  return (
    <div>
      <ul>{listProducts}</ul>
    </div>
  );
}
export default Products;
```



CSS Class Binding

Example with Props State and Events

- using react-bootstrap to make our button look more professional from <https://react-bootstrap.github.io>

```
npm install react-bootstrap bootstrap
```

- Add the following in index.js or App.js:

```
import 'bootstrap/dist/css/bootstrap.min.css';
```

- Improving the look by using the React icon library from <https://react-icons.github.io/react-icons/>

```
npm install react-icons
```

- Select the icons and import:

```
import { IoIosStar, IoIosStarOutline } from 'react-icons/io'
```

Props and State

Example with Props State and Events

- The 'props' object contains JSX attributes by passing data into a components.
- The 'props' are read-only.
- State is similar to props, but it is private and fully controlled by the component.
- State manages data that will change within a component. Whenever state changes,, the UI is re-rendered to reflect those changes.

```
import { useState } from "react";
```

- useState returns an array with two values: the current state value and a function that lets you update it.

```
const [rating, setRating] = useState(props.rating);
```


Events

Example with Props State and Events

- Handling events with React components is very similar to handling events on DOM elements.
- We CANNOT modify our state directly like `rating = 1`, Whenever we want to modify our state, we must use the state setter method we declared earlier, i.e. `setRating`

```
const styles = {  
  starStyle: {  
    color: 'orange'  
  }  
}
```

```
<div style={styles.starStyle} >
```

```
// Rating.js  
import React from "react";  
import { useState } from "react";  
import { IoIosStar, IoIosStarOutline } from "react-icons/io";  
  
export default function Rating(props) {  
  const [rating, setRating] = useState(props.rating);  
  let content = [];  
  for (let i = 1; i < 6; i++) {  
    content.push(  
      rating >= i ? (  
        <IoIosStar key={i} onClick={() => setRating(i)} />  
      ) : (  
        <IoIosStarOutline key={i} onClick={() => setRating(i)} />  
      )  
    );  
  }  
  return (  
    <div>  
      <h3>Rating: {rating} {content}</h3>  
    </div>  
  );  
}
```

Example Application

```
// Products.js
import React from "react";
import Product from "../Product";

function Products() {
  const getProducts = () => {
    return [
      {
        imageUrl: "http://loremflickr.com/150/150?random=1",
        productName: "Product 1",
        releasedDate: "May 31, 2016",
        description: "Lorem ipsum dolor sit amet, consectetur.",
        rating: 4,
        numOfReviews: 2,
      },
      {
        imageUrl: "http://loremflickr.com/150/150?random=2",
        productName: "Product 2",
        releasedDate: "October 31, 2016",
        description: "Lorem ipsum dolor sit amet, consectetur.",
        rating: 2,
        numOfReviews: 12,
      },
      {
        imageUrl: "http://loremflickr.com/150/150?random=3",
        productName: "Product 3",
        releasedDate: "July 30, 2016",
        description: "Lorem ipsum dolor sit amet, consectetur.",
        rating: 5,
        numOfReviews: 2,
      },
    ];
  };
}
```

```
const products = getProducts();
const listProducts = products.map((product) => (
  <Product key={product.productName} data={product} />
));

return (
  <div>
    {listProducts.length > 0 ? (
      <ul>{listProducts}</ul>
    ) : (
      <ul>No Products to display</ul>
    )}
  </div>
);
}

export default Products;
```

```
// App.js
import React from "react";
import "bootstrap/dist/css/bootstrap.min.css";
import Products from "../Products";

function App() {
  return (
    <div>
      <Products />
    </div>
  );
}

export default App;
```

```
// Products.js
import Rating from "../Rating";
import { Card } from "react-bootstrap";

const style = {
  cardImg:{
    width:128
  }
}

const Product = (props) => {
  return (
    <div>
      <Card>
        <Card.Img style={style.cardImg}
          src={props.data.imageUrl}
          alt="Image"
        />
        <Card.Body>
          <h4> {props.data.productName} </h4>
          {props.data.releaseDate}
          <Rating
            rating={props.data.rating}
            numOfReviews={props.data.numOfReviews}
          />
          <p> {props.data.description} </p>
        </Card.Body>
      </Card>
    </div>
  );
};

export default Product;
```



Product 1

May 31, 2016

Rating: 4 ★★★★★☆

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean porttitor, tellus laoreet venenatis facilisis, enim ex faucibus nulla, id rutrum ligula purus sit amet mauris.



Product 2

October 31, 2016

Rating: 2 ★★☆☆☆☆

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean porttitor, tellus laoreet venenatis facilisis, enim ex faucibus nulla, id rutrum ligula purus sit amet mauris.



Product 3

July 30, 2016

Rating: 5 ★★★★★★

Building Forms with Hooks

Email address

Enter email

We'll never share your email with anyone else.

Password

Password

Submit

Email entered:

Password entered:

```
// UserForm.js
import React, { useState } from "react";
import { Form, Button, Alert } from "react-bootstrap";
function UserForm() {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [emailError, setEmailError] = useState("");
  const [passwordError, setPasswordError] = useState("");
  const handleSubmit = (event) => {
    event.preventDefault();
    var emailValid = false;
    if (email.length == 0) {
      setEmailError("Email is required");
    } else if (email.length < 6) {
      setEmailError("Email should be minimum 6 characters");
    } else if (email.indexOf(" ") >= 0) {
      setEmailError("Email cannot contain spaces");
    } else {
      setEmailError("");      emailValid = true;
    }
    var passwordValid = false;
    if (password.length == 0) {
      setPasswordError("Password is required");
    } else if (password.length < 6) {
      setPasswordError("Password should be minimum 6 characters");
    } else if (password.indexOf(" ") >= 0) {
      setPasswordError("Password cannot contain spaces");
    } else {
      setPasswordError("");      passwordValid = true;
    }
    if (emailValid && passwordValid) {
      alert("Email: " + email + "\nPassword: " + password);
      setEmail("");
      setPassword("");
    }
  };
}
```

```

return (
  <div>
    <Form onSubmit={handleSubmit}>
      <Form.Group controlId="formBasicEmail">
        <Form.Label>Email address</Form.Label>
        <Form.Control
          type="email" placeholder="Enter email"
          onChange={(event) => setEmail(event.target.value)}
          value={email}
        />
        <Form.Text className="text-muted">
          We'll never share your email with anyone else.
        </Form.Text>
      </Form.Group>
      {emailError.length > 0 && <Alert variant="danger">{emailError}</Alert>}
      <Form.Group controlId="formBasicPassword">
        <Form.Label>Password</Form.Label>
        <Form.Control
          type="password" placeholder="Password"
          onChange={(event) => setPassword(event.target.value)}
          value={password}
        />
      </Form.Group>
      {passwordError.length > 0 && (
        <Alert variant="danger">{passwordError}</Alert>
      )}
      <Button variant="primary" type="submit">
        Submit
      </Button>
    </Form>
    Email entered: {email}
    <br />
    Password entered: {password}
  </div>
);
}
export default UserForm;

```

Getting Data From RESTful APIs with Hooks

- Learn more about the GitHub API at <https://developer.github.com/v3/>
- Try with <https://api.github.com/search/users?q=kmutt>
- Install axios library: `npm install axios`

```
// GitHub.js
import React, { useEffect } from "react";
import axios from "axios"; // npm install axios
function GitHub() {
  useEffect(() => {
    axios.get("https://api.github.com/search/users?q=kmutt").then((res) => {
      console.log(res.data.items);
    });
  }, []);
  return <div></div>;
}
export default GitHub;
```

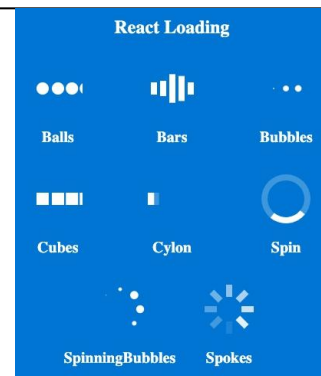
```
▼ Array(30) i
  ► 0: {login: 'KMUTT', id: 6086835, node_id: 'MDEyOk9yZ2FuaXphdGlvbjYwODY4MzU=', avatar_url: 'https://avatars.githubusercontent.com/u/6086835?v=4', ...}
  ► 1: {login: 'ChaiyapatOam', id: 83873103, node_id: 'MDQ6VXNlcjgzODczMTAz', avatar_url: 'https://avatars.githubusercontent.com/u/83873103?v=4', ...}
  ► 2: {login: 'jukbot', id: 7373631, node_id: 'MDQ6VXNlcjczNmZmZmE=', avatar_url: 'https://avatars.githubusercontent.com/u/7373631?v=4', ...}
  ► 3: {login: 'XiaoXuxxxx', id: 73125291, node_id: 'MDQ6VXNlcjczMTI1MjIx', avatar_url: 'https://avatars.githubusercontent.com/u/73125291?v=4', ...}
  ► 4: {login: 'kmutt-cpe-student', id: 177311134, node_id: 'O_kgDOCPGNg', avatar_url: 'https://avatars.githubusercontent.com/u/177311134?v=4', ...}
  ► 5: {login: 'praisan', id: 7611167, node_id: 'MDQ6VXNlcjczMTEwNjc=', avatar_url: 'https://avatars.githubusercontent.com/u/7611167?v=4', ...}
  ► 6: {login: 'siraom15', id: 51380509, node_id: 'MDQ6VXNlcjUxMzgWNTA5', avatar_url: 'https://avatars.githubusercontent.com/u/51380509?v=4', ...}
  ► 7: {login: 'GDSC-KMUTT', id: 114597589, node_id: 'O_kgDOBtSe1Q', avatar_url: 'https://avatars.githubusercontent.com/u/114597589?v=4', ...}
  ► 8: {login: 'Bazukaaa', id: 60100309, node_id: 'MDQ6VXNlcjYwMTAwMzA5', avatar_url: 'https://avatars.githubusercontent.com/u/60100309?v=4', ...}
  ► 9: {login: 'JeromeTana', id: 88102079, node_id: 'MDQ6VXNlcjg4MTAyMDc5', avatar_url: 'https://avatars.githubusercontent.com/u/88102079?v=4', ...}
  ► 10: {login: 'chon26909', id: 59905927, node_id: 'MDQ6VXNlcjU5OTA1OTI3', avatar_url: 'https://avatars.githubusercontent.com/u/59905927?v=4', ...}
  ► 11: {login: 'jigneng1', id: 83826754, node_id: 'MDQ6VXNlcjgzODI2NzU0', avatar_url: 'https://avatars.githubusercontent.com/u/83826754?v=4', ...}
```

```
{
  "total_count": 449,
  "incomplete_results": false,
  "items": [
    {
      "login": "KMUTT",
      "id": 6086835,
      "node_id": "MDEyOk9yZ2FuaXphdGlvbjYwODY4MzU=",
      "avatar_url": "https://avatars.githubusercontent.com/u/6086835?v=4",
      "gravatar_id": "",
      "url": "https://api.github.com/users/KMUTT",
      "html_url": "https://github.com/KMUTT",
      "followers_url": "https://api.github.com/users/KMUTT/followers",
      "following_url": "https://api.github.com/users/KMUTT/following{/other_login}",
      "gists_url": "https://api.github.com/users/KMUTT/gists{/gist_id}",
      "starred_url": "https://api.github.com/users/KMUTT/starred{/owner}/{repo}",
      "subscriptions_url": "https://api.github.com/users/KMUTT/subscriptions",
      "organizations_url": "https://api.github.com/users/KMUTT/orgs",
      "repos_url": "https://api.github.com/users/KMUTT/repos",
      "events_url": "https://api.github.com/users/KMUTT/events{/privacy}",
      "received_events_url": "https://api.github.com/users/KMUTT/received_events",
      "type": "Organization",
      "user_view_type": "public",
      "site_admin": false,
      "score": 1.0
    },
    {
      "login": "ChaiyapatOam",
      "id": 83873103,
      "node_id": "MDQ6VXNlcjgzODczMTAz",
      "avatar_url": "https://avatars.githubusercontent.com/u/83873103?v=4",
      "gravatar_id": "",
      "url": "https://api.github.com/users/ChaiyapatOam",
      "html_url": "https://github.com/ChaiyapatOam",
      "followers_url": "https://api.github.com/users/ChaiyapatOam/followers",
      "following_url": "https://api.github.com/users/ChaiyapatOam/following{/other_login}",
      "gists_url": "https://api.github.com/users/ChaiyapatOam/gists{/gist_id}",
      "starred_url": "https://api.github.com/users/ChaiyapatOam/starred{/owner}/{repo}",
      "subscriptions_url": "https://api.github.com/users/ChaiyapatOam/subscriptions",
      "organizations_url": "https://api.github.com/users/ChaiyapatOam/orgs",
      "repos_url": "https://api.github.com/users/ChaiyapatOam/repos",
      "events_url": "https://api.github.com/users/ChaiyapatOam/events{/privacy}",
      "received_events_url": "https://api.github.com/users/ChaiyapatOam/received_events",
      "type": "User",
      "user_view_type": "public",
      "site_admin": false,
      "score": 1.0
    },
    {
      "login": "jukbot",
      "id": 7373631,
      "node_id": "MDQ6VXNlcjczNmZmZmE=",
      "avatar_url": "https://avatars.githubusercontent.com/u/7373631?v=4",
      "gravatar_id": "",
      "url": "https://api.github.com/users/jukbot",
      "html_url": "https://github.com/jukbot",
      "followers_url": "https://api.github.com/users/jukbot/followers",
      "following_url": "https://api.github.com/users/jukbot/following{/other_login}",
      "gists_url": "https://api.github.com/users/jukbot/gists{/gist_id}",
      "starred_url": "https://api.github.com/users/jukbot/starred{/owner}/{repo}",
      "subscriptions_url": "https://api.github.com/users/jukbot/subscriptions",
      "organizations_url": "https://api.github.com/users/jukbot/orgs",
      "repos_url": "https://api.github.com/users/jukbot/repos",
      "events_url": "https://api.github.com/users/jukbot/events{/privacy}",
      "received_events_url": "https://api.github.com/users/jukbot/received_events",
      "type": "User",
      "user_view_type": "public",
      "site_admin": false,
      "score": 1.0
    }
  ]
}
```

```
// GitHub.js
import React, { useEffect, useState } from "react";
import axios from "axios"; // npm install axios
import ReactLoading from "react-loading";
import { Card } from "react-bootstrap";

function GitHub() {
  const [data, setData] = useState([]);
  const [searchTerm, setSearchTerm] = useState("");
  const [isLoading, setIsLoading] = useState(false);
  useEffect(() => {
    if (searchTerm.length) getData();
  }, []);
  const getData = () => {
    axios
      .get(`https://api.github.com/search/users?q=${searchTerm}`)
      .then((res) => {
        setData(res.data.items);
        setIsLoading(false);
      })
      .catch((error) => {
        console.log(error);
      });
  };
  const listUsers = data.map((user) => (
    <Card key={user.id}>
      <a href={user.html_url}>
        <img width={64} height={64} className="mr-3"
          src={user.avatar_url} alt="Generic placeholder"/>
      </a>
      <Card.Body>
        <h5>Login: {user.login}</h5>
        <p>Id: {user.id}</p>
      </Card.Body>
    </Card>
  ));
}
```

```
npm install react-loading
```



```
const handleSubmit = (event) => {
  event.preventDefault();
  if (searchTerm.length) {
    setIsLoading(true);
    getData();
  }
};
return (
  <div>
    <h3>GitHub Users Search</h3>
    <form onSubmit={handleSubmit}>
      <input
        type="text"
        onChange={(event) => setSearchTerm(event.target.value)}
      />
      <button type="submit">Search</button>
    </form>
    <h3>Search Results</h3>
    {isLoading && <ReactLoading type="spinningBubbles" color="#444" />}
    {listUsers}
  </div>
);
export default GitHub;
```

GitHub Users Search

Search Results



Login: KMUTT

Id: 6086835



Login: ChaipayatOam

Id: 83873103

First Vite Project

- To create Vite project type:

```
npm create vite@latest
```

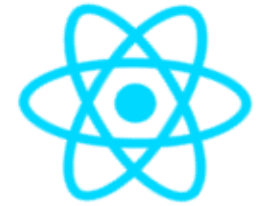
```
> npm create vite@latest  
Need to install the following packages:  
create-vite@6.1.1  
Ok to proceed? (y) y
```

- Change the project name as you want,
select a framework: **React**
and then select a variant: **Javascript**
- Go to project folder and type:

```
cd <project_folder>  
npm install  
npm run dev
```

Vite : Next Generation Frontend Tooling

- 💡 Instant Server Start
- ⚡ Lightning Fast HMR
- 🛠 Rich Features
- 📦 Optimized Build
- 🔌 Universal Plugin Interface
- 🔑 Fully Typed APIs



Vite + React

count is 2

Edit `src/App.jsx` and save to test HMR

[Click on the Vite and React logos to learn more](#)

JSX Features

- We can inject variables and execute JavaScript inside `{...}` `{ variable }`
- Comment, we have to open and close the `{...}` and then create a JS block comment `/* ... */` `{/* Some comment */}`
- The render can only contain one element.
- Some keywords like `class` or `for` are reserved, then `class` becomes `className` and `for` becomes `htmlFor`.
- We can write it directly in the style attribute.

```
<h1 style={{ color: 'coral', backgroundColor: 'floralwhite' }}>Hello React</h1>
```

- Or using a CSS file

```
.cute-paragraph
{
  color: coral;
  background: floralwhite;
}
```

```
import './style.css'
<h1 className="cute-paragraph">Hello React</h1>
```

App Component

- In `/src/`, create an `App.js` file and export an `App` function by default.

`index.js`

```
import { createRoot } from 'react-dom/client'
import App from './App.js'
import './style.css'

const root = createRoot(document.querySelector('#root'))
root.render(
  <div>
    <App></App>
  </div>
)
```

`App.js`

```
import Clicker from './Clicker.js'

export default function App()
{
  return <>
    <Clicker/>
  </>
}
```

`Clicker.js`

```
import { useState } from "react"

export default function Clicker()
{
  let count = 0

  const buttonClick = () =>
  {
    console.log('Button has been clicked')
    count++
  }

  return <div>
    <div>Click count: { count }</div>
    <button onClick={ buttonClick }>Click me</button>
  </div>
}
```

My First React App

Web Programming Course

Total count: 16

Hide Clicker

Click count: 4

Click me

Click count: 6

Click me

Click count: 6

Click me

The useState hook

- useState provides a variable and a function. When we want to change the variable, instead of re-assigning it, we will be using the function.

```
const [count, setCount] = useState(0)
```

Clicker.js

```
import { useState } from "react"

export default function Clicker()
{
  //const [count, setCount] = useState(0)
  const countState = useState(0)
  console.log(countState)

  const count = countState[0]
  const setCount = countState[1]

  const buttonClick = () =>
  {
    setCount(count + 1)
  }

  return <div>
    <div>Click count: { count }</div>
    <button onClick={buttonClick}>Click me</button>
  </div>
}
```

The useEffect hook for Save in localStorage

- we would like to save the value, when we close the tab and re-open it, we keep the value.
- In JavaScript, we have access to an API named localStorage that lets us save data locally as key and value.
- useEffect is being called, we can pass it an array of dependencies as the second argument. If that dependencies array is empty, our function will be called only during the first render.
- In useEffect, we are going to save count in localStorage with the setItem method. setItem expects a key and the value.
- To retrieve it when the component renders for the first time (when we open the page). We use another useEffect, but with an empty array as dependencies.
- To destroy the localStorage data, we add a return to the useEffect with the [] dependencies.

```
useEffect(() => {  
  console.log("First render");  
}, []);
```

```
useEffect(() => {  
  localStorage.setItem("count", count);  
}, [count]);
```

```
useEffect(() => {  
  const savedCount = parseInt(localStorage.getItem("count") ?? 0);  
  setCount(savedCount);  
}, []);
```

```
useEffect(() => {  
  return () => {  
    localStorage.removeItem("count");  
  };  
}, []);
```

Props in Component

App.js

```
import { useState } from "react"
import Clicker from "./Clicker.js"

export default function App({ children })
{
  const [ hasClicker, setHasClicker ] = useState(true)

  const toggleClickerClick = () =>
  {
    setHasClicker(!hasClicker)
  }

  return <>
    { children }

    <button onClick={toggleClickerClick}>{hasClicker ? 'Hide' : 'Show'} Clicker</button>

    { hasClicker && <>
      <Clicker increment = { increment } keyName="countA" color={`hsl(${Math.random() * 360}deg, 100%, 70%)`} />
      <Clicker increment = { increment } keyName="countB" color={`hsl(${Math.random() * 360}deg, 100%, 70%)`} />
      <Clicker increment = { increment } keyName="countC" color={`hsl(${Math.random() * 360}deg, 100%, 70%)`} />
    </>
  </>
}
```

index.js

```
import { createRoot } from 'react-dom/client'
import App from './App.js'
import './style.css'

const root = createRoot(document.querySelector('#root'))

root.render(
  <div>
    <App clickerCount={4}>
      <h1>My First React App</h1>
      <h2>Web Programming Course</h2>
    </App>
  </div>
)
```

Props in Component

clicker.js

Retrieve the variable from localStorage with the getItem() method

When the Clicker is being destroyed, the localStorage data will be deleted.

```
import { useEffect, useState } from "react"

export default function Clicker({ keyName, color })
{
  const [count, setCount] = useState(parseInt(localStorage.getItem(keyName) ?? 0))

  useEffect(() =>
  {
    return () =>
    {
      localStorage.removeItem(keyName)
    }
  }, [])

  useEffect(() =>
  {
    localStorage.setItem(keyName, count)
  }, [count ])

  const buttonClick = () =>
  {
    setCount(count + 1)
  }

  return <div>
    <div style={{color}}>Click count: { count }</div>
    <button onClick={buttonClick}>Click me</button>
  </div>
}
```

Loops and useMemo

useMemo needs a function as the first parameter and an array of dependencies as the second parameter.

useMemo works like a cache, and only a dependency value change can break it.

App.js

```
import { useState } from "react"
import Clicker from "../Clicker.js"
```

```
import { useMemo, useState } from "react"
```

```
export default function App({ clickerCount, children })
{
```

```
  const [ hasClicker, setHasClicker ] = useState(true)
  const [ count, setCount ] = useState(0)
```

```
  const toggleClickerClick = () =>
  { setHasClicker(!hasClicker) }
```

```
  const increment = () =>
  { setCount(count + 1) }
```

```
  const colors = []
  for(let i=0; i<clickerCount; i++)
    colors.push(`hsl(${Math.random() * 360}deg, 100%, 70%)`)
```

```
  const colors = useMemo(() =>
  {
    const colors = []
    for(let i=0; i<clickerCount; i++)
      colors.push(`hsl(${Math.random() * 360}deg, 100%, 70%)`)
    return colors
  }, [clickerCount])
```

If the clickersCount value changes, then useMemo will call the function again and we will always have the right amount of values in the colors array.

```
  return <>
  { children }
```

```
  <div>Total count: { count }</div>
  <button onClick={toggleClickerClick}>{hasClicker ? 'Hide' : 'Show'} Clicker</button>
```

```
  { hasClicker && <>
    { [...Array(clickerCount)].map((value, index) =>
      <Clicker
        key={ index }
        increment={ increment }
        keyName={ `count${index}` }
        color={ colors[index] }
      />
    ) }
  }
</>
```

```
}
```

useRef hook

To get access to the actual element, we can use the useRef hook. The idea is to create a reference first and then associate it with the targeted element.

We can now access the native DOM Element and do anything we want with it, but don't forget to use the current property and not the reference variable itself

Clicker.js

```
import { useRef, useEffect, useState } from "react"

export default function Clicker({ increment, keyName, color })
{
  const [count, setCount] = useState(parseInt(localStorage.getItem(keyName) ?? 0))
  const buttonRef = useRef()

  useEffect(() =>
  {
    buttonRef.current.style.backgroundColor = 'papayawhip'
    buttonRef.current.style.color = 'salmon'

    return () =>
    {
      localStorage.removeItem(keyName)
    }
  }, [])

  useEffect(() =>
  {
    localStorage.setItem(keyName, count)
  }, [count])

  const buttonClick = () =>
  {
    setCount(count + 1)
    increment()
  }

  return <div>
    <div style={{color}}>Click count: {count}</div>
    <button ref={buttonRef} onClick={buttonClick}>Click me</button>
  </div>
}
```

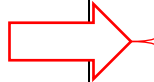

People Component

People.js

```
import { useState } from "react"

export default function People()
{
  const [ people, setPeople ] = useState([
    {id:1, name: 'John'},
    {id:2, name: 'Jane'},
    {id:3, name: 'Sudo'},
    {id:4, name: 'Boy'}
  ])

  return <div>
    <h2>People</h2>
    { people.map(person =>
      <li key={person.id}>{person.name}</li> ) }
    </div>
  }
```



```
const getPeople = async () =>
{
  const response = await fetch('https://jsonplaceholder.typicode.com/users')
  const result = await response.json()
  setPeople(result)

  // fetch('https://jsonplaceholder.typicode.com/users')
  //   .then(response => response.json())
  //   .then(result => console.log(result))
}

useEffect(() =>
{
  getPeople()
}, [])
```

App.js

```
import People from "../People.js"
```

Learn more React at <https://react.dev/learn>

Local Server and Build Tools

- There are many build tools available these days and you've probably heard of some of them like Webpack, Vite, Gulp, Parcel, etc.
- Vite is a build tool that will build the final website from web code like HTML/CSS/JS.
- Vite can add plugins in order to handle more features. (GLSL and React)
- Vite also do a bunch of things like optimizations, cache breaking, source mapping, running a local server, etc.
- Vite was created by Evan You, the creator of Vue.js

Go Live

- Traditional Build
 - Run npm run build
 - Upload all file from the /dist/
- Modern hosting solution : Vercel, Netlify, GitHub pages
 - Continuous integration (automatization of testing, deployment, etc.,)
 - Developer friendly
 - Easy setup
 - For complex and simple projects

Modified in `package.json`

```
"scripts": {  
  "dev": "vite",  
  "build": "vite build",  
},
```

`vite.config.js`

```
import restart from 'vite-plugin-restart'  
  
export default {  
  root: 'src/', // Sources files  
  publicDir: '../static/', // Path from "root" to static assets  
  server:  
  {  
    host: true, // Open to local network and display URL  
    open: !('SANDBOX_URL' in process.env || 'CODESANDBOX_HOST' in process.env)  
  },  
  build:  
  {  
    outDir: '../dist', // Output in the dist/ folder  
    emptyOutDir: true, // Empty the folder first  
    sourcemap: true // Add sourcemap  
  },  
  plugins:  
  [  
    restart({ restart: [ '../static/**', ] })  
  ],  
}
```

Vercel

- Automatically fetch your repository
- Help setup the live version
- Automatically update it when you push new versions on the repository
- How to use Vercel
 - Sign up to Vercel
 - Vercel is available as an NPM module `npm install vercel`
 - In `package.json` add a new script named `"deploy"` and write `"vercel --prod"` as the value
 - Deploy the project online use : `npm run deploy`
 - If an error occurs, run the following command: `npx vercel login`
 - Select Log in type

Modified in `package.json`

```
"scripts": {  
  "dev": "vite",  
  "build": "vite build",  
  "deploy": "vercel --prod"  
},
```

Example Deployment to Vercel

```
> npm run deploy
```

```
> counter@0.0.0 deploy
```

```
> vercel --prod
```

Vercel CLI 41.0.3

? Set up and deploy “D:\FIBO\code\FRA502\react\counter”? yes

? Which scope should contain your project? SEALAB-FIBO's projects

? Link to existing project? no

? What's your project's name? counter-react


? In which directory is your code located? ./


- Install Command: `yarn install`, `pnpm install`, `npm install`, or `bun install`

- Output Directory: dist

? Want to modify these settings? no

 Linked to sealab-fibos-projects/counter-react (created .vercel and added it to .gitignore)

 Inspect: <https://vercel.com/sealab-fibos-projects/counter-react/8dU4zguJZEZon8sZ4RqRif7YutJL> [5s]

 Production: <https://counter-react-55xb5ildt-sealab-fibos-projects.vercel.app> [5s]

Fun 04: Counter with React

- ให้ปรับเว็บไซต์ Fun 03 โดยใช้ React ให้สามารถนับจำนวนผู้ใช้บริการแยก ชาย-หญิง ได้

- ตัวเลขแสดงผลจำนวนผู้ใช้บริการ
- ปุ่มเพิ่มจำนวน
- ปุ่มลดจำนวน
- ปุ่มบันทึก เมื่อกดแล้วจะบันทึกผลจำนวนผู้ใช้บริการ พร้อมวันเวลาที่บันทึก
- ปุ่มรีเซ็ตให้จำนวนเป็น 0
- ส่วนแสดงผลการบันทึกข้อมูล

Counter

Man
7
UP DOWN

Woman
9
UP DOWN

Save Reset

[Mon Jan 27 2025 20:52:43] : M:11, F:10, T:21
[Mon Jan 27 2025 20:53:24] : M:15, F:20, T:35
[Mon Jan 27 2025 20:54:13] : M:4, F:4, T:8
[Mon Jan 27 2025 20:54:35] : M:4, F:3, T:7