



FPT POLYTECHNIC



THỰC HỌC – THỰC NGHIỆP



Conceive Design Implement Operate

LẬP TRÌNH FRONT-END FRAMEWORK 2

**REACT COMPONENT: STATE
COMPONENT'S LIFECYCLE**

- ⊙ Component: state
- ⊙ Component's Lifecycle





STATE

- ❑ State là 1 đối tượng javascript (javascript object) dùng lưu trữ dữ liệu động của component và cho phép component theo dõi những thay đổi giữa các lần render (có nghĩa là khi state object thay đổi, component sẽ re-render)

Luật của State

Chỉ có thể sử dụng với class component

Dễ nhầm lẫn giữa props và state

'state' là đối tượng javascript chứa dữ liệu liên quan đến component

Khi cập nhật 'state', component sẽ rerender (kết xuất lại)

'state' phải được cài đặt khi component được tạo

'state' có thể cập nhật khi dùng 'setState'

```
class Car extends React.Component {
```

```
  constructor(props) {
```

```
    super(props);
```

```
    this.state = {
```

```
      brand: "Ford",
```

```
      model: "Mustang",
```

```
      color: "red",
```

```
      year: 1964
```

```
    };
```

```
  }
```

```
  render() {
```

```
    return (
```

```
      <div>
```

```
        <h1>My {this.state.brand}</h1>
```

```
        <p>
```

```
          It is a {this.state.color}
```

```
          {this.state.model}
```

```
          from {this.state.year}.
```

```
        </p>
```

```
      </div>
```

```
    );
```

```
  }
```

```
}
```

*Chỉ có thể sử dụng
trong class component*

Cài đặt khi component được tạo

- ❑ Để thay đổi 1 giá trị trong state object, sử dụng `this.setState()`
- ❑ Khi 1 giá trị trong `state` object thay đổi, component sẽ re-render (xuất lại, vẽ lại), nghĩa là dữ liệu ra (output) sẽ thay đổi theo giá trị mới.

```
class Car extends React.Component {  
  constructor(props) {  
    super(props);  
    this.state = {  
      brand: "Ford",  
      model: "Mustang",  
      color: "red",  
      year: 1964  
    };  
  }  
  changeColor = () => {  
    this.setState({color: "blue"});  
  }  
  render() {  
    return (  
      <div>  
        <h1>My {this.state.brand}</h1>  
        <p>  
          It is a {this.state.color}  
            {this.state.model}  
          from {this.state.year}.  
        </p>  
        <button  
          type="button"  
          onClick={this.changeColor}  
        >Change color</button>  
      </div>  
    );  
  }  
}
```

*Thay đổi giá trị trong
state với setState()*

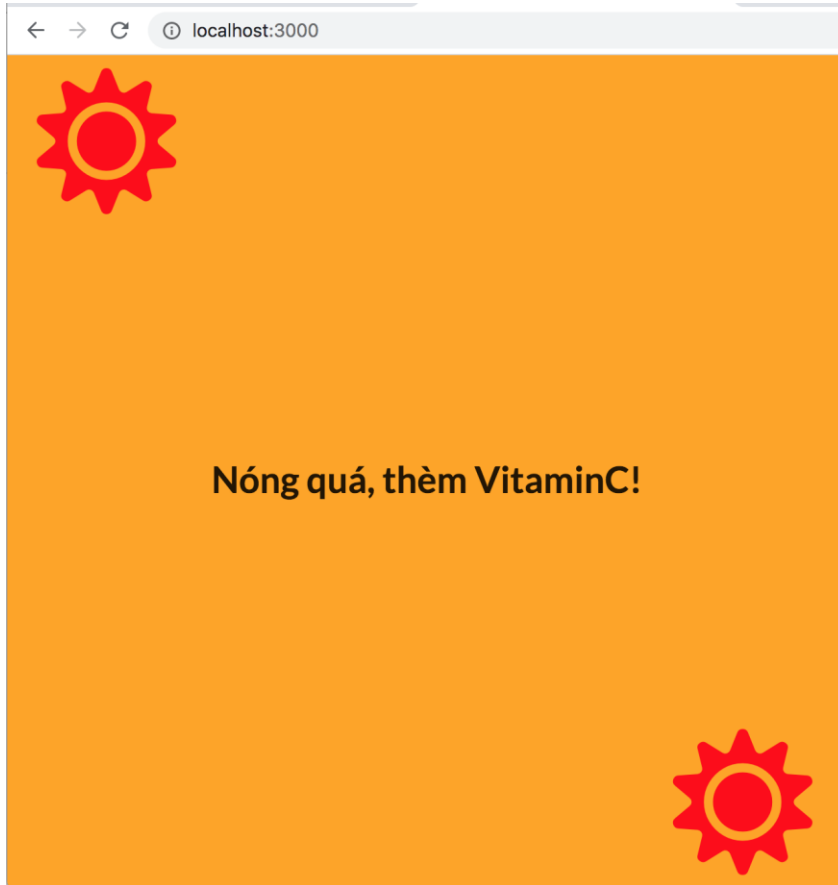
SO SÁNH PROPS VÀ STATE

	Props	State
Có thể nhận giá trị ban đầu từ component cha	✓	✓
Có thể thay đổi bởi component cha	✓	X
Có thể đặt giá trị mặc định bên trong component	✓	✓
Có thể thay đổi bên trong component	X	✓
Có thể đặt giá trị ban đầu cho các component con	✓	✓
Có thể thay đổi trong các component con	✓	X

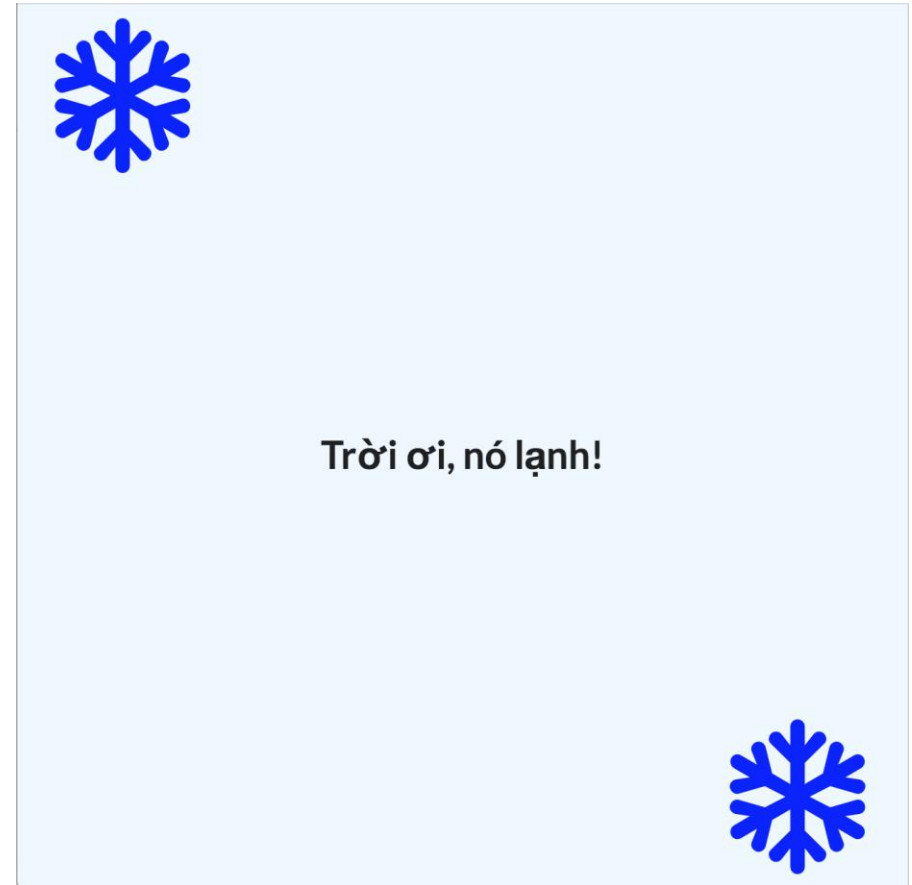


STATE - APP WEATHER

STATE – APP WEATHER

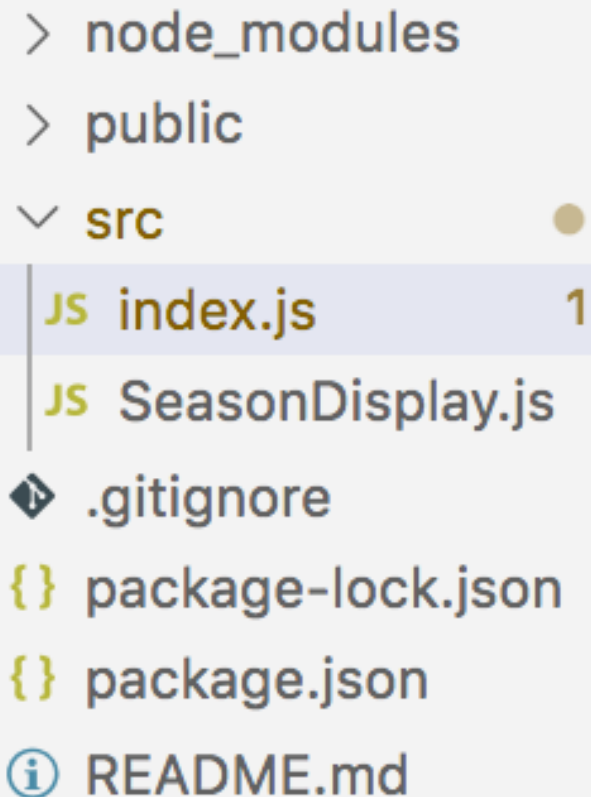


❖ Nam bán cầu – mùa hè



❖ Bắc bán cầu – mùa đông

❑ Cấu trúc project



```
> node_modules
> public
▼ src ●
  JS index.js 1
  JS SeasonDisplay.js
  .gitignore
  {} package-lock.json
  {} package.json
  ⓘ README.md
```

The image shows a file explorer interface with a tree view of a project. The 'src' directory is expanded, showing two JavaScript files: 'index.js' (with a line count of 1) and 'SeasonDisplay.js'. Below the 'src' directory are several other files: '.gitignore', 'package-lock.json', 'package.json', and 'README.md'. Each file has a small icon to its left representing its type (e.g., a diamond for .gitignore, curly braces for JSON files, and an 'i' in a circle for a markdown file).

JS SeasonDisplay.js 1 ×

src > JS SeasonDisplay.js > ...

```
1  import React from 'react';
2
3  const SeasonDisplay = () => {
4    |   return <div>Season Display</div>;
5  };
6
7  export default SeasonDisplay;
8
```

❑ Location (thay đổi)

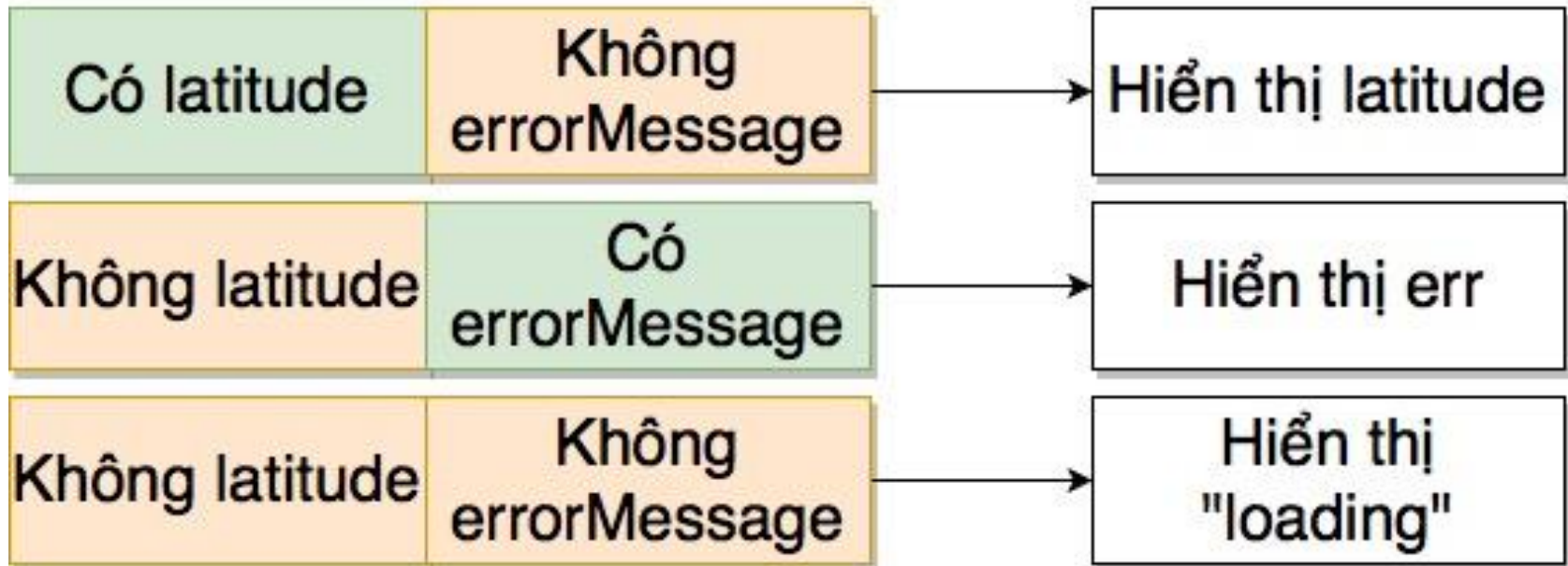
Khởi tạo

Cập nhật

```
constructor(props) {  
  super(props);  
  
  // THIS IS THE ONLY TIME we do direct assignment  
  // to this.state  
  this.state = { lat: null, errorMessage: '' };  
  
  window.navigator.geolocation.getCurrentPosition(  
    position => {  
      // we called setState!!!!  
      this.setState({ lat: position.coords.latitude });  
    },  
    err => {  
      this.setState({ errorMessage: err.message });  
    }  
  );  
}
```

ĐIỀU KIỆN RENDER NỘI DUNG

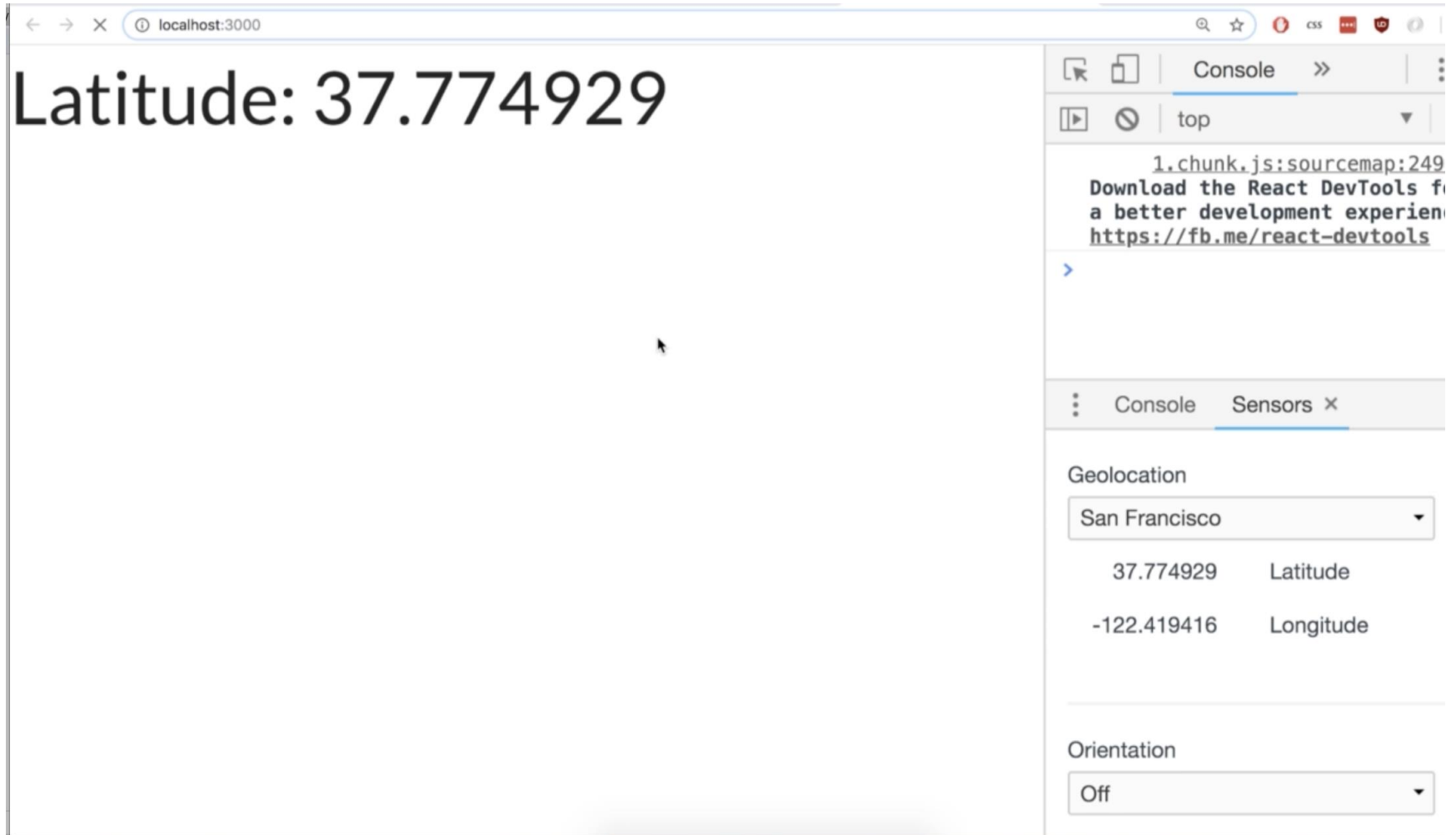
latitude: tọa độ
errorMessage: lỗi



```
// React says we have to define render!!
render() {
  if (this.state.errorMessage && !this.state.lat) {
    return <div>Error: {this.state.errorMessage}</div>;
  }

  if (!this.state.errorMessage && this.state.lat) {
    return <div>Latitude: {this.state.lat}</div>;
  }

  return <div>Loading!</div>;
}
```

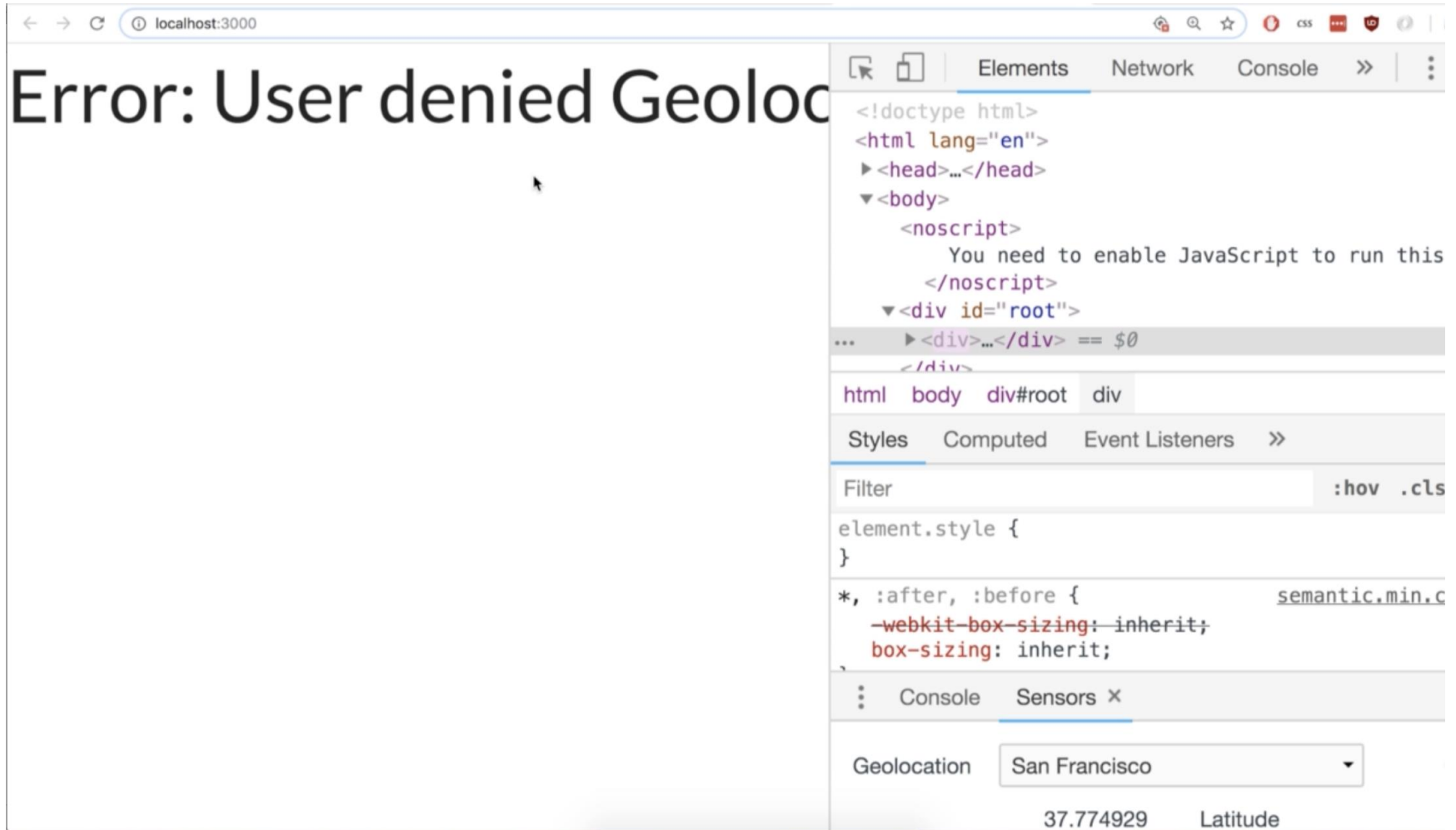
The screenshot shows a web browser at `localhost:3000` displaying the text "Latitude: 37.774929". To the right, the React DevTools interface is open, showing the Console and Sensors panels.

Console:

- Message: `1.chunk.js:sourcemap:249`
- Text: "Download the React DevTools for a better development experience." followed by the link <https://fb.me/react-devtools>

Sensors:

- Geolocation:**
 - Location: San Francisco
 - Latitude: 37.774929
 - Longitude: -122.419416
- Orientation:**
 - Orientation: Off



The screenshot shows a web browser window at `localhost:3000` displaying a large error message: "Error: User denied Geoloc". The browser's developer tools are open on the right side, showing the **Elements** panel. The DOM tree indicates a JavaScript error: `<div>...</div> == $0`. Below the error, the **Styles** panel shows the default `element.style` and `semantic.min.c` rules for `box-sizing`.

At the bottom of the developer tools, a **Geolocation** section is visible, showing a dropdown menu with "San Francisco" selected and a latitude value of "37.774929".

The screenshot shows a web browser at `localhost:3000`. A location permission dialog is displayed, asking "http://localhost:3000 wants to Know your location" with "Block" and "Allow" buttons. The page content shows the word "Loading!". The developer tools are open, showing the `Elements` panel with the following HTML structure:

```

<!doctype html>
<html lang="en">
  <head>...</head>
  <body>
    <noscript>
      You need to enable JavaScript to run th
    </noscript>
    <div id="root">
      ... <div>Loading!</div> == $0
    </div>
  
```

The `Styles` panel shows the default styles for the selected `div` element:

```

element.style {
}

*, :after, :before {
  -webkit-box-sizing: inherit;
  box-sizing: inherit;
}

```

The `Console` panel shows the `Geolocation` API with a dropdown menu set to "San Francisco" and a latitude value of "37.774929".



CHUYỂN STATE NHƯ PROPS

EXPLORER

...

JS index.js src

×

JS SeasonDisp... 1

SEASONS 2

> node_modules

> public

▼ src

JS index.js

JS SeasonDisplay... 1

📁 .gitignore

{ } package-lock.json

{ } package.json

📄 README.md

JS index.js

JS SeasonDisplay.js 1 ×

src > JS SeasonDisplay.js > ...

```

1  import React from 'react';
2
3  const SeasonDisplay = (props) => {
4    console.log(props.lat);
5    return <div>Season Display</div>;
6  };
7
8  export default SeasonDisplay;
9  
```

EXPLORER

...

OPEN EDITORS

× JS index.js src 1

JS SeasonDisplay.js...

SEASONS 2

> node_modules

> public

src

JS index.js 1

JS SeasonDisplay.js

.gitignore

package-lock.json

package.json

README.md

JS index.js 1 ×

JS SeasonDisplay.js

src > JS index.js > ...

```

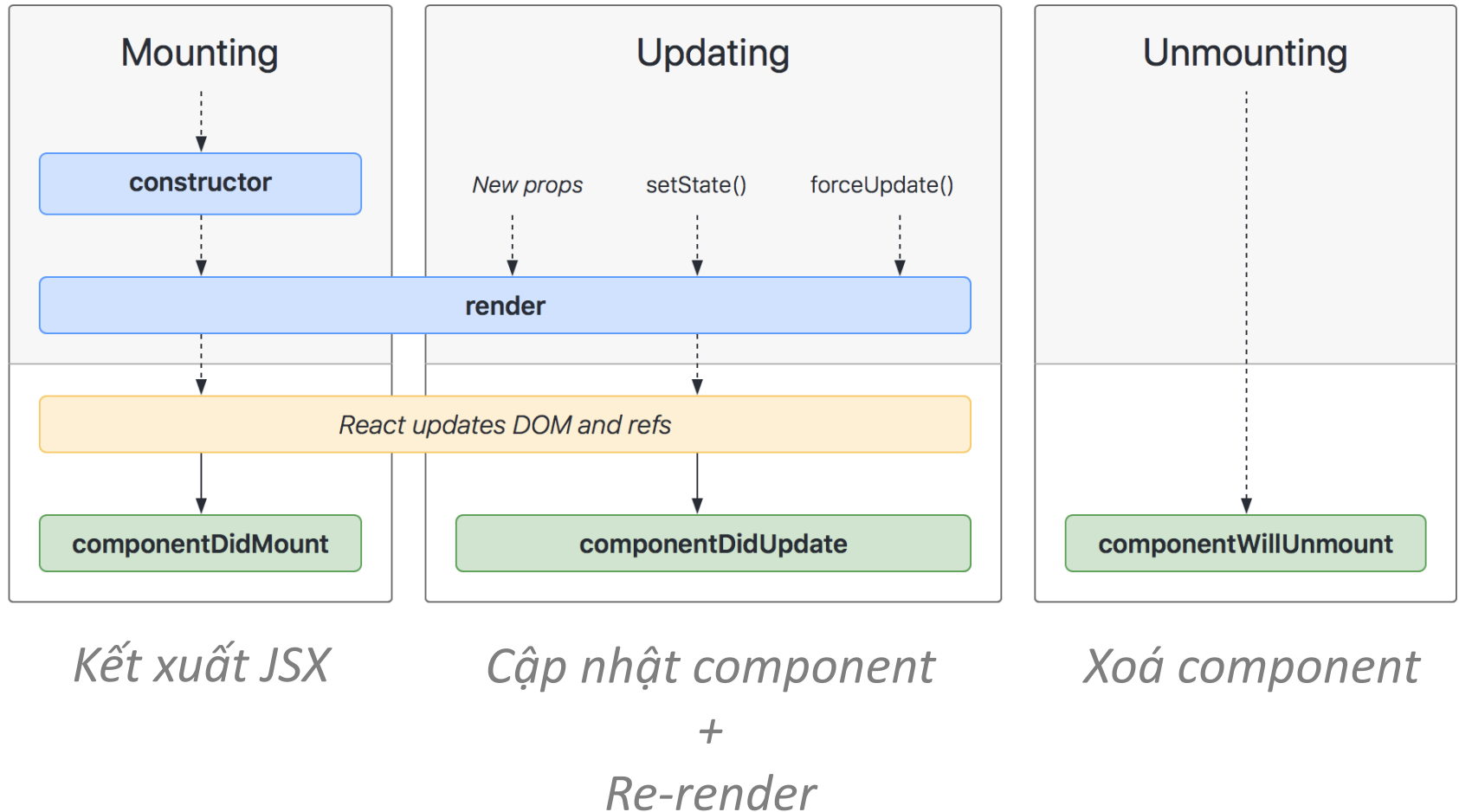
14     position => {
15         // we called setState!!!!
16         this.setState({ lat: position.coords.latitude });
17     },
18     err => {
19         this.setState({ errorMessage: err.message });
20     }
21 );
22 }
23
24 // React says we have to define render!!
25 render() {
26     if (this.state.errorMessage && !this.state.lat) {
27         return <div>Error: {this.state.errorMessage}</div>;
28     }
29
30     if (!this.state.errorMessage && this.state.lat) {
31         return <SeasonDisplay lat={this.state.lat} />;
32     }
33
34     return <div>Loading!</div>;
35 }
36 }
37
38 ReactDOM.render(<App />, document.querySelector('#root'));
39

```



LIFECYCLE METHOD

- ❑ Mỗi 1 component trong react có 1 vòng đời (lifecycle) phát triển của riêng nó.
- ❑ Lifecycle mỗi component có 3 giai đoạn:
mounting, updating, unmounting



□ Mounting: kết xuất JSX thành DOM.

- ❖ `constructor()`
- ❖ `render()`
- ❖ `componentDidMount()`

❑ Constructor(): được tạo trước hết

- ❖ Khởi tạo component
- ❖ Cài đặt state và các giá trị ban đầu.
- ❖ Gọi props (như là đối số)
- ❖ Sử dụng `super(props)` trước tất cả các khởi tạo khác để
 - Khởi tạo constructor cha
 - Cho phép các component kế thừa các phương thức từ cha của nó

❑ Render(): bắt buộc luôn được gọi (call)

- ❖ Hiển thị JSX thành DOM

```
class Header extends React.Component {  
  constructor(props) {  
    super(props);  
    this.state = {favoritecolor: "red"};  
  }  
  render() {  
    return (  
      <h1>My Favorite Color is {this.state.favoritecolor}</h1>  
    );  
  }  
}
```

```
ReactDOM.render(<Header />, document.getElementById('root'));
```

□ componentDidMount()

- ❖ Được gọi 1 lần trong lifecycle
- ❖ Được gọi sau khi component được kết xuất (rendered)

□ Ví dụ

```
import React from 'react';
import ReactDOM from 'react-dom';

class Header extends React.Component {
  constructor(props) {
    super(props);
    this.state = {favoritecolor: "red"};
  }
  componentDidMount() {
    setTimeout(() => {
      this.setState({favoritecolor: "yellow"})
    }, 1000)
  }
  render() {
    return (
      <h1>My Favorite Color is {this.state.favoritecolor}</h1>
    );
  }
}

ReactDOM.render(<Header />, document.getElementById('root'));
```

❑ Updating: khi component được cập nhật.

- ❖ `render()`

- ❖ `componentDidUpdate()`

❑ `Render()`:

- ❖ Được gọi khi component được cập nhật

- ❖ Kết xuất lại những thay đổi mới

❑ `componentDidUpdate()`

- ❖ Được gọi sau khi component được cập nhật

```
class Header extends React.Component {
  constructor(props) {
    super(props);
    this.state = {favoritecolor: "red"};
  }
  componentDidMount() {
    setTimeout(() => {
      this.setState({favoritecolor: "yellow"}), 1000)
    }
  }
  componentDidUpdate() {
    document.getElementById("mydiv").innerHTML =
      "The updated favorite is " + this.state.favoritecolor;
  }
  render() {
    return (
      <div>
        <h1>My Favorite Color is {this.state.favoritecolor}</h1>
        <div id="mydiv"></div>
      </div>
    );
  }
}
```

1. constructor
Render: My Favourite Color is red

2. componentDidMount
Render: My Favourite Color is yellow

3. componentDidUpdate
(red to yellow)
Render: The updated favourite is yellow

```
ReactDOM.render(<Header />, document.getElementById('root'));
```

□ Unmounting:

- ❖ Component được xóa khỏi DOM
- ❖ Kết thúc lifecycle của component
- ❖ `componentWillUnmount()`


```
class Container extends React.Component {
  constructor(props) {
    super(props);
    this.state = {show: true};
  }
  delHeader = () => {
    this.setState({show: false});
    console.log("delHeader");
  }
  componentDidMount() {
    console.log("ComponentDidAmount:Container");
  }

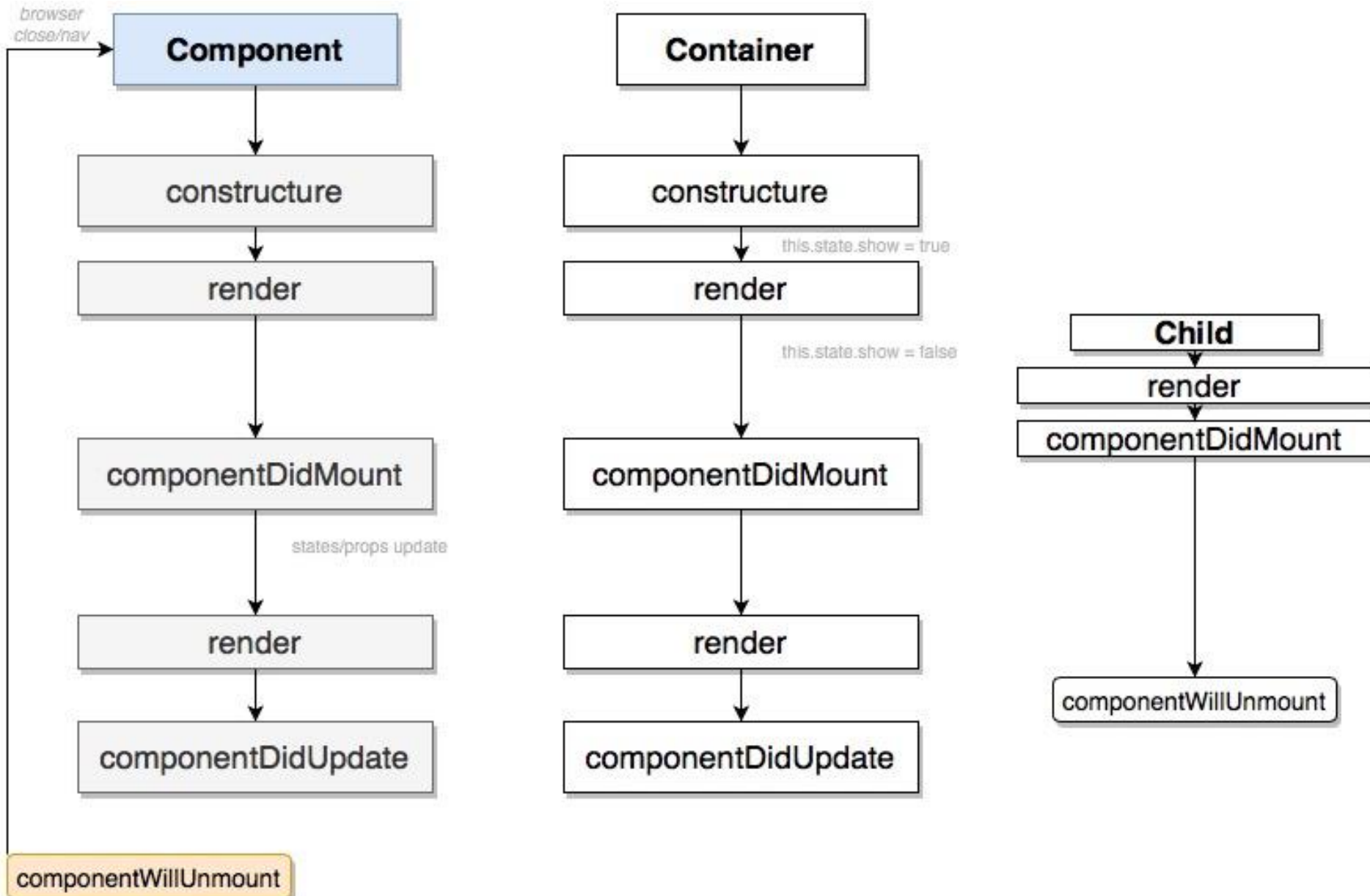
  componentDidUpdate() {
    console.log("ComponentDidUpdate:Container")
  }

  render() {
    let myheader;
    if (this.state.show) {
      myheader = <Child />;
    };
    return (
      <div>
        {myheader}
        <button type="button" onClick={this.delHeader}>
          Delete Header
        </button>
      </div>
    );
  }
}
```

```
class Child extends React.Component {
  componentDidMount() {
    console.log("ComponentDidAmount: Child");
  }

  componentWillUnmount() {
    console.log("The component is about to be unmounted: Child");
  }
  render() {
    return (
      <h1>Hello World!</h1>
    );
  }
}
```

```
ReactDOM.render(<Container />, document.getElementById('root'));
```



React App

×

+

←

→

↻

localhost:3000

☆

Ⓣ Paused

⋮

Delete Header

⌕

📄

Elements

Console

»

⚠️ 1

⚙️

⋮

×

▶

🚫

top

▼

👁️

Filter

Default levels

⚙️

[HMR] Waiting for update signal from WDS... log.js:24

⚠️

▶ [Deprecation] scheduler.development.js:305
 SharedArrayBuffer will require cross-origin isolation as of M91, around May 2021. See <https://developer.chrome.com/blog/enabling-shared-array-buffer/> for more details.

react-dom.development.js:24994

Download the React DevTools for a better development experience: <https://fb.me/react-devtools>

ComponentDidAmount: Child index.js:41

ComponentDidAmount:Container index.js:16

delHeader index.js:13

The component is about to be unmounted: Child index.js:45

ComponentDidUpdate:Container index.js:20

>



LIFECYCLE – APP WEATHER

index.js — seasons

EXPLORER

OPEN EDITORS

JS index.js src 1

SEASONS

node_modules

public

src

JS index.js 1

SeasonDisplay.css

JS SeasonDisplay.js

JS Spinner.js

.gitignore

package-lock.json

package.json

README.md

JS index.js 1 X

src > JS index.js > ...

1 import React from 'react';

2 import ReactDOM from 'react-dom';

3 import SeasonDisplay from './SeasonDisplay';

4 import Spinner from './Spinner';

5

6 class App extends React.Component {

7 state = { lat: null, errorMessage: '' };

8

9 componentDidMount() {

10 window.navigator.geolocation.getCurrentPosition(

11 position => this.setState({ lat: position.coords.latitude }),

12 err => this.setState({ errorMessage: err.message })

13 });

14 }

15

16 renderContent() {

17 if (this.state.errorMessage && !this.state.lat) {

18 return <div>Error: {this.state.errorMessage}</div>;

19 }

20

21 if (!this.state.errorMessage && this.state.lat) {

22 return <SeasonDisplay lat={this.state.lat} />;

23 }

24

25 return <Spinner message="Please accept location request" />;

26 }

27

28 render() {

29 return <div className="border red">{this.renderContent()}</div>;

30 }

31 }

32

33 ReactDOM.render(<App />, document.querySelector('#root'));

thank
you!