React - ES6 Classes

Topics : React JS
Written on January 01, 2024

In React, ES6 classes are commonly used to define components, especially class components. Class components have been a fundamental part of React since its early versions. Here's a brief overview of how ES6 classes are used in React:

Class Components:

In React, components can be defined as ES6 classes that extend from `React.Component`. These class components have a `render` method that returns the JSX representing the component's UI. Here's a basic example:

jsx code

```
import React, { Component } from 'react';

class MyComponent extends Component {
  render() {
    return <div>Hello, React!</div>;
  }
}

export default MyComponent;
```

In the above example:

- `MyComponent` is a class that extends `React.Component`.
- The `render` method is where you define what the component should render. It returns JSX, which gets converted to HTML.

Constructor and State:

You can use the constructor in a class component to initialize state or bind methods. State is an important concept in React for managing the internal state of a component.

jsx code

```
import React, { Component } from 'react';

class Counter extends Component {
  constructor(props) {
```

```
    super(props);
    this.state = {
      count: 0,
    };
  }

  render() {
    return (
      <div>
        <p>Count: {this.state.count}</p>
        <button onClick={() => this.setState({ count: this.state.count + 1
})}>
          Increment
        </button>
      </div>
    );
  }
}

export default Counter;
```

Event Handling:

In class components, you often define event handlers as methods within the class. For example:

jsx code

```
import React, { Component } from 'react';

class ButtonClicker extends Component {
  handleClick = () => {
    console.log('Button clicked!');
  };

  render() {
    return <button onClick={this.handleClick}>Click me</button>;
  }
}

export default ButtonClicker;
```

Lifecycle Methods:

Class components have lifecycle methods that you can override to perform actions at specific points in a component's life cycle. Common lifecycle methods include `componentDidMount`, `componentDidUpdate`, and `componentWillUnmount`.

jsx code

```
import React, { Component } from 'react';
```

```
class LifecycleExample extends Component {
  componentDidMount() {
    console.log('Component mounted');
  }

  componentDidUpdate() {
    console.log('Component updated');
  }

  componentWillUnmount() {
    console.log('Component will unmount');
  }

  render() {
    return <div>Check the console for lifecycle messages</div>;
  }
}

export default LifecycleExample;
```