ARYATECHNO

React Portals

Topics : React JS
Written on January 03, 2024

React Portals provide a way to render children into a DOM node that exists outside the hierarchy of the parent component. This allows you to render components in a different part of the DOM, which can be useful for scenarios like modals, tooltips, or any situation where you need to render content outside the normal React component tree.

Here's a basic example of using React Portals:

```jsx
import React, { useState } from 'react';
import ReactDOM from 'react-dom';

const Modal = ({ isOpen, onClose, children }) => {
if (!isOpen) return null;

return ReactDOM.createPortal(
<div className="modal-overlay">
<div className="modal">
<button onClick={onClose}>Close</button>
{children}
</div>
</div>,
document.body // Render the modal outside the parent component's DOM hierarchy
);
};

const App = () => {
const [modalIsOpen, setModalIsOpen] = useState(false);

return (
<div>
<h1>Hello, React Portals!</h1>
<button onClick={() => setModalIsOpen(true)}>Open Modal</button>

<Modal isOpen={modalIsOpen} onClose={() => setModalIsOpen(false)}>
<p>This is the content of the modal.</p>
</Modal>
</div>
);
};

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

In this example:

1. The `Modal` component renders its content inside a `div` with the class `modal-overlay`. The entire modal is then rendered inside a `div` with the class `modal`. The `Modal` component uses `ReactDOM.createPortal` to render its content outside the normal component hierarchy.

2. The `App` component includes a button to open the modal. The `Modal` component is conditionally rendered based on the `modalIsOpen` state.

3. The `Modal` component is appended to the `document.body`, ensuring that it is rendered outside the normal React component tree.