Portal

The Portal component lets you render its children into a DOM node that exists outside of the Portal's own DOM hierarchy.

View as Markdown Feedback Bundle size

Introduction

Portal is a utility component built around React's `createPortal()` API. It gives you the functionality of `createPortal()` in a convenient component form. It's used internally by the Modal and Popper components.

> According to the React docs, portals are useful when "you need the child element to visually 'break out' of its container"—for instance, modals and tooltips, which need to exist outside of the normal flow of the document.

Normally, children of a component are rendered within that component's DOM tree. But sometimes it's necessary to mount a child at a different location in the DOM. The Portal component accepts a `container` prop that passes a `ref` to the DOM node where its children will be mounted.

The following demo shows how a `<span>` nested within a Portal can be appended to a node outside of the Portal's DOM hierarchy—click **Mount children** to see how it behaves:

Mount children

It looks like I will render here.

Edit in Chat JS TS Collapse code

```
import * as React from 'react';
import Portal from '@mui/material/Portal';
import { Box } from '@mui/system';

export default function SimplePortal() {
  const [show, setShow] = React.useState(false);
  const container = React.useRef(null);

  const handleClick = () => {
    setShow(!show);
  };

  return (
    <div>
      <button type="button" onClick={handleClick}>
        {show ? 'Unmount children' : 'Mount children'}
      </button>
      <Box sx={{ p: 1, my: 1, border: '1px solid' }}>
        It looks like I will render here.
        {show ? (
          <Portal container={() => container.current!}>
            <span>But I actually render here!</span>
          </Portal>
        ) : null}
      </Box>
      <Box sx={{ p: 1, my: 1, border: '1px solid' }} ref={container} />
    </div>
  );
}
```

Basics

Import

```
import Portal from '@mui/material/Portal';
```

Customization

Server-side Portals

The DOM API isn't available on the server, so you need to use the `container` prop callback. This callback is called during a React layout effect:

```
<Portal container={() => document.getElementById('filter-panel')!}>
  <Child />
</Portal>
```

The Portal component cannot be used to render child elements on the server—client-side hydration is necessary. This is because React doesn't support the `createPortal()` API on the server. See this GitHub issue for details.

API

See the documentation below for a complete reference to all of the props and classes available to the components mentioned here.

- `<Portal />`

Edit this page

Was this page helpful?

Popper

Textarea Autosize