Popover

A Popover can be used to display some content on top of another.

Things to know when using the `Popover` component:

- The component is built on top of the `Modal` component.
- The scroll and click away are blocked unlike with the `Popper` component.

View as Markdown | Feedback | Bundle size | Source

Basic Popover

OPEN POPOVER

Edit in Chat | JS | TS | Collapse code

```tsx
import * as React from 'react';
import Popover from '@mui/material/Popover';
import Typography from '@mui/material/Typography';
import Button from '@mui/material/Button';

export default function BasicPopover() {
  const [anchorEl, setAnchorEl] = React.useState<HTMLButtonElement | null>(null);

  const handleClick = (event: React.MouseEvent<HTMLButtonElement>) => {
    setAnchorEl(event.currentTarget);
  };

  const handleClose = () => {
    setAnchorEl(null);
  };

  const open = Boolean(anchorEl);
  const id = open ? 'simple-popover' : undefined;

  return (
```

```
  <div>
    <Button aria-describedby={id} variant="contained" onClick={handleClick}>
      Open Popover
    </Button>
    <Popover
      id={id}
      open={open}
      anchorEl={anchorEl}
      onClose={handleClose}
      anchorOrigin={{
        vertical: 'bottom',
        horizontal: 'left',
      }}
    >
      <Typography sx={{ p: 2 }}>The content of the Popover.</Typography>
    </Popover>
  </div>
);
```

Anchor playground

Use the radio buttons to adjust the `anchorOrigin` and `transformOrigin` positions. You can also set the `anchorReference` to `anchorPosition` or `anchorEl`. When it is `anchorPosition`, the component will, instead of `anchorEl`, refer to the `anchorPosition` prop which you can adjust to set the position of the popover.

anchorReference

(•) anchorEl ( ) anchorPosition

anchorPosition.top

200

anchorPosition.left

400

anchorOrigin.vertical

(•) Top

( ) Center

( ) Bottom

transformOrigin.vertical

(•) Top

( ) Center

( ) Bottom

anchorOrigin.horizontal

(•) Left ( ) Center ( ) Right

transformOrigin.horizontal

(•) Left ( ) Center ( ) Right

```
<Popover
  anchorOrigin={{
    vertical: 'top',
    horizontal: 'left',
```

Copy

```
    }}
    transformOrigin={{
      vertical: 'top',
      horizontal: 'left',
    }}
  >
    The content of the Popover.
  </Popover>
```

Mouse hover interaction

This demo demonstrates how to use the `Popover` component with `mouseenter` and `mouseleave` events to achieve popover behavior.

Hover with a Popover.

Edit in Chat | JS | TS | Hide code

```
import * as React from 'react';
import Popover from '@mui/material/Popover';
import Typography from '@mui/material/Typography';

export default function MouseHoverPopover() {
  const [anchorEl, setAnchorEl] = React.useState<HTMLElement | null>(null);

  const handlePopoverOpen = (event: React.MouseEvent<HTMLElement>) => {
    setAnchorEl(event.currentTarget);
  };

  const handlePopoverClose = () => {
    setAnchorEl(null);
  };

  const open = Boolean(anchorEl);

  return (
    <div>
      <Typography
        aria-owns={open ? 'mouse-over-popover' : undefined}
        aria-haspopup="true"
        onMouseEnter={handlePopoverOpen}
        onMouseLeave={handlePopoverClose}
      >
        Hover with a Popover.
      </Typography>
      <Popover
        id="mouse-over-popover"
        sx={{ pointerEvents: 'none' }}
        open={open}
        anchorEl={anchorEl}
```

```
        anchorOrigin={{
          vertical: 'bottom',
          horizontal: 'left',
        }}
        transformOrigin={{
          vertical: 'top',
```

Virtual element

The value of the `anchorEl` prop can be a reference to a fake DOM element. You need to provide an object with the following interface:

```
interface PopoverVirtualElement {
  nodeType: 1;
  getBoundingClientRect: () => DOMRect;
}
```

Highlight part of the text to see the popover:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam ipsum purus, bibendum sit amet vulputate eget, porta semper ligula. Donec bibendum vulputate erat, ac fringilla mi finibus nec. Donec ac dolor sed dolor porttitor blandit vel vel purus. Fusce vel malesuada ligula. Nam quis vehicula ante, eu finibus est. Proin ullamcorper fermentum orci, quis finibus massa. Nunc lobortis, massa ut rutrum ultrices, metus metus finibus ex, sit amet facilisis neque enim sed neque. Quisque accumsan metus vel maximus consequat. Suspendisse lacinia tellus a libero volutpat maximus.

Edit in Chat | JS | TS | Hide code

```
import * as React from 'react';
import Popover, { PopoverProps } from '@mui/material/Popover';
import Typography from '@mui/material/Typography';
import Paper from '@mui/material/Paper';

export default function VirtualElementPopover() {
  const [open, setOpen] = React.useState(false);
  const [anchorEl, setAnchorEl] = React.useState<PopoverProps['anchorEl']>(null);

  const handleClose = () => {
    setOpen(false);
  };

  const handleMouseUp = () => {
    const selection = window.getSelection();

    // Skip if selection has a length of 0
```

```
    if (!selection || selection.anchorOffset === selection.focusOffset) {
      return;
    }

    const getBoundingClientRect = () => {
      return selection.getRangeAt(0).getBoundingClientRect();
    };

    setOpen(true);

    setAnchorEl({ getBoundingClientRect, nodeType: 1 });
  };

  const id = open ? 'virtual-element-popover' : undefined;

  return (
    <div>
      <Typography aria-describedby={id} onMouseUp={handleMouseUp}>
        Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam ipsum purus,
        bibendum sit amet vulputate eget, porta semper ligula. Donec bibendum
```

For more information on the virtual element's properties, see the following resources:

- getBoundingClientRect
- DOMRect
- Node types

> ⚠ The usage of a virtual element for the Popover component requires the `nodeType` property. This is different from virtual elements used for the `Popper` or `Tooltip` components, both of which don't require the property.

Supplementary projects

For more advanced use cases, you might be able to take advantage of:

material-ui-popup-state

Star 472 downloads 1.1M/month

The package `material-ui-popup-state` that takes care of popover state for you in most cases.

OPEN POPOVER

Edit in Chat JS TS Hide code

```tsx
import Typography from '@mui/material/Typography';
import Button from '@mui/material/Button';
import Popover from '@mui/material/Popover';
import PopupState, { bindTrigger, bindPopover } from 'material-ui-popup-state';

export default function PopoverPopupState() {
  return (
    <PopupState variant="popover" popupId="demo-popup-popover">
      {(popupState) => (
        <div>
          <Button variant="contained" {...bindTrigger(popupState)}>
            Open Popover
          </Button>
          <Popover
            {...bindPopover(popupState)}
            anchorOrigin={{
              vertical: 'bottom',
              horizontal: 'center',
            }}
            transformOrigin={{
              vertical: 'top',
              horizontal: 'center',
            }}
          >
            <Typography sx={{ p: 2 }}>The content of the Popover.</Typography>
          </Popover>
        </div>
      )}
    </PopupState>
  );
}
```

API

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

- `<Grow />`
- `<Popover />`

Edit this page

Was this page helpful?

No SSR

Popper