App Bar

The App Bar displays information and actions relating to the current screen.

The top App Bar provides content and actions related to the current screen. It's used for branding, screen titles, navigation, and actions.

It can transform into a contextual action bar or be used as a navbar.

{{"component": "modules/components/ComponentLinkHeader.js"}}

Basic App Bar

{{"demo": "ButtonAppBar.js", "bg": true}}

App Bar with menu

{{"demo": "MenuAppBar.js", "bg": true}}

App Bar with responsive menu

{{"demo": "ResponsiveAppBar.js", "bg": true}}

App Bar with search field

A side searchbar.

{{"demo": "SearchAppBar.js", "bg": true}}

App Bar with a primary search field

A primary searchbar.

{{"demo": "PrimarySearchAppBar.js", "bg": true}}

Dense (desktop only)

{{"demo": "DenseAppBar.js", "bg": true}}

Prominent

A prominent app bar.

{{"demo": "ProminentAppBar.js", "bg": true}}

Bottom App Bar

{{"demo": "BottomAppBar.js", "iframe": true, "maxWidth": 400}}

Fixed placement

When you render the app bar position fixed, the dimension of the element doesn't impact the rest of the page. This can cause some part of your content to be invisible, behind the app bar. Here are 3 possible solutions:

1. You can use `position="sticky"` instead of fixed. ⚠️ sticky is not supported by IE11.
2. You can render a second `<Toolbar />` component:

```jsx
function App() {
  return (
    <React.Fragment>
      <AppBar position="fixed">
        <Toolbar>{/* content */}</Toolbar>
      </AppBar>
      <Toolbar />
    </React.Fragment>
  );
}
```

3. You can use `theme.mixins.toolbar` CSS:

```jsx
const Offset = styled('div')(({ theme }) => theme.mixins.toolbar);

function App() {
  return (
    <React.Fragment>
      <AppBar position="fixed">
        <Toolbar>{/* content */}</Toolbar>
      </AppBar>
      <Offset />
    </React.Fragment>
  );
}
```

Scrolling

You can use the `useScrollTrigger()` hook to respond to user scroll actions.

Hide App Bar

The app bar hides on scroll down to leave more space for reading.

{{"demo": "HideAppBar.js", "iframe": true}}

Elevate App Bar

The app bar elevates on scroll to communicate that the user is not at the top of the page.

{{"demo": "ElevateAppBar.js", "iframe": true}}

Back to top

A floating action buttons appears on scroll to make it easy to get back to the top of the page.

{{"demo": "BackToTop.js", "iframe": true}}

`useScrollTrigger([options]) => trigger`

Arguments

1. `options` (*object* [optional]):
   - `options.disableHysteresis` (*bool* [optional]): Defaults to `false`. Disable the hysteresis. Ignore the scroll direction when determining the `trigger` value.
   - `options.target` (*Node* [optional]): Defaults to `window`.
   - `options.threshold` (*number* [optional]): Defaults to `100`. Change the `trigger` value when the vertical scroll strictly crosses this threshold (exclusive).

Returns

`trigger`: Does the scroll position match the criteria?

Examples

```jsx
import useScrollTrigger from '@mui/material/useScrollTrigger';

function HideOnScroll(props) {
  const trigger = useScrollTrigger();
  return (
    <Slide in={!trigger}>
      <div>Hello</div>
    </Slide>
  );
}
```

Enable color on dark

Following the Material Design guidelines, the `color` prop has no effect on the appearance of the app bar in dark mode. You can override this behavior by setting the `enableColorOnDark` prop to `true`.

{{"demo": "EnableColorOnDarkAppBar.js", "bg": true}}