Search... Ctrl+K

Links

The Link component allows you to easily customize anchor elements with your theme colors and typography styles.

View as Markdown Feedback Bundle size Source WAI-ARIA Figma Sketch

Basic links

The Link component is built on top of the Typography component, meaning that you can use its props.

Link color="inherit" variant="body2"

Edit in Chat JS TS Collapse code

```tsx
import * as React from 'react';
import Box from '@mui/material/Box';
import Link from '@mui/material/Link';

const preventDefault = (event: React.SyntheticEvent) => event.preventDefault();

export default function Links() {
  return (
    <Box
      sx={{
        typography: 'body1',
        '& > :not(style) ~ :not(style)': {
          ml: 2,
        },
      }}
      onClick={preventDefault}
    >
      <Link href="#">Link</Link>
      <Link href="#" color="inherit">
        {'color="inherit"'}
      </Link>
      <Link href="#" variant="body2">
        {'variant="body2"'}
      </Link>
    </Box>
  );
}
```

However, the Link component has some different default props than the Typography component:

- `color="primary"` as the link needs to stand out.
- `variant="inherit"` as the link will, most of the time, be used as a child of a Typography component.

Underline

The `underline` prop can be used to set the underline behavior. The default is `always`.

underline="none" underline="hover" underline="always"

Edit in Chat JS TS Collapse code

```tsx
import * as React from 'react';
import Box from '@mui/material/Box';
import Link from '@mui/material/Link';

const preventDefault = (event: React.SyntheticEvent) => event.preventDefault();

export default function UnderlineLink() {
  return (
    <Box
      sx={{
        display: 'flex',
        flexWrap: 'wrap',
        justifyContent: 'center',
        typography: 'body1',
        '& > :not(style) ~ :not(style)': {
          ml: 2,
        },
      }}
      onClick={preventDefault}
    >
      <Link href="#" underline="none">
        {'underline="none"'}
      </Link>
      <Link href="#" underline="hover">
        {'underline="hover"'}
      </Link>
      <Link href="#" underline="always">
        {'underline="always"'}
      </Link>
    </Box>
  );
}
```

Security

When you use `target="_blank"` with Links, it is [recommended](#) to always set `rel="noopener"` or `rel="noreferrer"` when linking to third party content.

- `rel="noopener"` prevents the new page from being able to access the `window.opener` property and ensures it runs in a separate process. Without this, the target page can potentially redirect your page to a malicious URL.
- `rel="noreferrer"` has the same effect, but also prevents the *Referer* header from being sent to the new page. ⚠️ Removing the referrer header will affect analytics.

Third-party routing library

One frequent use case is to perform navigation on the client only, without an HTTP round-trip to the server. The `Link` component provides the `component` prop to handle this use case. Here is a [more detailed guide](#).

Accessibility

(WAI-ARIA: https://www.w3.org/WAI/ARIA/apg/patterns/link/ ↗)

- When providing the content for the link, avoid generic descriptions like "click here" or "go to". Instead, use [specific descriptions](#).
- For the best user experience, links should stand out from the text on the page. For instance, you can keep the default `underline="always"` behavior.
- If a link doesn't have a meaningful href, [it should be rendered using a `<button>` element](#). The demo below illustrates how to properly link with a `<button>`:

[Button Link](#)

Edit in Chat JS TS Collapse code

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

export default function ButtonLink() {
  return (
    <Link
      component="button"
```

```
        variant="body2"
        onClick={() => {
          console.info("I'm a button.");
        }}
      >
        Button Link
      </Link>
    );
  }
```

Keyboard accessibility

- Interactive elements should receive focus in a coherent order when the user presses the `Tab` key.
- Users should be able to open a link by pressing `Enter`.

Screen reader accessibility

- When a link receives focus, screen readers should announce a descriptive link name. If the link opens in a new window or browser tab, add an `aria-label` ↗ to inform screen reader users—for example, *"To learn more, visit the About page which opens in a new window."*

API

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

- `<Link />`

Edit this page

Was this page helpful?