Button

Botões permitem que os usuários tomem ações e decisões com um simples toque.

Botões comunicam ações que os usuários podem realizar. Eles são normalmente colocados em toda a interface do usuário, em lugares como:

- Janelas modais
- Formulários
- Cartões
- Barras de ferramentas

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

Basic Button

O `Botão` vem com três variantes: texto (padrão), contido e delineado.

{{"demo": "BasicButtons.js"}}

Botões de texto

Text buttons are typically used for less-pronounced actions, including those located: in dialogs, in cards. Em cartões, os botões de texto ajudam a manter a ênfase no conteúdo do cartão.

{{"demo": "TextButtons.js"}}

Botões contidos

Botões Contidos tem alta ênfase, distinguem-se pelo uso de elevação e preenchimento. Eles contém as principais ações da sua aplicação.

{{"demo": "ContainedButtons.js"}}

Você pode remover a elevação com a propriedade `disableElevation`.

{{"demo": "DisableElevation.js"}}

Botões delineados

Outlined buttons are medium-emphasis buttons. They contain actions that are important but aren't the primary action in an app.

Botões delineados são uma alternativa de menor ênfase comparado com botões contidos, ou uma uma alternativa de maior ênfase comparado com botões de texto.

{{"demo": "OutlinedButtons.js"}}

Manipulando cliques

Todos os componentes aceitam um método manipulador `onClick` que é aplicado ao elemento DOM raiz.

```
<Button
  onClick={() => {
    alert('clicado');
```

```
  }}
>
  Clique aqui
</Button>
```

Note que a documentação evita mencionar as propriedades nativas (existem várias) na seção de API dos componentes.

Cor

{{"demo": "ColorButtons.js"}}

Além de usar as cores de botão padrão, você pode adicionar outras personalizadas ou desativar as que não forem necessárias. See the Adding new colors example for more info.

Tamanhos

For larger or smaller buttons, use the `size` prop.

{{"demo": "ButtonSizes.js"}}

Botão de upload

{{"demo": "UploadButtons.js"}}

Botões com ícones e rótulo

Às vezes você pode querer ter ícones para certos botões para aprimorar a experiência do usuário, pois reconhecem logotipos mais facilmente do que texto. Por exemplo, se você tem um botão com a ação de "deletar", você pode rotulá-lo com o ícone de lata de lixo.

{{"demo": "IconLabelButtons.js"}}

Ícone do botão

Botões de ícones são comumente encontrados em barras de aplicativos e barras de ferramentas.

Ícones são também adequados para botões de alternância que permitem uma escolha única para ser selecionado ou desmarcado, como adicionar ou remover uma estrela para um item.

{{"demo": "IconButtons.js"}}

Tamanhos

For larger or smaller icon buttons, use the `size` prop.

{{"demo": "IconButtonSizes.js"}}

Cores

Use a propriedade `color` para aplicar uma paleta de cores ao componente.

{{"demo": "IconButtonColors.js"}}

Botões customizados

Aqui estão alguns exemplos de customização do componente. Você pode aprender mais sobre isso na página de documentação de sobrescritas.

{{"demo": "CustomizedButtons.js", "defaultCodeOpen": false}}

🎨 If you are looking for inspiration, you can check MUI Treasury's customization examples.

Botão de carregamento

Os botões de carregamento podem mostrar estado de carregamento e desativar as interações.

{{"demo": "LoadingButtons.js"}}

Alterne o interruptor de carregamento para ver a transição entre os diferentes estados.

{{"demo": "LoadingButtonsTransition.js"}}

Botão complexo

Os botões de texto, botões contidos, botões de ação flutuante e botões de ícone são construídos com base no mesmo componente: O componente `ButtonBase`. Você pode usar esse componente para construir interações diferentes.

{{"demo": "ButtonBase.js"}}

Biblioteca de roteamento de terceiros

One frequent use case is to perform navigation on the client only, without an HTTP round-trip to the server. Um caso de uso comum é usar o botão para acionar uma navegação para uma nova página. Aqui está um guia mais detalhado.

Limitações

Propriedade CSS Cursor not-allowed

O componente ButtonBase define `pointer-events: none;` ao desabilitar os botões, o que previne que o cursor desabilitado seja exibido.

Se você deseja usar `not-allowed`, você tem duas opções:

1. **Apenas com CSS**. You can remove the pointer-events style on the disabled state of the `<button>` element:

```
<span style={{ cursor: 'not-allowed' }}>
    <Button component={Link} disabled>
      disabled
    </Button>
  </span>
```

Então:

- Você deve adicionar `pointer-events: none;` de volta quando você precisar exibir dicas em elementos desabilitados.
- O cursor não irá mudar se você renderizar algum outro elemento que não seja um botão, por exemplo, um elemento link `<a>`.

2. **Alteração no DOM**. Você pode encapsular o botão:

```
<span style={{ cursor: 'not-allowed' }}>
  <Button component={Link} disabled>
    disabled
  </Button>
</span>
```

Isso tem a vantagem de suportar qualquer elemento, por exemplo, um elemento de link `<a>`.

Unstyled

The button also comes with an unstyled version. It's ideal for doing heavy customizations and minimizing bundle size.

Componente sem estilo

```
import ButtonUnstyled from '@mui/base/ButtonUnstyled';
```

{{"demo": "UnstyledButtonsSimple.js"}}

Customizando o elemento raiz

By default, the `ButtonUnstyled` renders a native `button` element. You are free to override this by setting the `component` or `components. Root` prop. If a non-interactive element (such as a span) is provided this way, the `ButtonUnstyled` will take care of adding accessibility attributes.

{{"demo": "UnstyledButtonsSpan.js"}}

Compare the attributes on the span with the button from the previous demo.

Complex customization

You are not limited to using HTML elements for the button structure. Elementos SVG, mesmo que com uma estrutura complexa, são igualmente aceitáveis.

{{"demo": "UnstyledButtonCustom.js"}}

useButton hook

```
import { useButton } from '@mui/base/ButtonUnstyled';
```

If you need to use Button's functionality in another component, you can use the `useButton` hook. It returns props to be placed on a custom button element and fields representing the internal state of the button.

The `useButton` hook requires the ref of the element it'll be used on. Additionally, you need to provide the `component` prop (unless you intend to use the plain `button`).

{{"demo": "UseButton.js"}}