

# Snackbar

Snackbars fornecem mensagens breves sobre os processos de aplicativos. O componente também é conhecido como toast(torrada).

[Snackbars](#) informam aos usuários de um processo que a aplicação realizou ou irá executar. Eles aparecem temporariamente, na parte inferior da tela. They shouldn't interrupt the user experience, and they don't require user input to disappear.

Snackbars contêm uma única linha de texto diretamente relacionada à operação realizada. Eles podem conter uma ação de texto, mas não ícones. Você pode usá-los para exibir notificações.

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

Apenas um snackbar pode ser exibido por vez.

## Snackbars simples

Um snackbar básico que tem como objetivo reproduzir o comportamento do Google Keep's snackbar.

```
{{"demo": "SimpleSnackbar.js"}}
```

## Snackbars customizados

Aqui estão alguns exemplos de customização do componente. Você pode aprender mais sobre isso na [página de documentação de sobrescritas](#).

```
{{"demo": "CustomizedSnackbars.js"}}
```

## Snackbars posicionados

Em layouts amplos, os snackbars podem ser alinhados para a esquerda ou alinhados ao centro se forem colocados consistentemente no mesmo lugar na parte inferior da tela, no entanto, pode haver circunstâncias em que a posição do snackbar tenha de ser mais flexível. Você pode controlar a posição do snackbar especificando a propriedade

```
anchorOrigin .
```

```
{{"demo": "PositionedSnackbar.js"}}
```

## Comprimento da mensagem

Alguns snackbars com tamanho variável de mensagem.

```
{{"demo": "LongTextSnackbar.js"}}
```

## Transições

### Snackbars Consecutivos

Quando várias atualizações de snackbar são necessárias, eles devem aparecer um por vez.

```
{{"demo": "ConsecutiveSnackbars.js"}}
```

### Snackbars e botões de ação flutuante (BAFs)

Snackbars devem aparecer acima de BAFs (no mobile).

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

## Modificando a transição

[Grow](#) é a transição padrão, mas você pode usar uma diferente.

```
{{"demo": "TransitionsSnackbar.js"}}
```

## Controlando a direção do Slide

Você pode alterar a direção da transição do [Slide](#).

Example of making the slide transition to the left:

```
import Slide from '@material-ui/core/Slide';

function TransitionLeft(props) {
  return <Slide {...props} direction="left" />;
}

export default function MyComponent() {
  return <Snackbar TransitionComponent={TransitionLeft} />;
}
```





Para situações de uso mais avançadas, você pode tirar proveito com:

```
{{"demo": "DirectionSnackbar.js"}}
```

## Projetos Complementares

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

### notistack

 Stars  3k  downloads  1.9M/month

Este exemplo demonstra como usar com [notistack](#). notistack tem uma **API imperativa** que facilita a exibição de snackbars, sem ter que lidar com seu estado de aberto/fechado. It also enables you to **stack** them on top of one another (although this is discouraged by the Material Design guidelines).

```
{{"demo": "IntegrationNotistack.js", "defaultCodeOpen": false}}
```

## Acessibilidade

(WAI-ARIA: <https://www.w3.org/TR/wai-aria-1.1/#alert>)

By default, the snackbar won't auto-hide. However, if you decide to use the `autoHideDuration` prop, it's recommended to give the user [sufficient time](#) to respond.

When open, **every** `Snackbar` will be dismissed if `Escape` is pressed. Unless you don't handle `onClose` with the `"escapeKeyDown"` reason. If you want to limit this behavior to only dismiss the oldest currently open Snackbar

call `event.preventDefault` in `onClose` .

```
export default function MyComponent() {
  const [open, setOpen] = React.useState(true);

  return (
    <React.Fragment>
      <Snackbar
        open={open}
        onClose={(event, reason) => {
          // `reason` === `escapeKeyDown` if `Escape` was pressed
          setOpen(false);
          // call `event.preventDefault` to only close one Snackbar at a time.
        }}
      />
      <Snackbar open={open} onClose={() => setOpen(false)} />
    </React.Fragment>
  );
}
```