

Imagina o whatsapp, onde temos uma mensagem que enviamos para um usuário e uma ação

Ação = remover mensagem, desfazer mensagem, marca como lida...

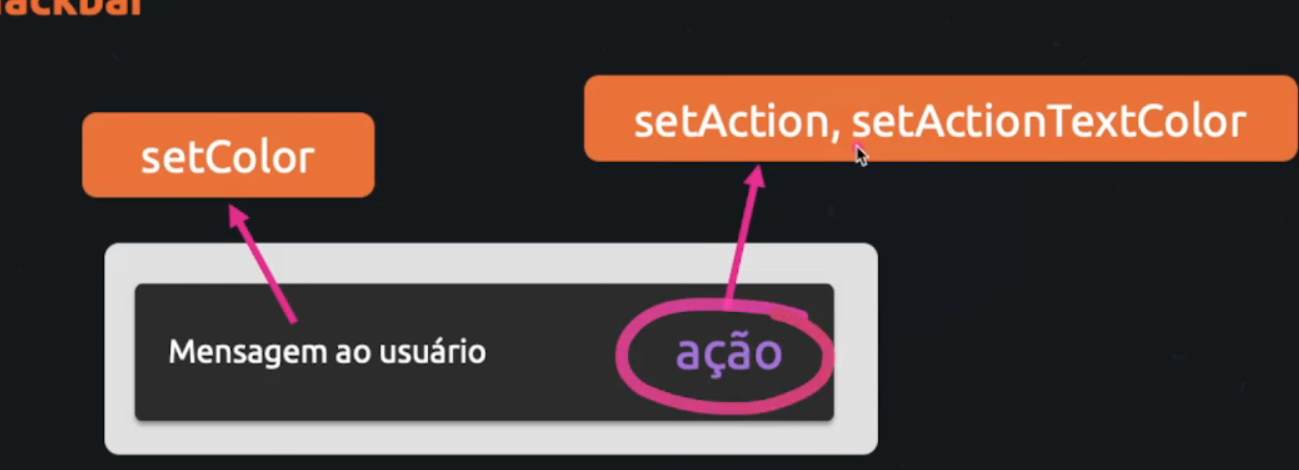
E por ai vai...

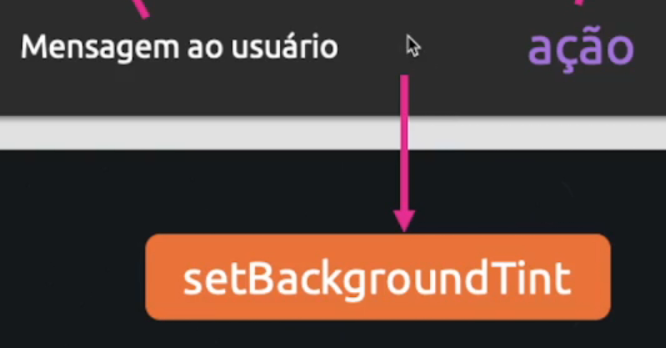
# Aqui ela esta funcionando, mas sem a ação, apenas exibindo a mensagem

*with*(binding) **{** btnExecutar.setOnClickListener **{** */\*\* Na SnackBar, precisamos passar 3 parametros  
 \* 1 view  
 \* 1 texto  
 \* 1 duração  
 \* igual ao toast  
 \*  
 \* Lembrar que a snackBar, voce pode passar qualquer view, ela mesma vai tentar encontrar o objeto do tipo viewGroup  
 \* Que pode sim exibir a snackBar  
 \* Diferente do toast, quando voce clica ela abre por cima de outros itens  
 \* Ja a snack bar ela precisa de um viewGroup para ela abrir.  
 \*  
 \* Usando dentro do setOnClickListener que já é do tipo View, podemos utilizar ela mesmo  
 \*/* Snackbar.make(  
 **it**,  
 "Alteração feita com sucesso",  
 Snackbar.*LENGTH\_INDEFINITE* // é um tipo de duração idefinida o item é exibido e só sai da tela se o usuario clicar em outra coisa.  
 ).show()  
  
 */\*\*Agora podemos configurar uma ação para nossa snackBar  
 \*   
 \*/* **}**

## Agora vamos configurar com uma ação

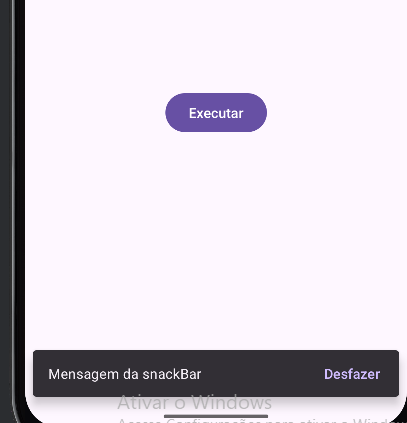
Vantagens de utilizar uma variável/objeto para colocar o snack dentro.





Aqui mudaríamos essa cor do fundo.

Observa a ação que fizemos



*with*(binding) **{** btnExecutar.setOnClickListener **{** viewSetOnClick **->** exibirSnackBar(viewSetOnClick) // Aqui estamos passando nossa View como parametro  
  
  
// /\*\* Na SnackBar, precisamos passar 3 parametros  
// \* 1 view  
// \* 1 texto  
// \* 1 duração  
// \* igual ao toast  
// \*  
// \* Lembrar que a snackBar, voce pode passar qualquer view, ela mesma vai tentar encontrar o objeto do tipo viewGroup  
// \* Que pode sim exibir a snackBar  
// \* Diferente do toast, quando voce clica ela abre por cima de outros itens  
// \* Ja a snack bar ela precisa de um viewGroup para ela abrir.  
// \*  
// \* Usando dentro do setOnClickListener que já é do tipo View, podemos utilizar ela mesmo  
// \*/  
// Snackbar.make(  
// viewSetOnClick,  
// "Alteração feita com sucesso",  
// Snackbar.LENGTH\_INDEFINITE // é um tipo de duração idefinida o item é exibido e só sai da tela se o usuario clicar em outra coisa.  
// ).show()  
//  
// /\*\*Agora podemos configurar uma ação para nossa snackBar  
// \*  
// \*/  
  
 **}  
 }** }  
  
 private fun exibirSnackBar(view: View) {  
  
  
 val snack = Snackbar.make(  
 view,  
 "Mensagem da snackBar",  
 Snackbar.*LENGTH\_INDEFINITE* )  
  
  
 // Aqui podemos passar a açao e ainda criar uma funçao lambda por fora  
  
 snack.setAction("Desfazer")**{** Toast.makeText(this, "Desfeito", Toast.*LENGTH\_SHORT*).show()  
 **}** // Configurando as cores e os demais para testes.  
  
 snack.setTextColor(  
 // Porque usamos o this? porque o contexto realmente é essa classe  
 // Se fosse outra classe teriamos que passar ela.  
 ContextCompat.getColor(this, R.color.*black*)  
 // E ai dessa forma podemos mudar a cor de varios itens..  
 )  
   
  
 // Aqui estamos fazendo um encadeamento de metodo  
 // O Show, poderiamos utilizar ali em cima depois do (  
 // Mas estamos passando tudo para dentro de uma variavel e exibindo com ela.  
 // Vantagens de usar a snack passando ela para uma variavel e transformando em objeto  
 // Agora podemos utilizar varios metodos com a variavel.  
 snack.show()  
 }  
  
}

Configurando outras coisas, além da cor

Mudando a cor do botão da ação



Mudando a cor do fundo da snackBar

