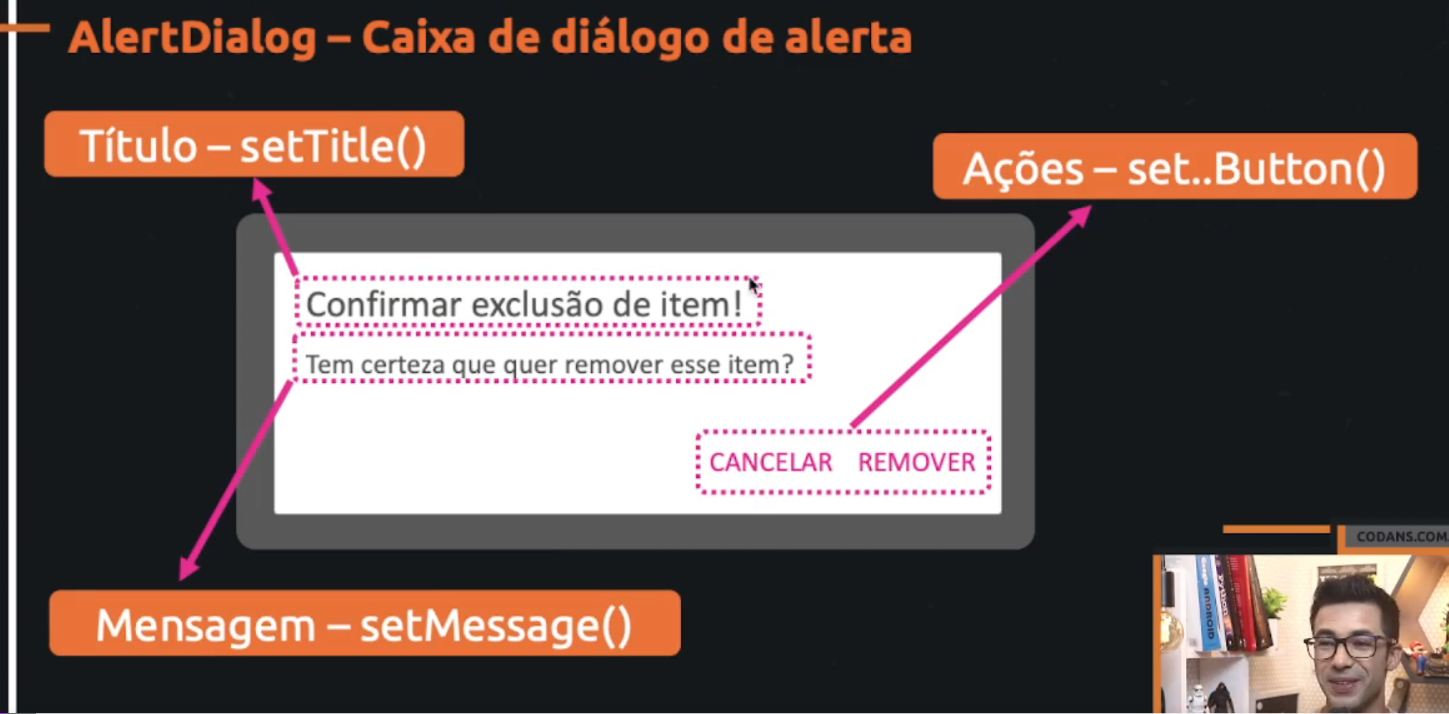
Caixa de dialogo de alerta

AlertDialog

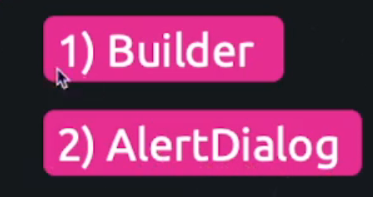


Para os botões temos os

SetNegativeButton = para cancelar

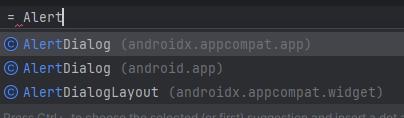
setPositiveButton = para confirmar

Para construir a caixa vamos precisar de 2 classes



Builder vamos construir tudo que vamos precisar na nossa caixa

Alert vai ser para exibir os itens ou ate mesmo esconder



Repara que temos 2 classes, mas sempre vamos da preferencia em utilizar o AndroidX, porque é a nova maneira de organizar os recursos do android.

# Todo o código

with(binding) **{** btnCadastrar.setOnClickListener **{** alertDialogResumido()  
 **}  
 }** }  
  
 private fun caixaDialogAlerta() {  
  
 */\*\*  
 \* Estamos acessando a classe AlertDialog  
 \* Depois vamos usar o metodo Builder()  
 \* Ele serve para construir o alertDialog para gente.  
 \*/* val alertBuilder = AlertDialog.Builder(this) // Aqui é para construir  
  
 */\*\*  
 \* Agora que criamos nosso alertDialog, podemos fazer todas as configurações  
 \*/* alertBuilder.setTitle("Cadastro de posto")  
 alertBuilder.setMessage("Deseja realmente cadastrar esse posto?")  
  
 */\*\* Agora vamos utilizar as ações do nosso AlertDialog  
 \*  
 \*/* alertBuilder.setNegativeButton("Cancelar")**{** dialog, posicao **->** */\*\* Aqui dentro vamos utilizar 2 parametros, o dialog e posição  
 \* Posição seria a posição do botao que estamos criando.  
 \* Cada posição vai ter ex um numero e ai vamos poder saber qual item foi clicado  
 \* Imagina que a posiçao 1 seria adicionarPosto, e com isso iriamos chamar um metodo para adicionar novos postos  
 \* Imagina que a posição 2 seria cancelar, então usuariamos igual a baixo  
 \*  
 \* Lembrar que toda vez que queremos executar uma função lambda como 2 parametro podemos fazer isso  
 \* Resumindo o metodo setNegativeButton ele espera receber 2 parametros, e com isso podemos fazer o segundo parametro de função lambda  
 \*/* Toast.makeText(this, "Cancelar clicado $posicao.", Toast.LENGTH\_SHORT).show()  
  
 */\*\* Tudo que voce quiser fazer com o alert, podemos utilizar o dialog  
 \* //dialog.dismiss() //dimiss = liberar a caixa de dialago  
 \*/* **}** alertBuilder.setPositiveButton("Adiconar")**{**dialog, posicao **->** */\*\*  
 \* Aqui vamos seguir o mesmo parametro  
 \*/* Toast.makeText(this, "Posto adiconado $posicao", Toast.LENGTH\_SHORT).show()  
  
 **}** */\*\* Outras ações  
 \* Aqui temos 2 ações, se em Cancelable estiver FALSO  
 \* O usuario tem que fazer alguma ação, ele nao tem a opção de cancelar.  
 \* \*\* CANCELAR \*\* lembra que quando clicamos fora ele automaticamente fecha o Alert? então isso nao é mais possivel se cancelar estiver false  
 \*  
 \*  
 \* O NeutralButton serve para voce colocar um botao onde o usuario pode pedir ajuda.  
 \*  
 \* setIcon podemos utilizar para exibir um icone que fizemos.  
 \*  
 \*/* alertBuilder.setCancelable(false)  
  
  
 alertBuilder.setNeutralButton("Ajuda")**{**dialog, posicao **->** Toast.makeText(this, "Clique em adicionar para adicionar um novo posto\nCancelar para não adicionar um posto.", Toast.LENGTH\_SHORT).show()  
 **}** alertBuilder.setIcon(R.drawable.ic\_alert\_24)  
  
  
  
 */\*\*  
 \* Agora vamos criar nosso alertDialog  
 \*/* val alertDialog = alertBuilder.create() // Aqui é para criar  
  
 */\*\*  
 \* Agora vamos exibir ele  
 \*/* alertDialog.show()  
  
  
  
  
 }  
  
 fun alertDialogResumido() {  
 */\*\*  
 \* Aqui estamos fazendo de forma resumida com encadeamento de metodos.  
 \*/* AlertDialog.Builder(this)  
 .setTitle("O titulo")  
 .setMessage("Mensagem do meu alerta")  
 .setCancelable(false) // O usuario é obrigado a clicar em alguma informação  
 .setNeutralButton("?")**{**dialog, posicao **->** Toast.makeText(this, "Voce quer adicionar um novo posto?", Toast.LENGTH\_SHORT).show()  
 **}** .setNegativeButton("Cancelar")**{**dialog, posicao **->** Toast.makeText(this, "Cancelando... $posicao", Toast.LENGTH\_SHORT).show()  
 **}** .setPositiveButton("Adicionar")**{**dialog, posicao **->** Toast.makeText(this, "Adicionando novo posto... $posicao", Toast.LENGTH\_SHORT).show()  
 **}** .setIcon(R.drawable.ic\_alert\_24)  
 .create()  
 .show()  
 }  
}