Explicação sobre os métodos.

 registerForActivityResult:

* Esse método é usado para registrar uma atividade que retornará um resultado. Por exemplo, capturar uma imagem, gravar um vídeo ou selecionar um arquivo. Ele foi introduzido como uma alternativa moderna ao método startActivityForResult, que agora está obsoleto.
* Ele aceita dois parâmetros:
  + Um contrato, como o ActivityResultContract, que define o tipo de ação e o resultado esperado.
  + Um callback, que lida com o resultado retornado.

 ActivityResultContracts.CaptureVideo:

* Esse é um contrato embutido no Android que permite capturar vídeos diretamente da câmera do dispositivo.
* Ele solicita que o aplicativo lance uma intenção (intent) para gravar um vídeo e retorna a URI do vídeo capturado.

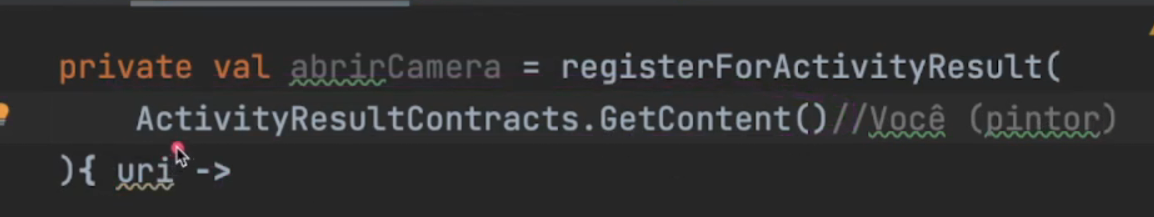
Agora vamos começar com a variável

abrirCamera

registrar uma activity e retornar seu resultado

Em baixo vamos definir o contrato

E no GetContent seria o tipo de informação que iriamos pegar.



Contrato define quem vai ser o objeto que vai manipular o resultado e oque vai ser retornado, nesse exemplo de cima seria um quadro...

Mao na massa

# Criando a variável

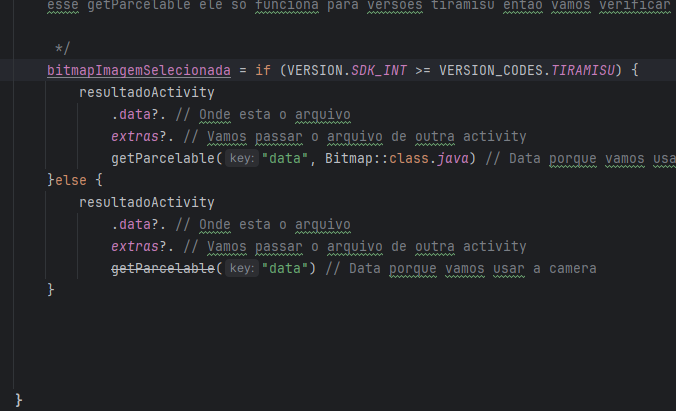
private val abrirCamera = registerForActivityResult( // Registrar um resultado de uma activity  
 ActivityResultContracts.StartActivityForResult()  
) **{** resultadoActivity **->** /\*// resultCode = codigo de resultado.  
 // Aqui só vai entrar no IF se o resultado der certo.  
 if (resultadoActivity.resultCode == RESULT\_OK) {  
  
 }else {  
  
 }  
 Não vamos fazer dessa forma, vamos fazer de uma forma mais curta.  
 \*/  
  
 /\* extras e quando pegamos dados de uma outra activity  
 Como aqui estamos abrindo uma activity(camera)  
 E pegando seu resultado  
 aqui estamos usando o extras para poder captpurar seus dados.  
  
 data = recuperar imagem da camera  
  
 Bitmap = podemos trocar o formato da imagem ex: PNG para JPG  
  
 esse getParcelable ele so funciona para versoes tiramisu então vamos verificar isso  
  
 \*/  
 if (VERSION.*SDK\_INT* >= VERSION\_CODES.*TIRAMISU*) {  
 resultadoActivity  
 .data?. // Onde esta o arquivo  
 *extras*?. // Vamos passar o arquivo de outra activity  
 getParcelable("data", Bitmap::class.*java*) // Data porque vamos usar a camera // Bitmap para converter o tipo da imagem.  
 }else {  
 resultadoActivity  
 .data?. // Onde esta o arquivo  
 *extras*?. // Vamos passar o arquivo de outra activity  
 getParcelable("data") // Data porque vamos usar a camera   
 }

Agora vamos precisar do nosso atributo

private var bitMapImagemSelecionada: Bitmap? = null

Agora quando o usuário selecionar a câmera vamos usar o atributo

E isso depende da versão se for a 33 vai usar o primeiro IF se for mais velha vamos usar la em baixo.

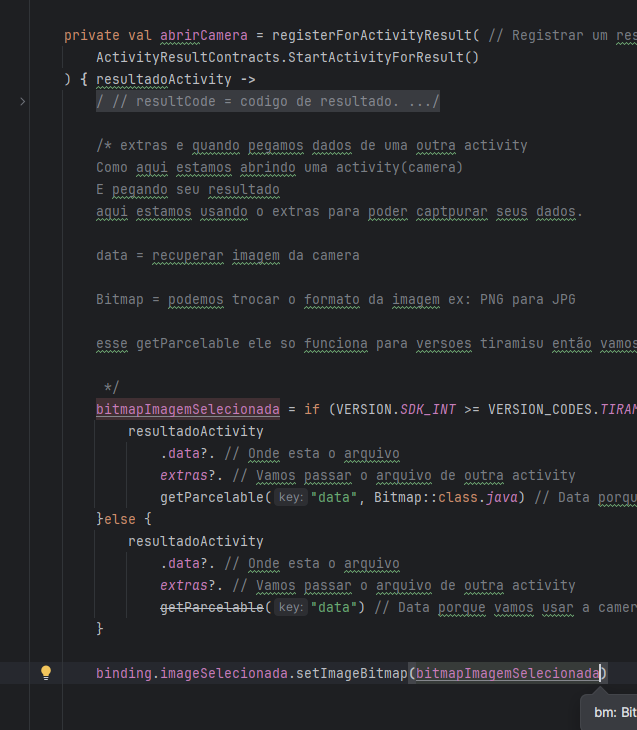


Capturando imagem no onClick

binding.btnCamera.setOnClickListener **{** /\* Aqui como usamos o StartActivityForResult precisamos passar uma intent  
 \* Ja na galeria informamos o tipo de dados que queremos pegar.  
 \*  
 \* MediaStore = vai acessar varios recursos que podemos usar, nesse caso aqui vamos acessar uma ação e essa ação vai ser capturar imagem...\*/  
 val intent = Intent(MediaStore.*ACTION\_IMAGE\_CAPTURE*)  
 abrirCamera.launch(intent)  
   
**}**

Agora vamos exibir a imagem no quadro.

Dentro do atributo abrirCamera



Agora que estamos usando a câmera, precisamos pedir a permissão também.

<uses-feature  
 android:name="android.hardware.camera"  
 android:required="false" />  
  
<uses-permission android:name="android.permission.READ\_MEDIA\_IMAGES" />  
<uses-permission android:name="android.permission.CAMERA" android:**requiredFeature**="true"/>

Aqui estamos dando a permissão e dizendo que a câmera é necessária para acessar o aplicativo.

Assim qualquer celular que não tenha câmera, o aplicativo não pode ser instalado.

