

Imagina que o usuário da a permissão somente enquanto o aplicativo esta sendo utilizado...

Ele também pode solicitar a permissão de “Apenas uma única vez”

Então vamos primeiro verificar as permissões negadas, antes de solicitar.

# Verificar as permissões negadas!

class Permissao {  
  
 companion object {  
  
  
  
  
  
  
  
  
 fun requisitarPermissoes(activity : Activity, permissoes: List<String>){  
 /\*Verificar as permissoes negadas  
 \* Esse Check ele vai fazer uma verificação individual de cada permissao  
 \* check = checar  
 \*  
 \* Esse metodo ele checa cada permissao de modo individual  
 \* E o parametro esta passando uma lista, então vamos percorrer essa lista e verificar cada permissao de forma individual  
 \* Se a permissao for concedida ele retorna ( 0 ) PER... GRANTED  
 \* Se for negada ele retorna ( -1 ) PER... DENIED  
 \*  
 \* temPermisao é onde vai ficar o valor 0 ou -1  
 \*  
 \* Agora vamos verificar se a permissao é aceita == 0 / PackageManager.PERMISSION\_GRANTED /  
 \*  
 \* Vamos também criar o permissoesNegadas para ir adicionando as permissoes que o usuario nao quis dar.  
 \*  
 \* Lembrar que o temPermissoes é um boleano, então ele vai retornar sempre um verdadeiro ou falso  
 \* \*/  
  
 val permissoesNegadas = *mutableListOf*<String>()  
  
 permissoes.*forEach* **{** permissao **->** val temPermisao = ContextCompat.checkSelfPermission(  
 activity,  
 permissao  
 ) == PackageManager.*PERMISSION\_GRANTED* // Verificar se foi aceita a permissao  
  
 if ( !temPermisao ) {  
 /\* Aqui estou usando negação  
 Então sempre quando a permissao for false ele vai cair aqui dentro.  
 E vai guardando as permissoes que foram negadas "false"  
 \*/  
  
 permissoesNegadas.add(permissao)  
  
 }  
  
 /\* Agora so vamos usar aqui, se tiver permissoes negadas  
 \* Se tiver permissoes negadas ai sim, pedimos novamente  
 \* Resumindo se o usuario ja deu a permissao, nao vamos pedir novamente  
 \* Mas se essa permissao estiver NEGADA vamos pedir novamente.\*/  
 if (permissoesNegadas.*isNotEmpty*()) {  
 ActivityCompat.requestPermissions(  
 activity,  
 permissoesNegadas.*toTypedArray*(),  
 0  
 )  
 }  
 **}**

Toda vez que o usuário abrir o aplicativo ele vai ser obrigado a aceitar a permissão que ele negou novamente.

Agora imagina que, o usuário negou a permissão, igual quando eu faço com a localização da Uber, ai teríamos que travar a busca por viagens também...

Para isso vamos ter outro método, esse método vai ser executado após o ser utilizado o método requestPermissions

Apos o requestPermissions solicitar todas as permissões, ele vai executar o método onRequestPermissionsResult..

Entao na nossa uploadImagemActivity, antes do onCreate vamos criar esse método

override fun onRequestPermissionsResult(  
 requestCode: Int,  
 permissions: Array<out String>,  
 grantResults: IntArray,  
 deviceId: Int  
) {  
 super.onRequestPermissionsResult(requestCode, permissions, grantResults, deviceId)  
}

Agora vamos passar o requestCode que faltou na outra aula e vamos entender melhor ele.

fun requisitarPermissoes(activity : Activity, permissoes: List<String>, requestCode : Int){

Vamos por ele dentro do método também

if (permissoesNegadas.*isNotEmpty*()) {  
 ActivityCompat.requestPermissions(  
 activity,  
 permissoesNegadas.*toTypedArray*(),  
 requestCode  
 )  
 }  
**}**

E na nossa activity vamos chamar ele.

Passamos ele com código 100



Resumindo aqui esse método vai sr chamado após as permissões serem negadas.

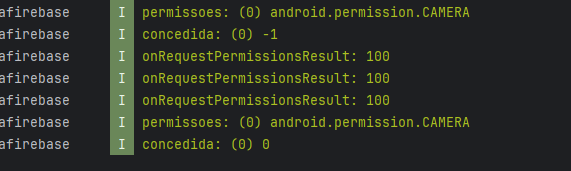
Poderiamos aqui dentro pegar o CODE e desabilitar um botão por exemplo

override fun onRequestPermissionsResult(  
 requestCode: Int,  
 permissions: Array<out String>,  
 grantResults: IntArray,  
 deviceId: Int  
) {  
 super.onRequestPermissionsResult(requestCode, permissions, grantResults, deviceId)  
 Log.i("permissao\_app", "onRequestPermissionsResult: $requestCode ")  
  
 // Também vamos exibir as permissoes  
 // Vamos usar o forEachIndexed ele vai exibir também os indices  
  
 permissions.*forEachIndexed* **{**indice, valor **->** Log.i("permissao\_app", "permissoes: ($indice) $valor ")  
 **}** // grantResults  
 grantResults.*forEachIndexed* **{**indice, valor **->** Log.i("permissao\_app", "concedida: ($indice) $valor ")  
 **}**}

Agora se você reparar temos essa lista de permissão

Aqui esta a posicação delas também

// Agora vamos passar as permissoes.  
// Para acessar a permissao que vai ser passada.  
private val permissoes = *listOf*(  
 Manifest.permission.*CAMERA*, // 0  
 Manifest.permission.*READ\_MEDIA\_IMAGES*, // 1  
 Manifest.permission.*ACCESS\_COARSE\_LOCATION // 2*)



Repara que na posicação 0 == câmera foi negada ficou -1

Agora se você quisesse desabilitar o botão seria só entrar dentro do método e desabilitar, dessa forma.

ESTUDAR MAIS SOBRE!!!!