

Actiobar – você pode configurar:

Titulos, botões de ações e : mais ações...

Toolbar

Podemos ao invés de fazer um titulo, podemos adicionar uma imagem.

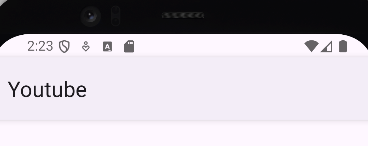
Diferenças: a toolbar, podemos customizar muito mais.

# PRATICA

Primeiro fomos em strings.xml e configuramos o nome do nosso app para Youtube, para simular que estamos criando a actionBar do youtube

Depois fomos em themes e tiramos a opção de não exibir o actionBar

Ficando dessa forma -> <style name="Base.Theme.ToolbarvsActionBar" parent="Theme.Material3.DayNight">



Mas fazendo dessa forma ele vai adicionar a actionBar em todos os itens e não é dessa forma que queremos.

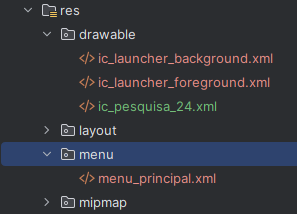
E para remover a acitionBar onde não queremos exibir, podemos ir no onCreate e colocar esse método

*supportActionBar*?.hide()

hide = esconder

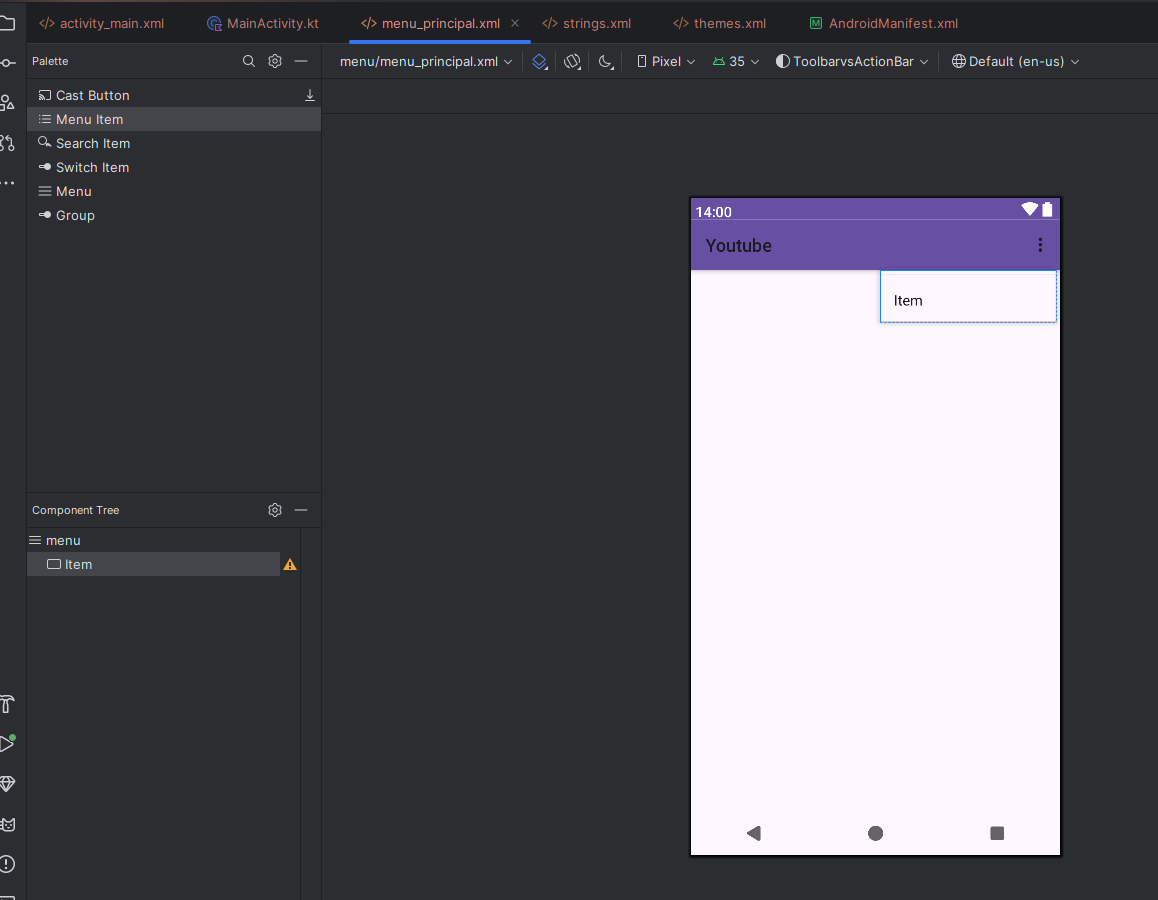
# Agora estamos criando nosso próprio menu.

Primeiro criamos nossa própria pasta



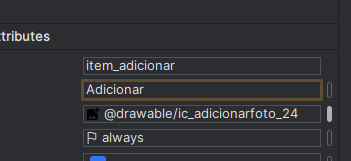
Após isso criamos nosso próprio diretório XML

Após criar nosso XML temos esses atributos e repara que já estamos com as opções do actionBar, onde temos o nome do app + a opção de configuração



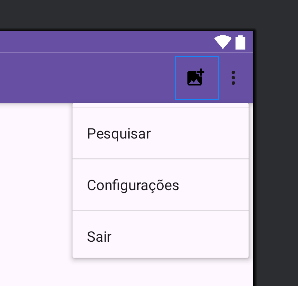


# Adicionando ícones a barra

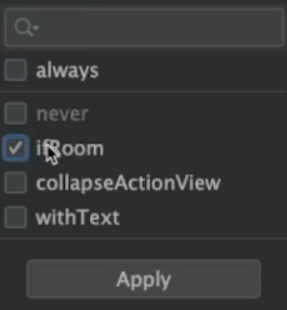


Aqui adicionamos um novo ícone, mas para ele aparecer na barra precisamos clicar em showAsAction e selecionar a opção **Always**

Ficando dessa forma



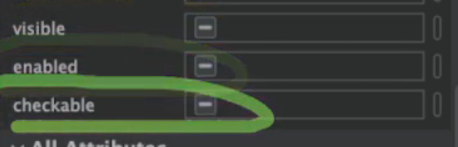
# Opções para aparecer na BARRA



Always = aparecer na barra do actionBar

Never = nunca aparecer

IfRoom = se tiver espaço ele aparece





Essas opções usamos geralmente para o Switch.

Aqui são as outras opções, se ele vai ficar visível ou não.

Se ele vai ficar habilitado ou não

Se ele vai ficar marcado ou não.

# Adicionando nosso ActionBar na tela

Criamos esse método em baixo do onCreate

override fun onCreateOptionsMenu(menu: Menu?): Boolean {  
 */\*\* AO CRIAR OPCOES DE MENU  
 \* Esse metodo permite que voce reconstrua o menu  
 \* Aqui vamos precisar inflar e construir nosso menu  
 \* Esse metodo é chamado quando a nossa tela é carregada e ai se tiver ele constroi e menu para a gente  
 \*  
 \*  
 \* Primeiro parametro é nosso XML que construimos  
 \* Segundo parametro é o proprio menu que vai construir.  
 \*  
 \* Objeto menu: -> é o menu que é utilizado pelo proprio android, seria aquelas opções que tem em cima.  
 \* Resumindo ele vai criar e passar para voce, e voce vai inflar utilizando seu proprio layout  
 \* Ai sim ele vai configurar e vai inflar no seu layout.  
 \*  
 \* Agora os menus não sao clicaveis, agora vamos utilizar outros metodos para isso.  
 \*/  
  
 menuInflater*.inflate(R.menu.*menu\_principal*, menu)  
  
 return true  
}

# Adicionando ações aos itens do actionBar

Método criado em baixo do onCreateOptionsMenu

override fun onOptionsItemSelected(item: MenuItem): Boolean {  
 */\*\* 2 AO UTILIZAR AS OPCOES SELECIONADAS  
 \* Atenção não vamos criar eventos de clique, porque ele já vai de forma automatica nesse metodo aqui  
 \* Quando o usuario clicar no item na tela, o metodo vai ser chamado e ele vai saber qual foi o menuItem que foi clicado  
 \* Ai sim ele vai te passar o menuItem que foi clicado  
 \*  
 \*  
 \* Podemos fazer um teste, pegamos o item e dentro dele vai ter o ID dos itens  
 \* Lembra que configuramos um ID para cada item do actionBar? la no menu.xml?  
 \* Então agora ele vai saber qual foi o id do item clicado  
 \*  
 \* Ou seja aqui podemos adicionar também uma opçao para adicionar tema dark...  
 \*/* when(item.*itemId*){  
 R.id.*item\_adicionar* -> {  
 Toast.makeText(this, "Adicionar foto", Toast.*LENGTH\_SHORT*).show()  
 }  
 R.id.*item\_pesquisar* -> {  
 Toast.makeText(this, "Pesquisar foto", Toast.*LENGTH\_SHORT*).show()  
 }  
 R.id.*item\_configuracoes* -> {  
 Toast.makeText(this, "Configurar foto", Toast.*LENGTH\_SHORT*).show()  
 }  
 R.id.*item\_sair* -> {  
 // Aqui vamos fazer a ação para fechar o app  
 Toast.makeText(this, "Fechando app", Toast.*LENGTH\_SHORT*).show()  
 finish()  
 }  
  
 }  
  
  
  
 return true  
}