

<https://github.com/PacktPublishing/Android-Programming-with-Kotlin-for-Beginners>

# Android com Kotlin

Aula 04 - Layouts

## Android Programming with Kotlin for Beginners

Build Android apps starting from zero programming experience  
with the new Kotlin programming language



John Horton

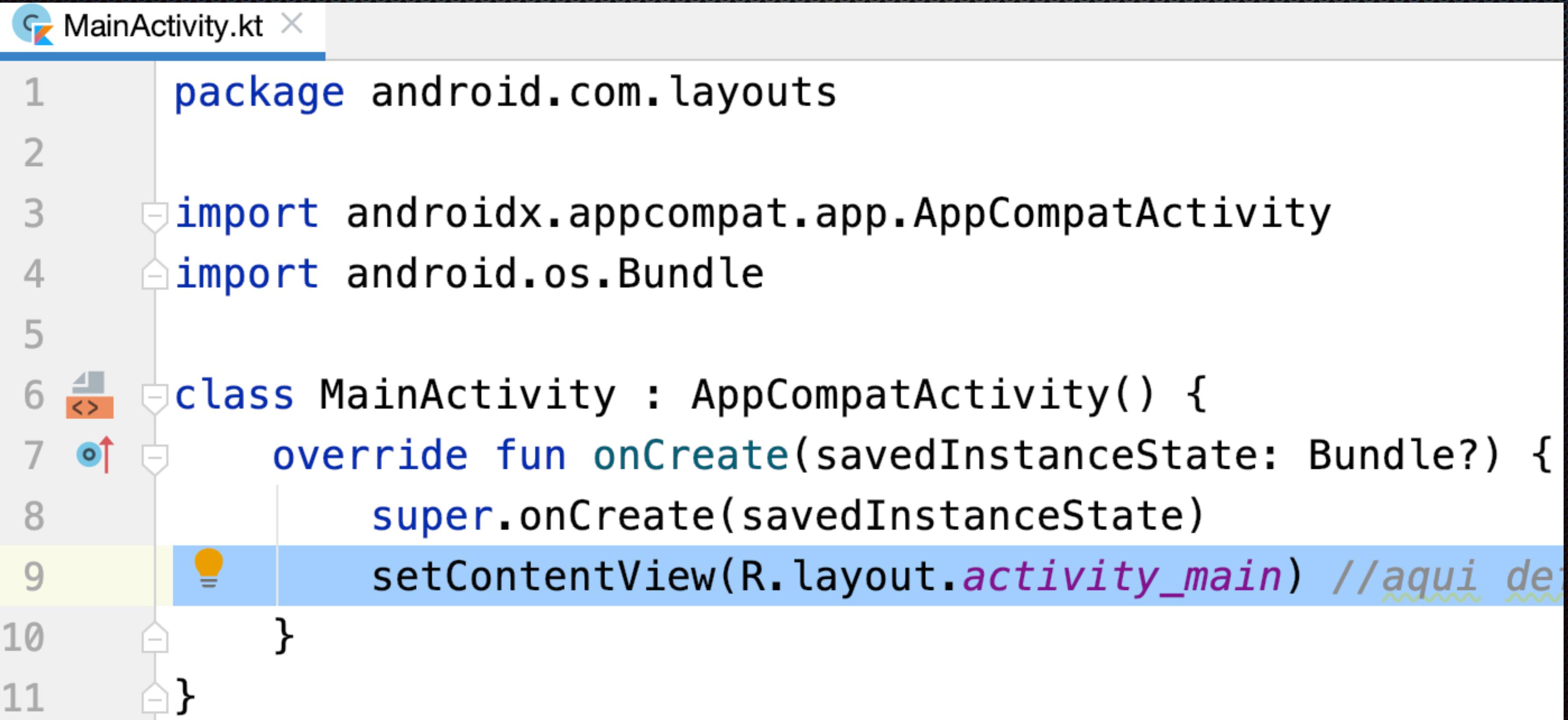
# Aula 04 - Constraint Layout e Table Layout

- Layout dentro de outro layout (LinearLayout)
- Interação entre Views (TableLayout e ConstraintLayout)
- CardView
- ScrollView

# Layout dentro de outro Layout

# Layouts

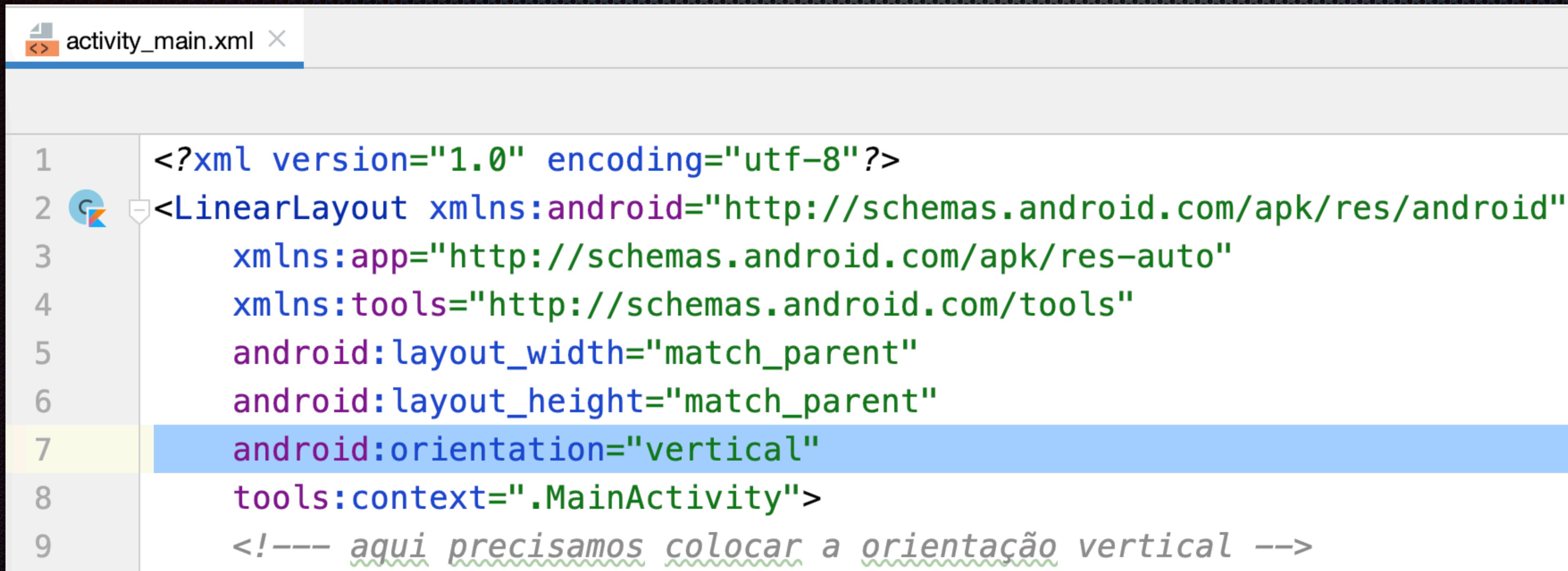
- São formas de agrupar componentes ou até mesmo outros layouts.
- Usamos o método setContentView() para definir qual estamos usando.



```
>MainActivity.kt
```

```
1 package android.com.layouts
2
3 import androidx.appcompat.app.AppCompatActivity
4 import android.os.Bundle
5
6 class MainActivity : AppCompatActivity() {
7     override fun onCreate(savedInstanceState: Bundle?) {
8         super.onCreate(savedInstanceState)
9         setContentView(R.layout.activity_main) // aqui de
10    }
11 }
```

# Linear Layout em Modo Vertical



```
activity_main.xml <--> 1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     android:orientation="vertical"
8     tools:context=".MainActivity">
9     <!-- aqui precisamos colocar a orientação vertical -->
```

# Adicionamos aqui um TextView

The screenshot shows the Android Studio interface with the XML code for `activity_main.xml` on the left and a preview of the layout on the right.

**XML Code:**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <!-- aqui precisamos colocar a orientação vertical -->

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="50sp"
        android:gravity="center_horizontal"
        android:text="Menu" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

**Preview Pane:**

The preview shows a single `TextView` with the text "Menu" centered horizontally and vertically within its parent `LinearLayout`.

# Adicionamos um Multi-line TextView

The screenshot shows the Android Studio interface with the XML code for `activity_main.xml` on the left and a layout preview on the right.

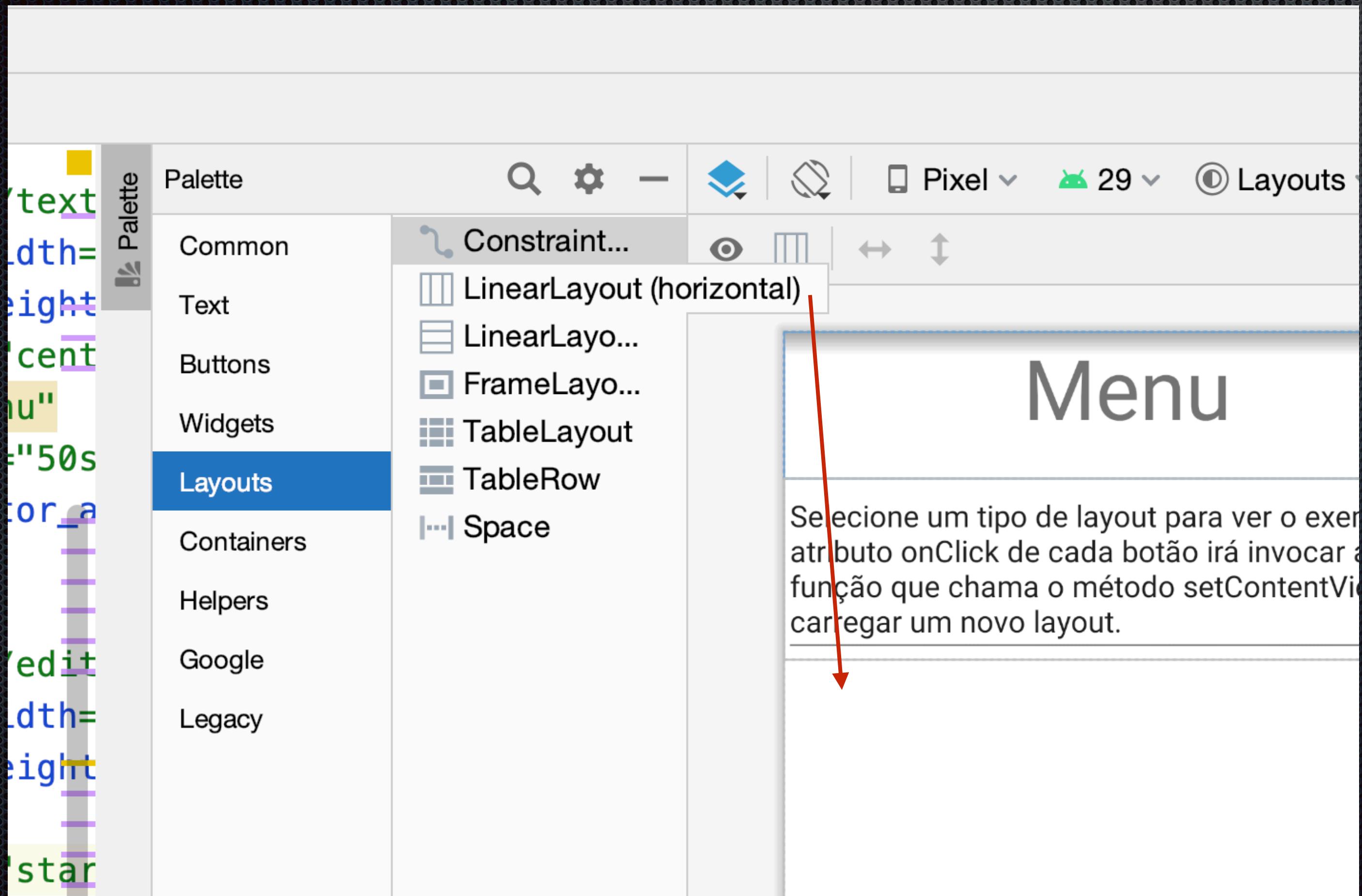
**XML Code:**

```
11 <TextView  
12     android:id="@+id/textView"  
13     android:layout_width="match_parent"  
14     android:layout_height="wrap_content"  
15     android:textSize="50sp"  
16     android:gravity="center_horizontal"  
17     android:text="Menu" />  
18  
19 <EditText  
20     android:id="@+id/editTextTextMultiLine"  
21     android:layout_width="match_parent"  
22     android:layout_height="wrap_content"  
23     android:ems="10" ← define quantos caracteres por linha  
24     android:gravity="start|top"  
25     android:inputType="textMultiLine"  
26     android:text="Selecionar um tipo de layout para ver o exemplo. O atributo onClick de cada botão irá chamar a função que chama o método setContentView para carregar um novo layout."  
27     tools:layout_editor_absoluteX="20dp"  
28     tools:layout_editor_absoluteY="133dp" />  
29  
30 </androidx.constraintlayout.widget.ConstraintLayout>  
31
```

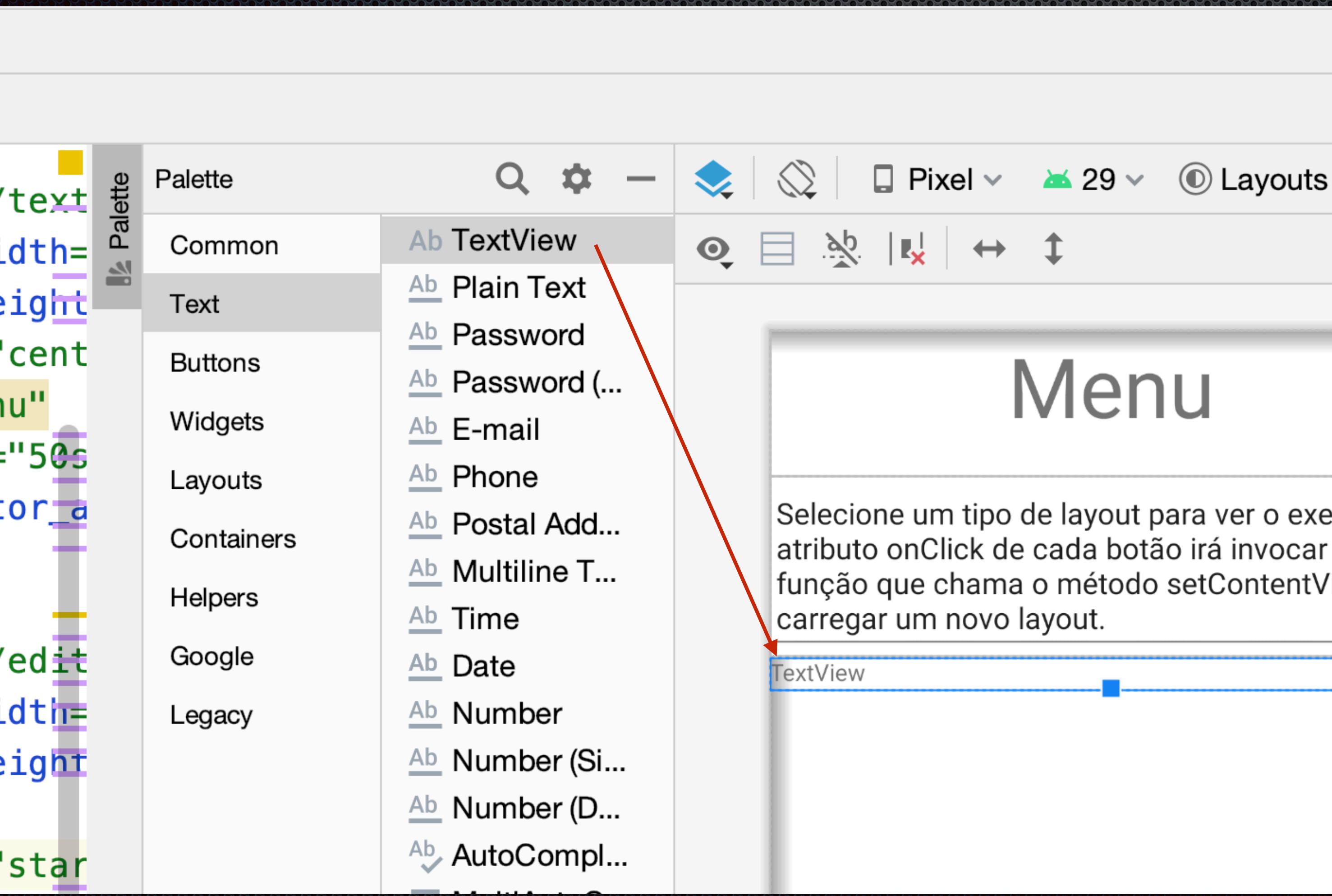
**Layout Preview:**

The layout preview shows a single `EditText` field with the placeholder text: "Selecionar um tipo de layout para ver o exemplo. O atributo onClick de cada botão irá chamar a função que chama o método setContentView para carregar um novo layout." The text is displayed in multiple lines, demonstrating the `textMultiLine` attribute.

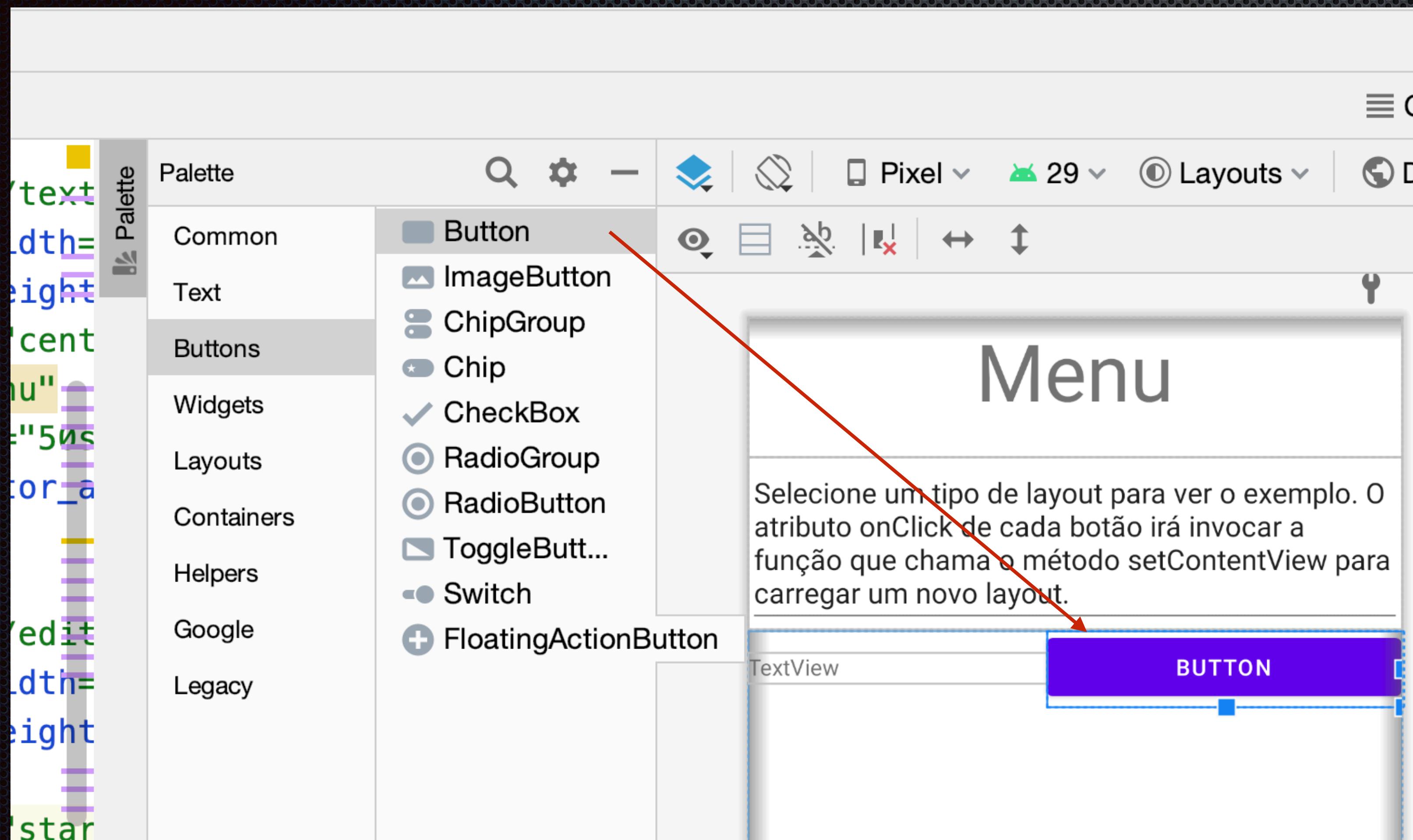
# Vamos adicionar um LinearLayout abaixo do Multi-line TextView



# E adicionamos um TextView neste novo LinearLayout



# É um Button ao lado do TextView



Alteramos aqui a propriedade da altura para terminar o layout ao final do tamanho dos componentes.

The screenshot shows the Android Studio interface with the XML code for `activity_main.xml` and the Java code for `MainActivity.kt`. The XML code defines a `ScrollView` containing a `LinearLayout` with a `TextView` and a `Button`. A red arrow points from the `android:layout_height="wrap_content"` line in the `LinearLayout` to the `wrap_content` entry in the palette's list of available height values.

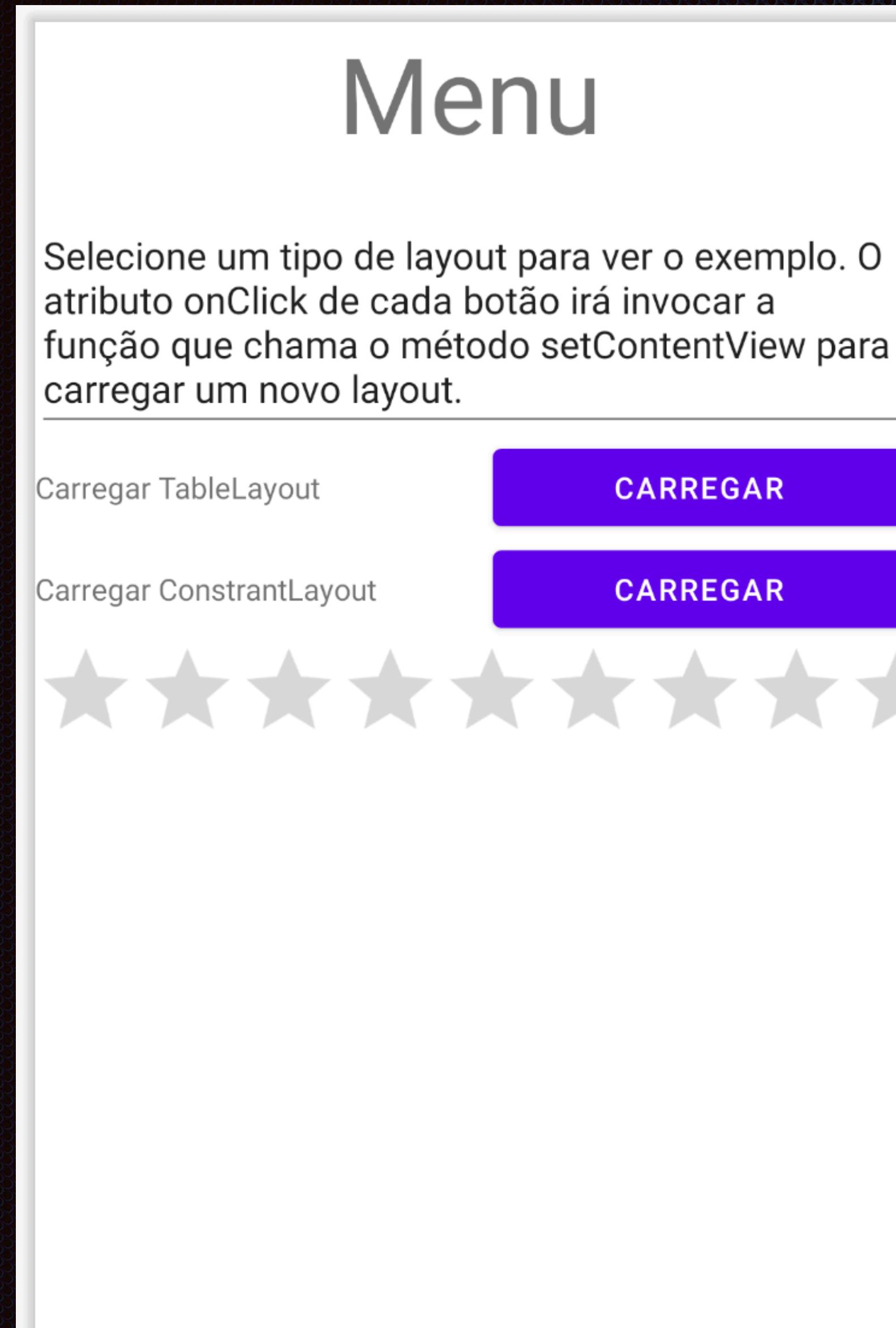
```
23 <ScrollView android:layout_width="match_parent" android:layout_height="wrap_content">
24     android:ems="10"
25     android:gravity="start|top"
26     android:inputType="textMultiLine"
27     android:text="Selecione um tipo de layout"
28     app:layout_constraintTop_toBottomOf="@+id/textView1"
29     tools:layout_editor_absoluteX="0dp" />
30
31 <LinearLayout
32     android:layout_width="match_parent"
33     android:layout_height="wrap_content" // Red arrow points here
34     android:orientation="horizontal">
```

Paleta:

- Pixel: 29
- Layouts
- Text View
- Button

Atributo `onClick` de cada botão irá invocar a função que chama o método `setContentView` para carregar um novo layout.

Vamos criar um novo LinearLayout abaixo do anterior e um RatingBar

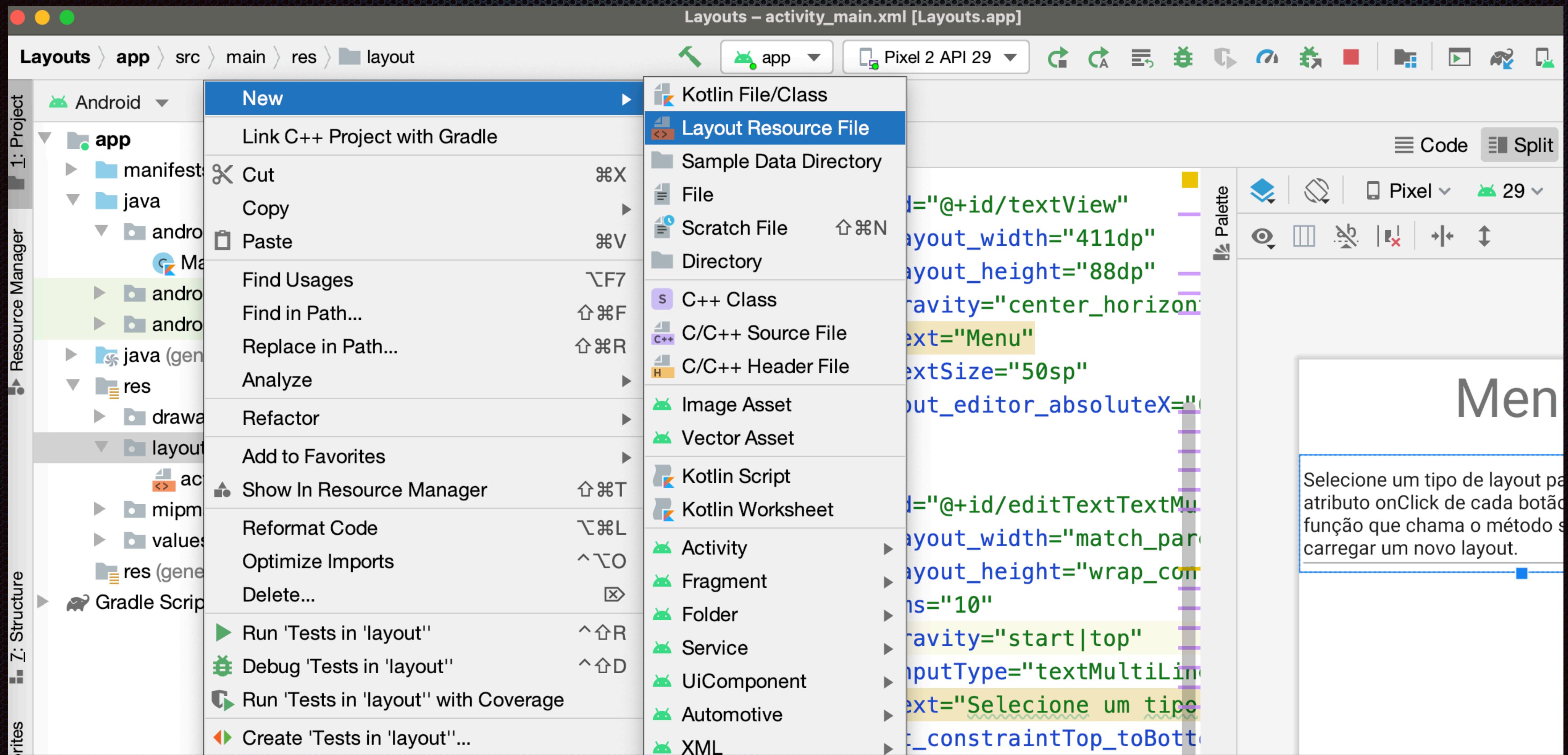


Alterando algumas propriedades...

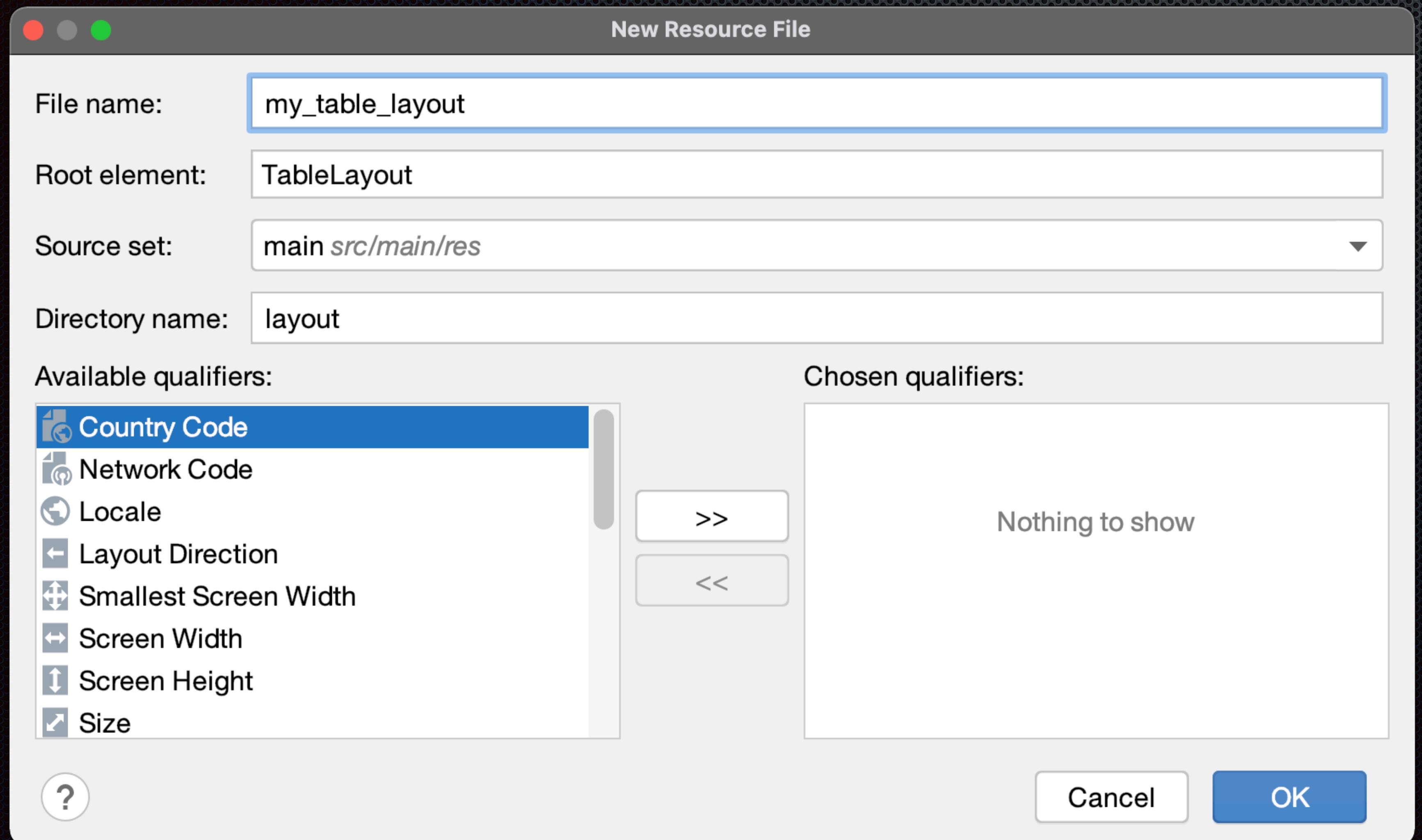


# Interação entre Views

# Vamos criar um novo arquivo de layout



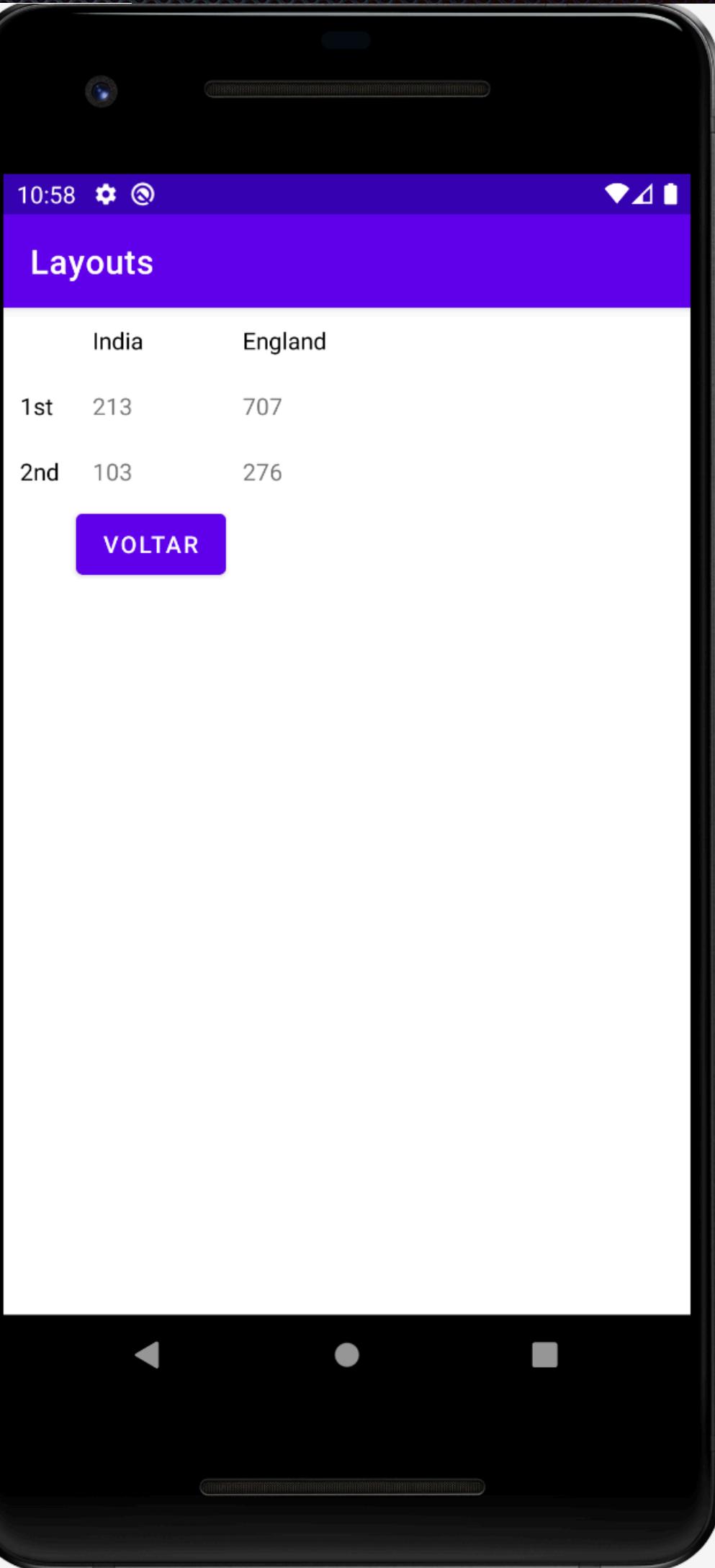
# Criaremos um TableLayout



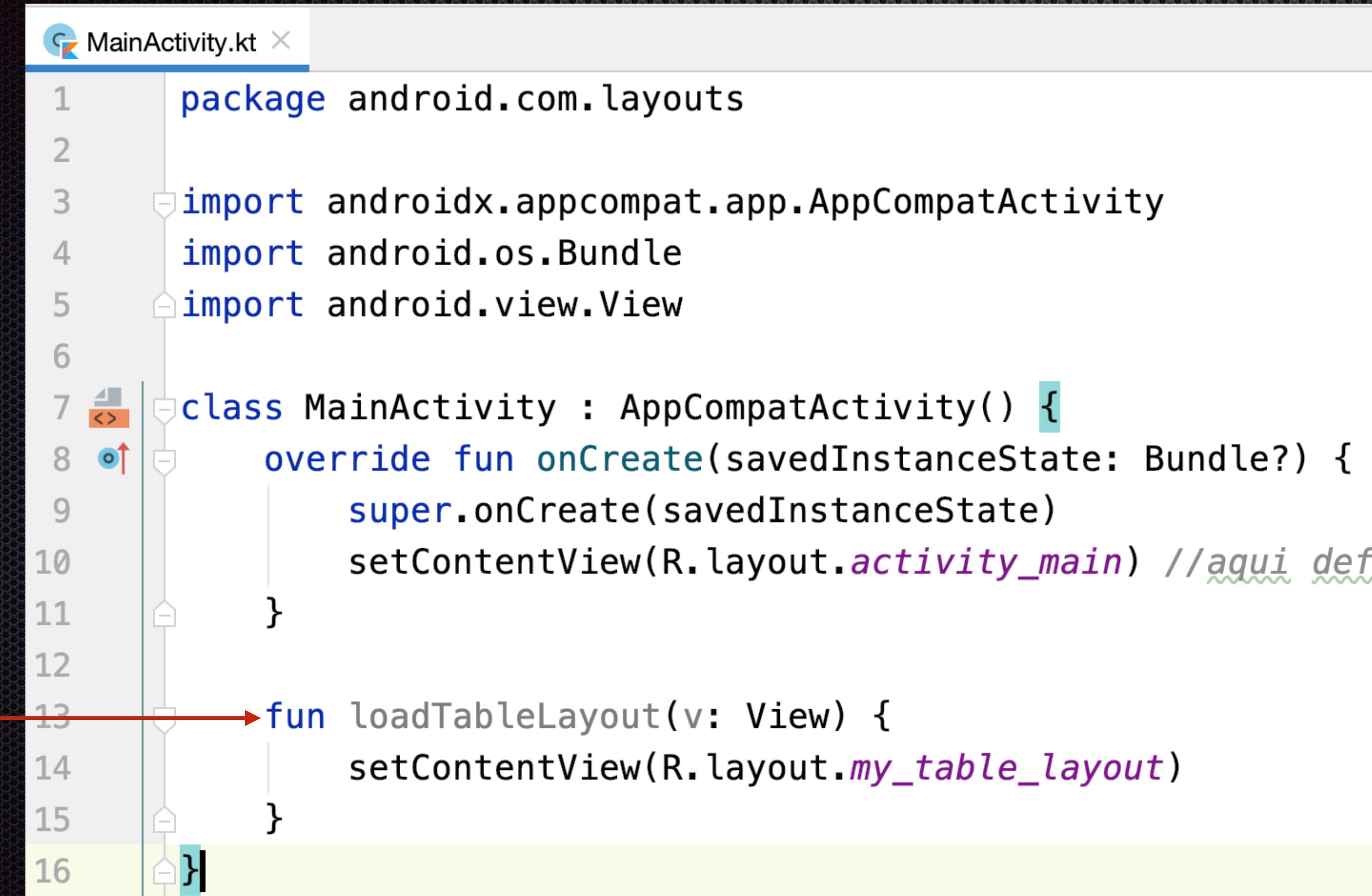
# Criaremos no TableLayout uma tabela de dados

```
2 <TableLayout xmlns:android="http://schemas.android.com/
3     android:layout_width="wrap_content"
4     android:layout_height="wrap_content">
5
6     <TableRow
7         android:layout_width="wrap_content"
8         android:layout_height="wrap_content">
9
10        <TextView
11            android:id="@+id/textView2"
12            android:layout_width="wrap_content"
13            android:layout_height="wrap_content"
14            android:layout_column="1"
15            android:padding="10sp"
16            android:text="India"
17            android:textColor="@android:color/black" />
18
19        <TextView
20            android:id="@+id/textView1"
21            android:layout_width="wrap_content"
22            android:layout_height="wrap_content"
23            android:layout_column="2"
24            android:padding="10sp"
25            android:text="England" |
```

83 android:layout\_height="wrap\_content"
84 android:layout\_column="2"
85 android:padding="10sp"
86 android:text="276" />
87 </TableRow>
88
89 <TableRow
90 android:layout\_width="match\_parent"
91 android:layout\_height="match\_parent" >
92
93 <Button
94 android:id="@+id/button4"
95 android:layout\_width="wrap\_content"
96 android:layout\_height="wrap\_content"
97 android:layout\_column="1"
98 android:onClick="loadLinearLayout"
99 android:text="Voltar" />
100 </TableRow>
101 </TableLayout>



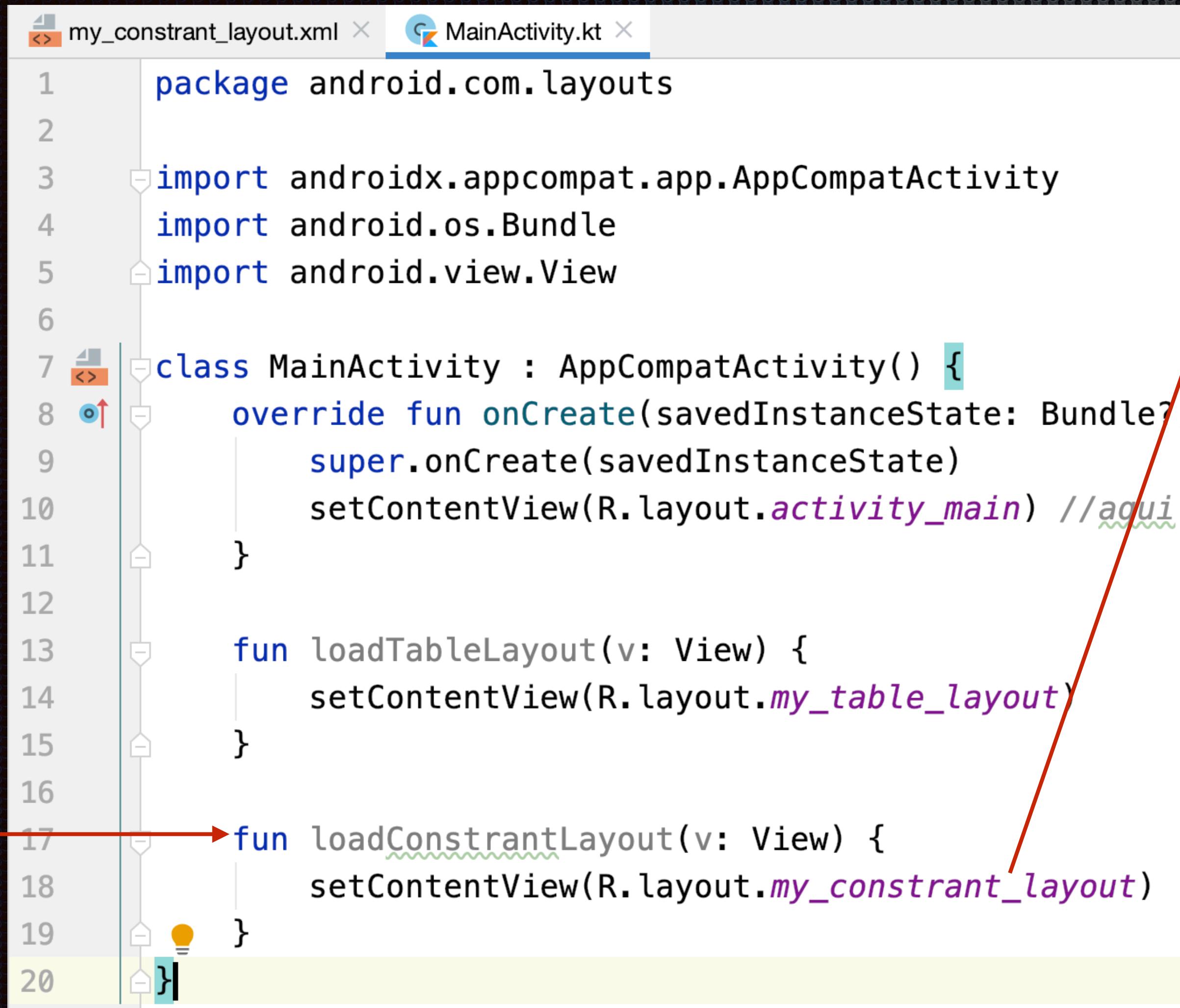
# E criar uma função no código para carregar o novo layout



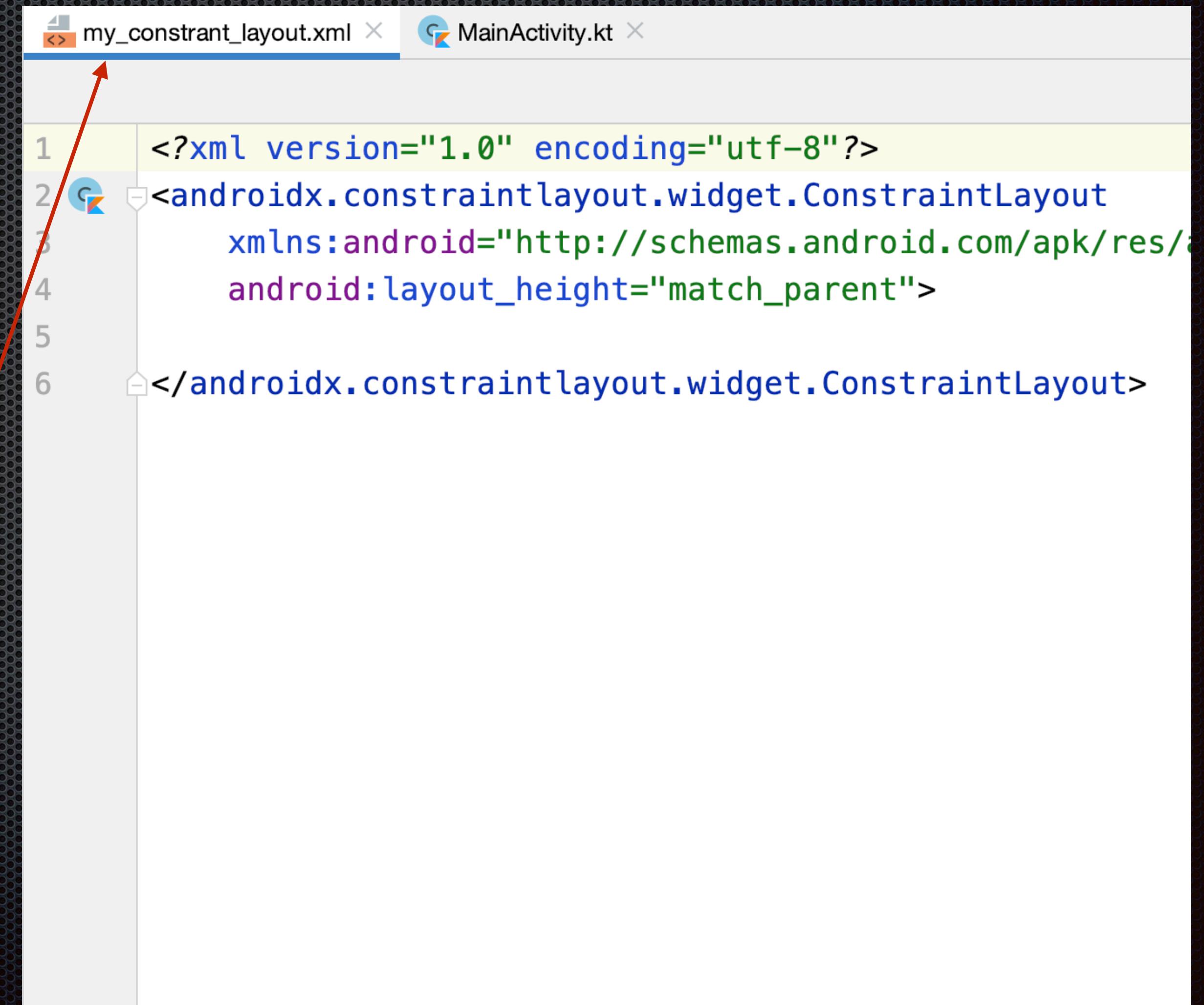
```
>MainActivity.kt
```

```
1 package android.com.layouts
2
3 import androidx.appcompat.app.AppCompatActivity
4 import android.os.Bundle
5 import android.view.View
6
7 class MainActivity : AppCompatActivity() {
8     override fun onCreate(savedInstanceState: Bundle?) {
9         super.onCreate(savedInstanceState)
10        setContentView(R.layout.activity_main) //aqui definimos o layout
11    }
12
13     fun loadTableLayout(v: View) {
14         setContentView(R.layout.my_table_layout)
15     }
16 }
```

# E faremos o mesmo procedimento para um ConstraintLayout que conterá um CalendarView

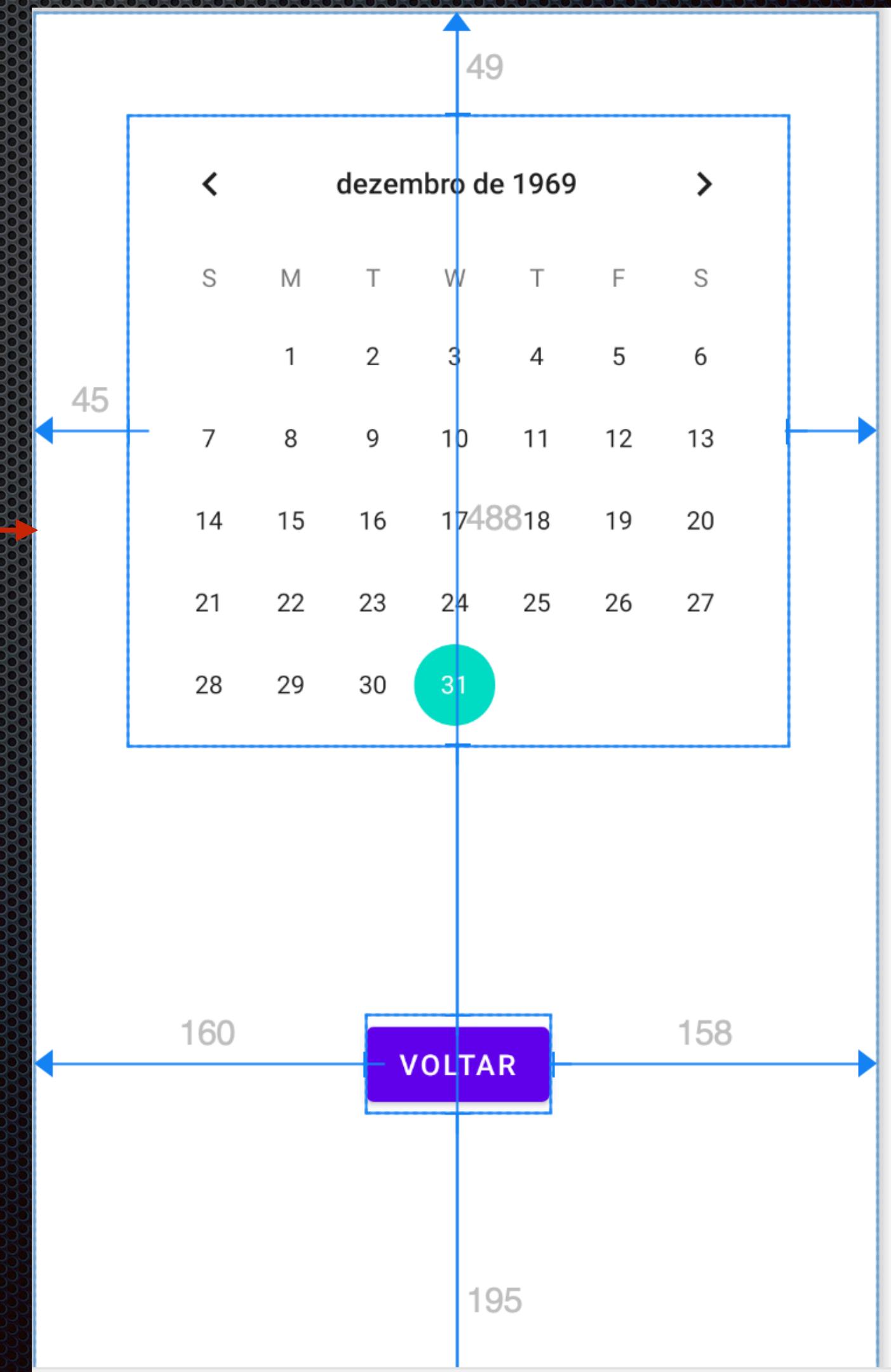
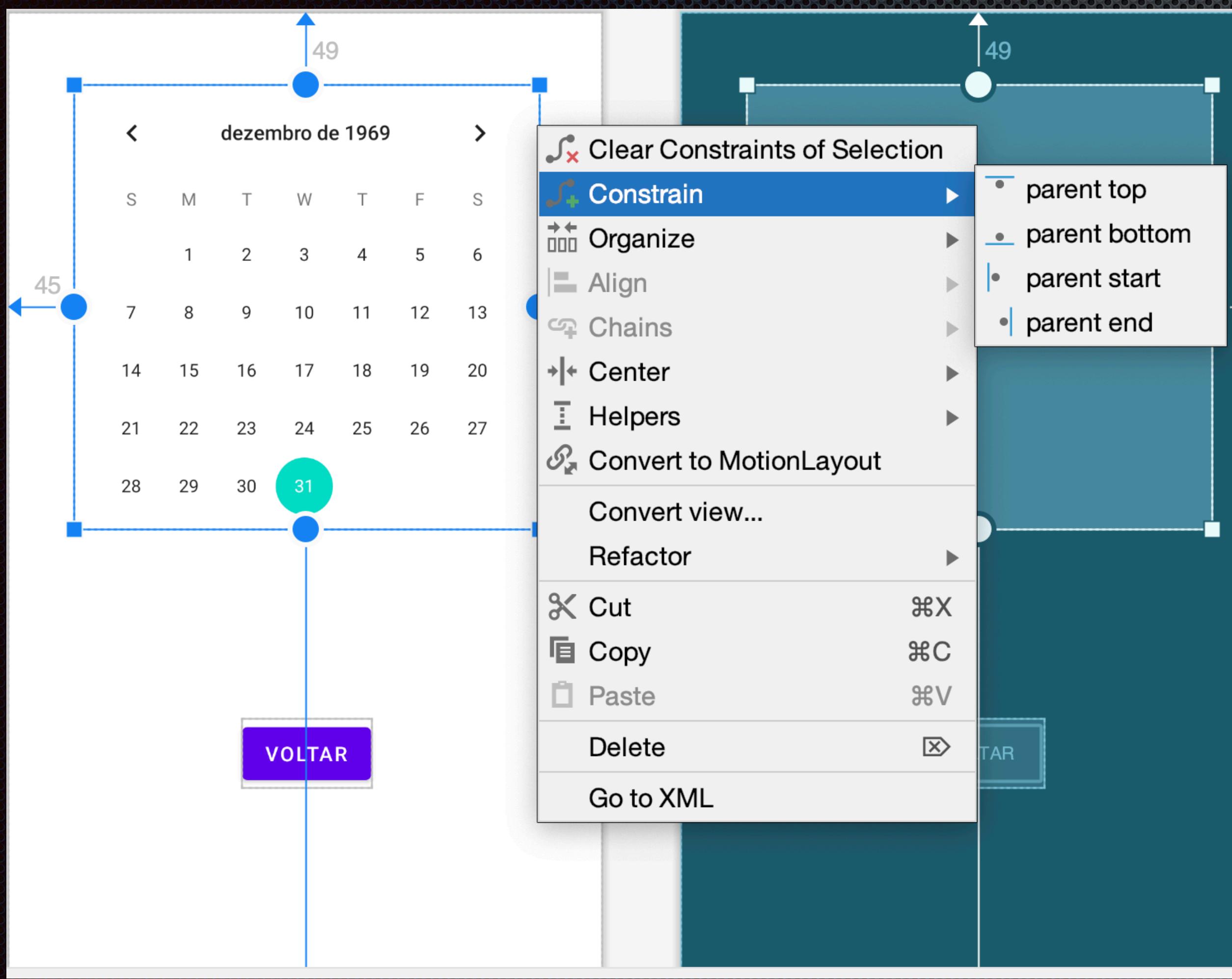


```
1 package android.com.layouts
2
3 import androidx.appcompat.app.AppCompatActivity
4 import android.os.Bundle
5 import android.view.View
6
7 class MainActivity : AppCompatActivity() {
8     override fun onCreate(savedInstanceState: Bundle?) {
9         super.onCreate(savedInstanceState)
10        setContentView(R.layout.activity_main) // aqui
11    }
12
13    fun loadTableLayout(v: View) {
14        setContentView(R.layout.my_table_layout)
15    }
16
17    → fun loadConstraintLayout(v: View) {
18        setContentView(R.layout.my_constraint_layout)
19    }
20 }
```



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     android:layout_height="match_parent">
5
6 </androidx.constraintlayout.widget.ConstraintLayout>
```

Podemos usar restrições (constraints) para deixar fixos os componentes em proporção aos limites da tela.



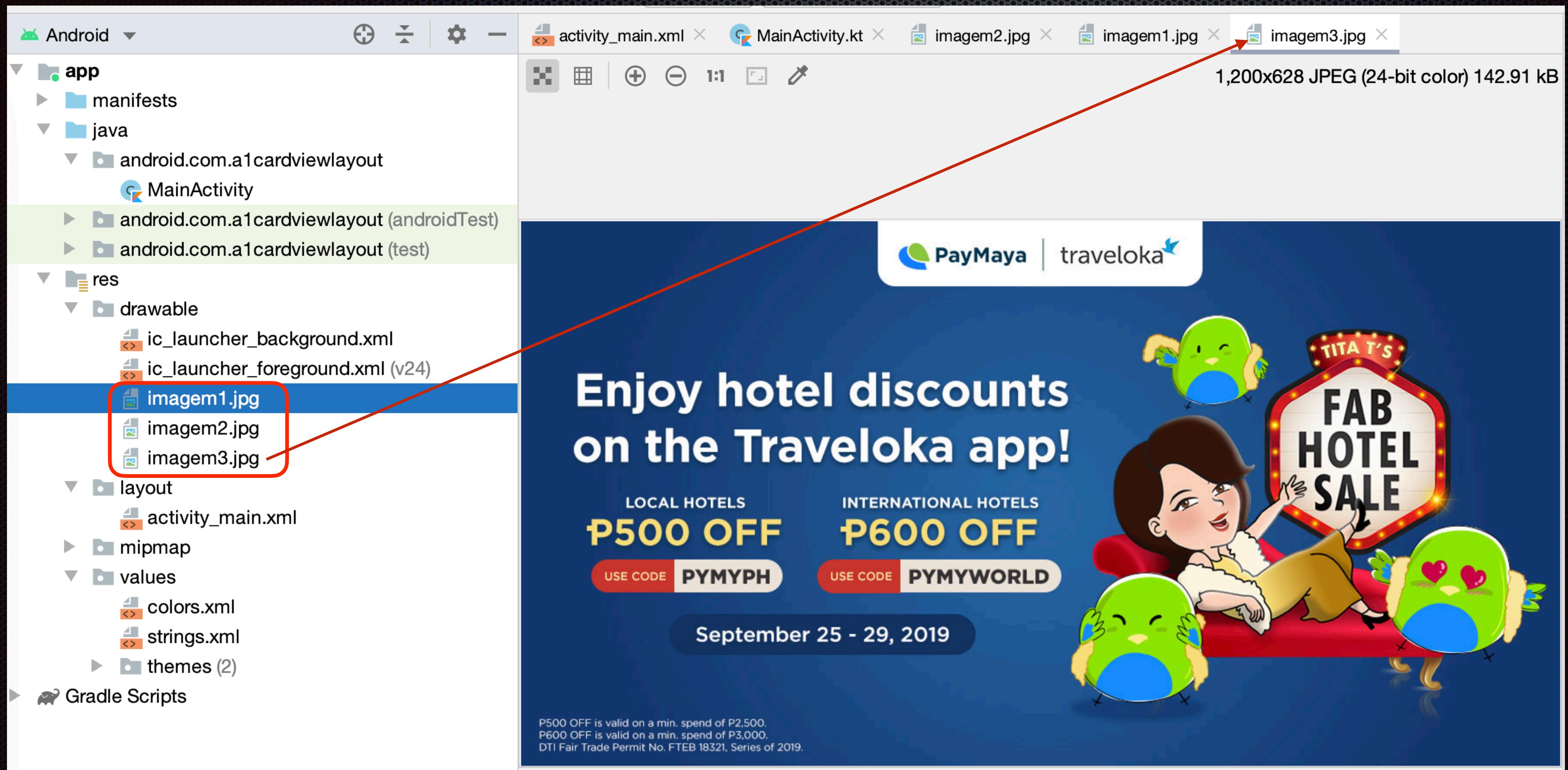
# CardView

# Layout Baseado em Cartões (Cards)

- Muitas vezes, os apps precisam exibir dados em contêineres com estilo semelhante.
- Esses contêineres são frequentemente usados em listas para armazenar as informações de cada item.
- O sistema fornece a API CardView como uma maneira fácil de mostrar informações dentro de cards que têm uma aparência consistente em toda a plataforma.



# Precisamos adicionar algumas imagens à pasta drawable



# Criaremos um Arquivo de Layout para cada Imagem

The screenshot shows the Android Studio interface with the XML layout editor open. The title bar indicates the current file is `card1.xml`. The XML code is as follows:

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical"  
    tools:layout_editor_absoluteX="21dp"  
    tools:layout_editor_absoluteY="27dp">  
  
    <TextView  
        android:id="@+id/textView"  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        android:text="@string/name1"  
        android:textSize="30sp"  
        android:textStyle="bold" />  
  
    <ImageView  
        android:id="@+id/imageView"  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        app:srcCompat="@drawable/imagem1"  
        android:contentDescription="@string/n</LinearLayout>
```

A red arrow points from the line `android:text="@string/name1"` to the `text` field in the `Text` section of the `Palette`. Another red arrow points from the line `app:srcCompat="@drawable/imagem1"` to the `Image` icon in the `Common` section of the `Palette`. The `Component Tree` panel on the right shows the structure of the layout.

# Vamos um criar um arquivo de dimensões

The screenshot shows the Android Studio interface with the project structure on the left and the code editor on the right.

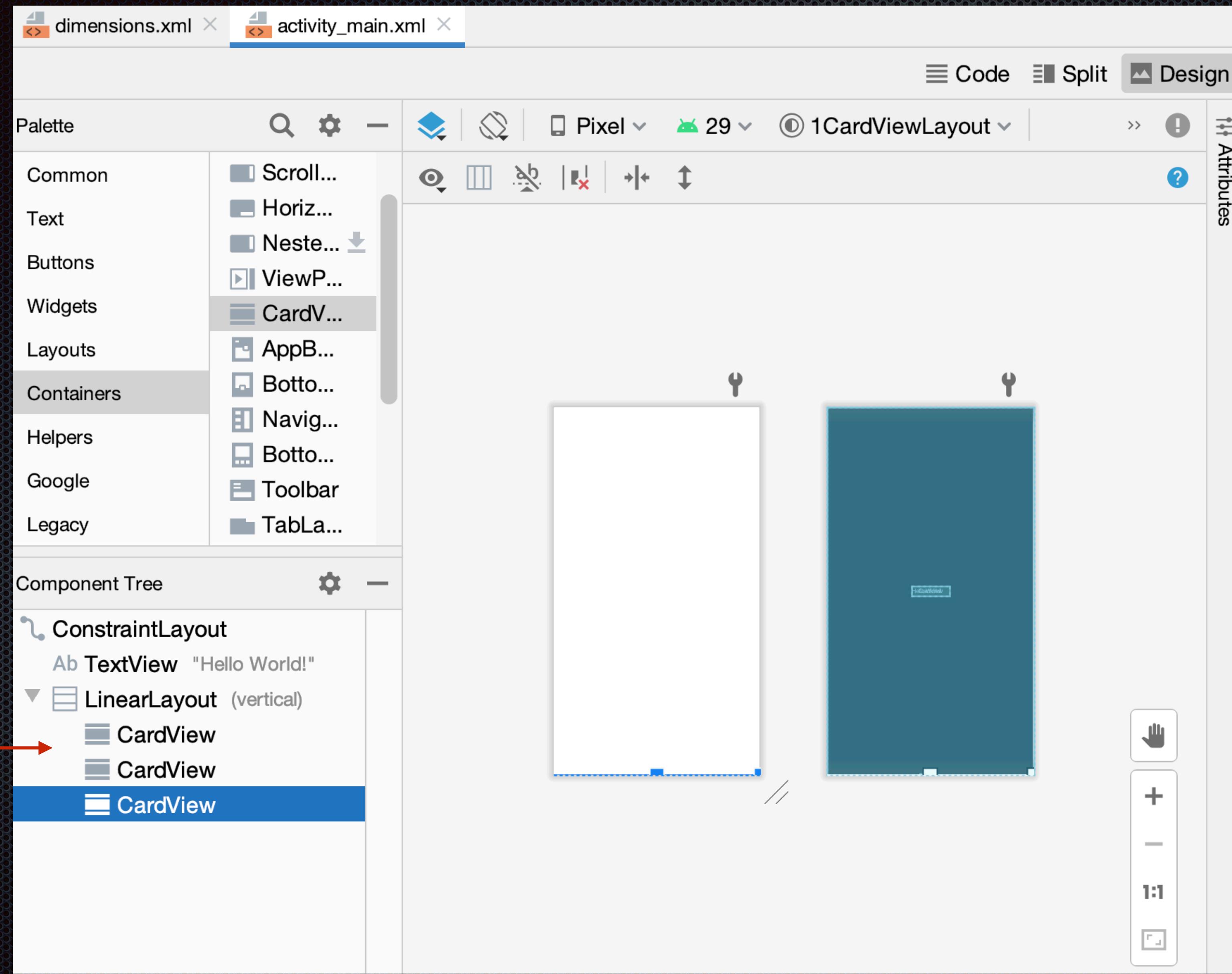
**Project Structure:**

- app
- manifests
- java
  - android.com.a1cardviewlayout
    - MainActivity
  - android.com.a1cardviewlayout (androidTest)
  - android.com.a1cardviewlayout (test)
- java (generated)
- res
  - drawable
    - ic\_launcher\_background.xml
    - ic\_launcher\_foreground.xml (v24)
    - imagem1.jpg
    - imagem2.jpg
    - imagem3.jpg
  - layout
    - activity\_main.xml
    - card1.xml
    - card2.xml
    - card3.xml
  - mipmap
  - values
    - colors.xml
    - dimensions.xml
    - strings.xml
  - themes (2)

**Code Editor (dimensions.xml):**

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <dimen name="card_corner_radius">16dp</dimen>
    <dimen name="card_margin">10dp</dimen>
</resources>
```

# E no arquivo principal, adicionar 3 CardViews em 1 LinearLayout



# Usaremos o xml de dimensões para definir o raio e as margens

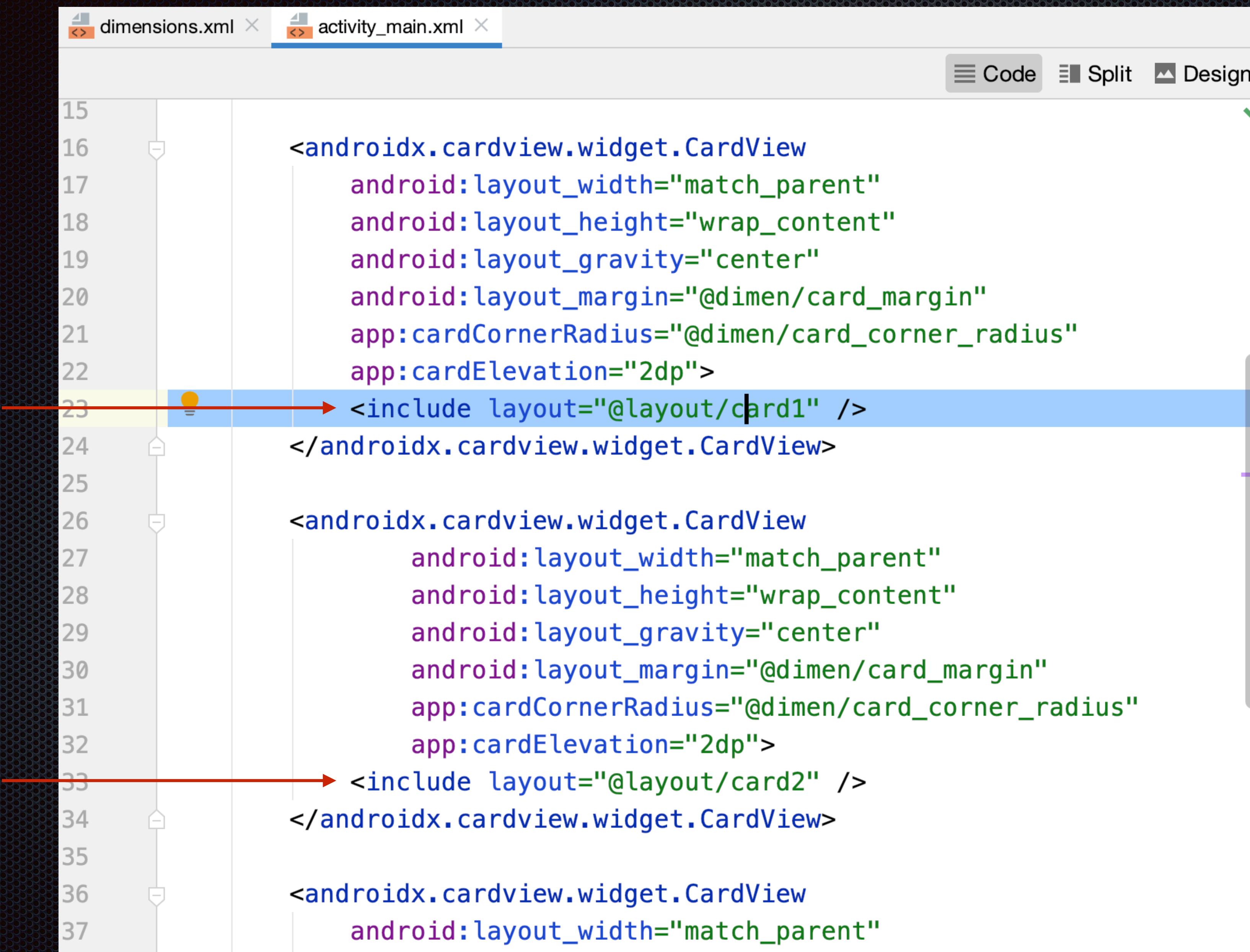
```
dimensions.xml x activity_main.xml x
Code Split Design

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:layout_editor_absoluteX="150dp"
    tools:layout_editor_absoluteY="217dp">

    <androidx.cardview.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_margin="@dimen/card_margin"
        app:cardCornerRadius="@dimen/card_corner_radius"
        app:cardElevation="2dp" />

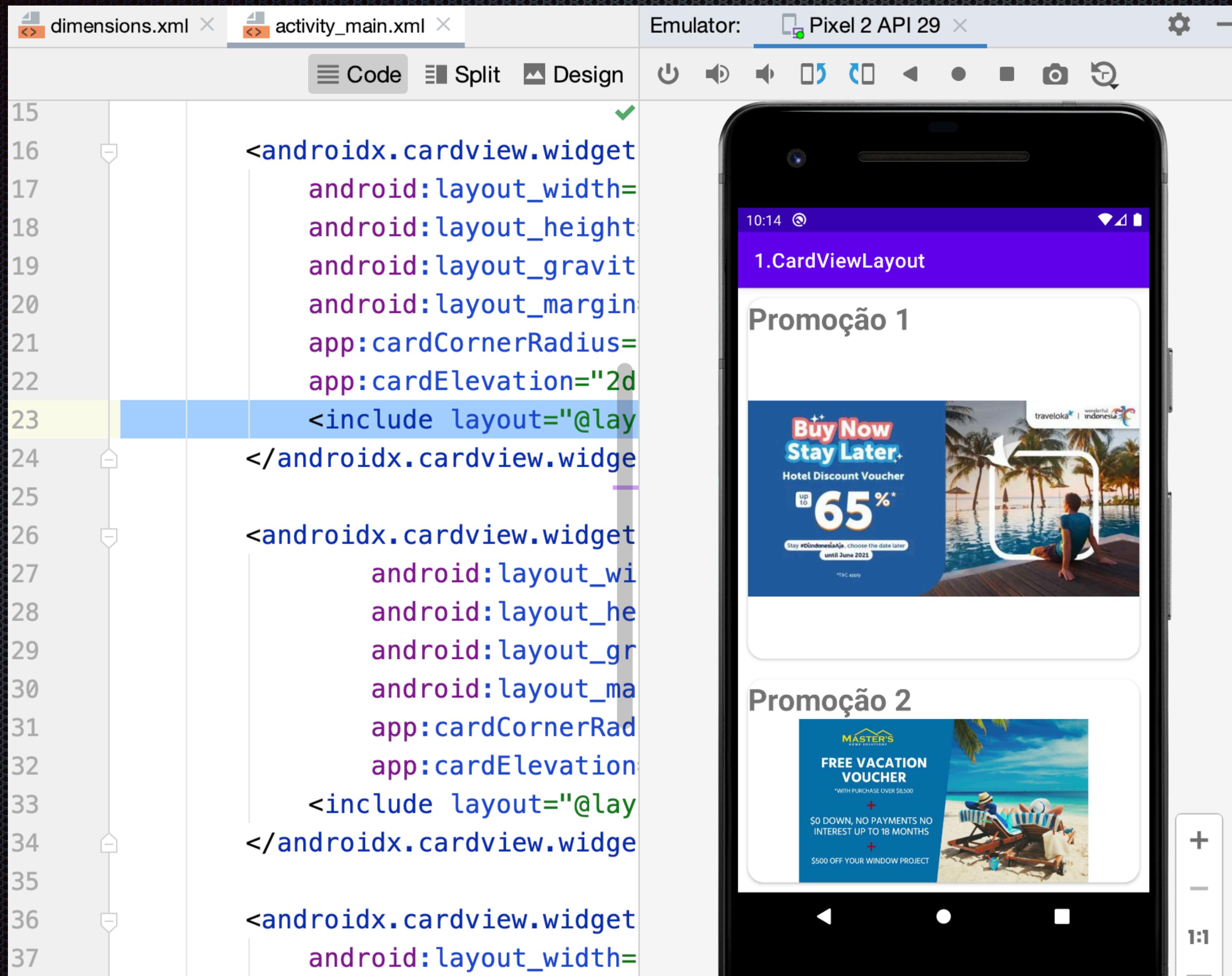
    <androidx.cardview.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_margin="@dimen/card_margin"
        app:cardCornerRadius="@dimen/card_corner_radius"
        app:cardElevation="2dp" />
```

# Adicionaremos os arquivos de layouts dentro dos Containers CardView



```
15
16     <androidx.cardview.widget.CardView
17         android:layout_width="match_parent"
18         android:layout_height="wrap_content"
19         android:layout_gravity="center"
20         android:layout_margin="@dimen/card_margin"
21         app:cardCornerRadius="@dimen/card_corner_radius"
22         app:cardElevation="2dp">
23         <include layout="@layout/card1" />
24     </androidx.cardview.widget.CardView>
25
26     <androidx.cardview.widget.CardView
27         android:layout_width="match_parent"
28         android:layout_height="wrap_content"
29         android:layout_gravity="center"
30         android:layout_margin="@dimen/card_margin"
31         app:cardCornerRadius="@dimen/card_corner_radius"
32         app:cardElevation="2dp">
33         <include layout="@layout/card2" />
34     </androidx.cardview.widget.CardView>
35
36     <androidx.cardview.widget.CardView
37         android:layout_width="match_parent"
```

Agora temos o layout criado, mas não conseguimos rolar a tela para ver a promoção 3



The screenshot shows the Android Studio interface with the activity\_main.xml file open in the code editor. The code defines three CardViewLayouts, each containing an include tag pointing to another layout file. The emulator on the right displays the first two card views, titled 'Promoção 1' and 'Promoção 2', which show promotional banners for travel discounts.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <androidx.cardview.widget.CardView
        android:id="@+id/cardView1"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_margin="10dp"
        android:layout_weight="1"
        android:background="#f0f0f0"
        android:elevation="2dp"
        android:radius="10dp"
        app:cardCornerRadius="10dp"
        app:cardElevation="2dp"
        app:strokeWidth="1dp"
        tools:ignore="ContentDescription">
        <include layout="@layout/include_promotion"/>
    </androidx.cardview.widget.CardView>

    <androidx.cardview.widget.CardView
        android:id="@+id/cardView2"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_margin="10dp"
        android:layout_weight="1"
        android:background="#f0f0f0"
        android:elevation="2dp"
        android:radius="10dp"
        app:cardCornerRadius="10dp"
        app:cardElevation="2dp"
        app:strokeWidth="1dp"
        tools:ignore="ContentDescription">
        <include layout="@layout/include_promotion"/>
    </androidx.cardview.widget.CardView>

    <androidx.cardview.widget.CardView
        android:id="@+id/cardView3"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_margin="10dp"
        android:layout_weight="1"
        android:background="#f0f0f0"
        android:elevation="2dp"
        android:radius="10dp"
        app:cardCornerRadius="10dp"
        app:cardElevation="2dp"
        app:strokeWidth="1dp"
        tools:ignore="ContentDescription">
        <include layout="@layout/include_promotion"/>
    </androidx.cardview.widget.CardView>
</androidx.constraintlayout.widget.ConstraintLayout>
```

# ScrollView

Para isto, precisamos adicionar um ScrollView para ter a barra de rolagem

The screenshot shows the Android Studio interface with the XML code for `activity_main.xml` on the left and a Pixel 2 API 29 emulator on the right.

**XML Code:**

```
<?xml version="1.0" encoding="utf-8"?>
<ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical"
            tools:layout_editor_absoluteX="150dp"
            tools:layout_editor_absoluteY="217dp">

            <androidx.cardview.widget.CardView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_gravity="center"
                android:layout_margin="@dimen/card_margin">
        
```

**Emulator Screen:**

The emulator displays a purple header with the text "1.CardViewLayout". Below it, there is a white area containing the text "Promoção 3". At the bottom, there is a dark blue banner for "Traveloka" advertising hotel discounts with offers like "P500 OFF" and "P600 OFF".

# Links

- <https://www.jetbrains.com/help/idea/reformat-and-rearrange-code.html>
- <https://github.com/udacity/Court-Counter>