

Linguagem e Técnica de Programação Mobile

AULA 6 – Conhecendo os tipos de estruturas de layout do Android

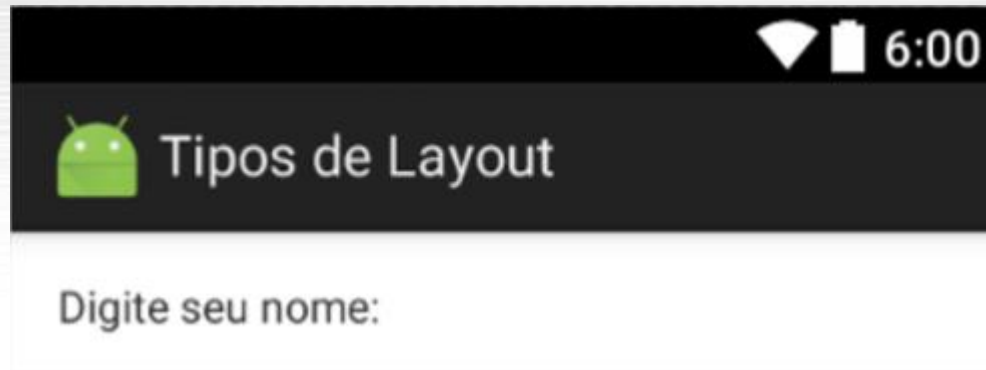
Prof. João Paulo Pimentel
joao.pimentel@projecao.br



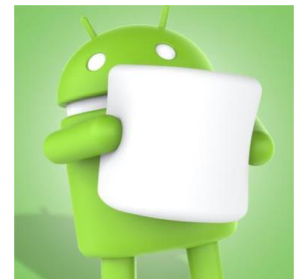
Roteiro da Aula



- Conhecendo os tipos de estruturas de layout do Android
 - RelativeLayout
 - TableLayout
 - LinearLayout
 - ConstraintLayout (está disponível em uma biblioteca de API compatível com Android 2.3 (API 9) ou superiores)
 - Construindo nossas aplicações no Android
 - Desenvolvendo uma aplicação simples de compras – Versão 3 do Android Studio
-



Conhecendo os tipos de estruturas de layout do Android





Estruturas de layout do Android



- A plataforma Android nos oferece vários tipos de estruturas de layouts que podemos utilizar em nossas aplicações.
 - Até a aula anterior (a aula 5), trabalhamos com o **RelativeLayout** (estrutura padrão das telas do dispositivo Android). Agora vamos conhecer outros tipos de estruturas de layouts disponíveis no Android.
 - Vamos começar esta aula criando primeiramente um projeto no Android Studio com os seguintes dados apresentados no próximo slide.
-



Estruturas de layout do Android



- Configurem assim:

Application Name: Tipos de Layout

Company Domain : app.usuario

Project location : (Fica a sua escolha)

Activity Name: TiposLayoutActivity

Layout Name : tela_relative_layout





Estruturas de layout do Android



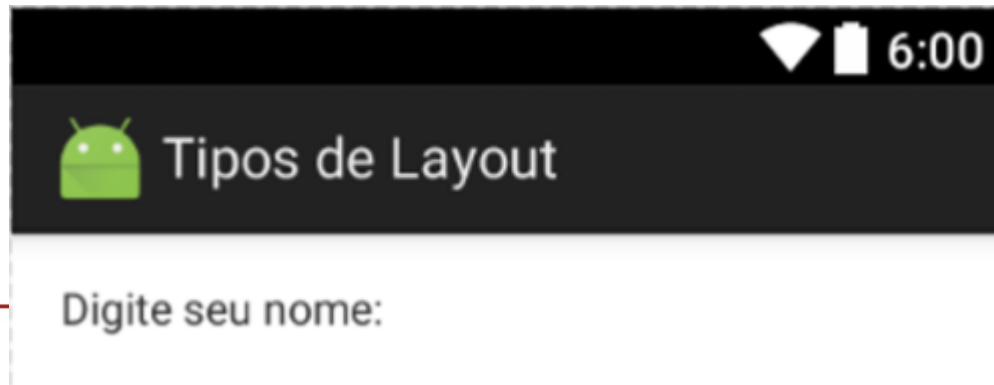
- **RelativeLayout**
 - A estrutura **RelativeLayout** (padrão do Android Studio) organiza os componentes relativamente sobre outros componentes já inseridos na estrutura.
 - Quando criamos um projeto no Android Studio, automaticamente, a estrutura de layout padrão utilizada pela tela do dispositivo Android é **RelativeLayout**.
-



RelativeLayout

Estruturas de layout do Android

- Na tela do dispositivo Android visualizamos uma frase escrita “**Hello world**”. Vamos selecionar essa frase (o componente **TextView**) e em seguida na propriedade “**text**” desse vamos digitar a seguinte frase: “**Digite seu nome**”. E na propriedade **ID** como um nome para o componente, chamando-o de “**textViewNome**” Veja o resultado na figura seguinte:





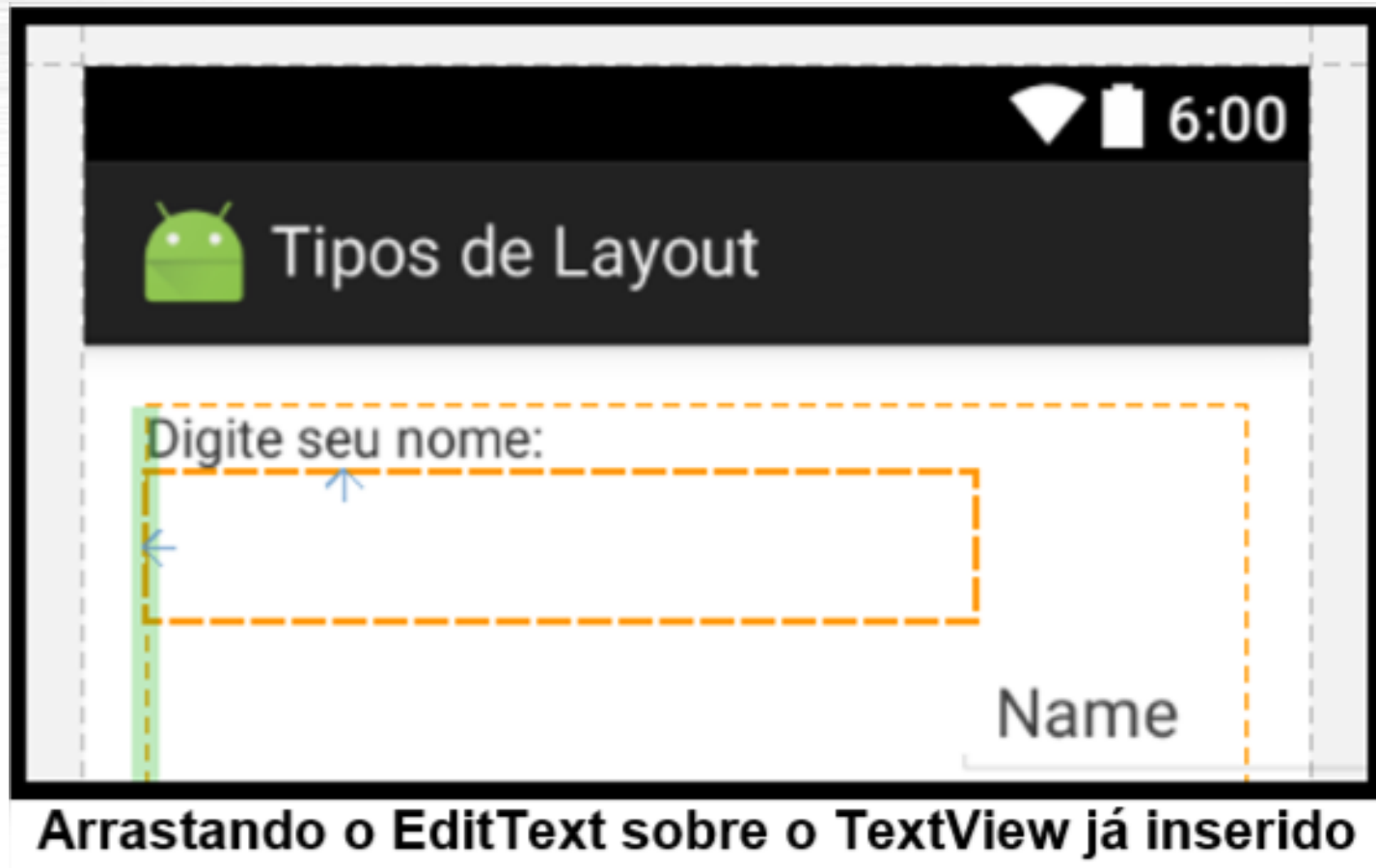
Estruturas de layout do Android



- Agora vamos inserir um componente do tipo **EditText**, porém, com o seguinte detalhe: iremos arrastá-lo sobre o componente **TextView** que acabamos de inserir e “não diretamente” na estrutura **RelativeLayout**, pois a organização do componente inserido é baseada relativamente em outro componente já inserido anteriormente. Se arrastarmos o componente **EditText** sobre o componente **TextView** será exibida algumas “guias” nas quais você pode ver para organizar seu componente, conforme você confere no próximo slide.
-

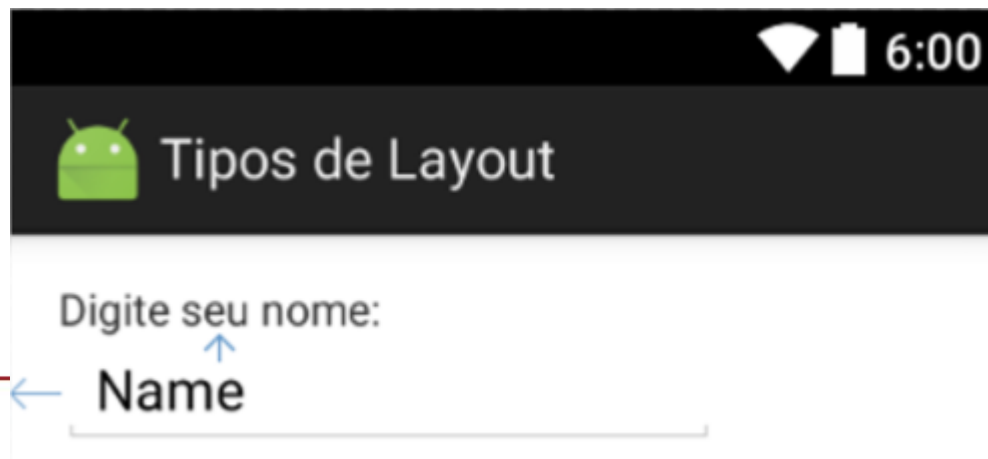
Estruturas de layout do Android

- EditText sobre o TextView já inserido fica assim:



Estruturas de layout do Android

- Vamos colocar o nosso componente **EditText** de forma fique abaixo do nosso **TextView** já inserido colocando-o ele em uma das guias exibidas, conforme a indicação da figura anterior.
- Escolhido a posição desejada, bastar soltar o botão do mouse e o componente será inserido, conforme você observa na figura abaixo:





Estruturas de layout do Android



- Agora mude as propriedades da **EditText** conforme tabela seguinte:

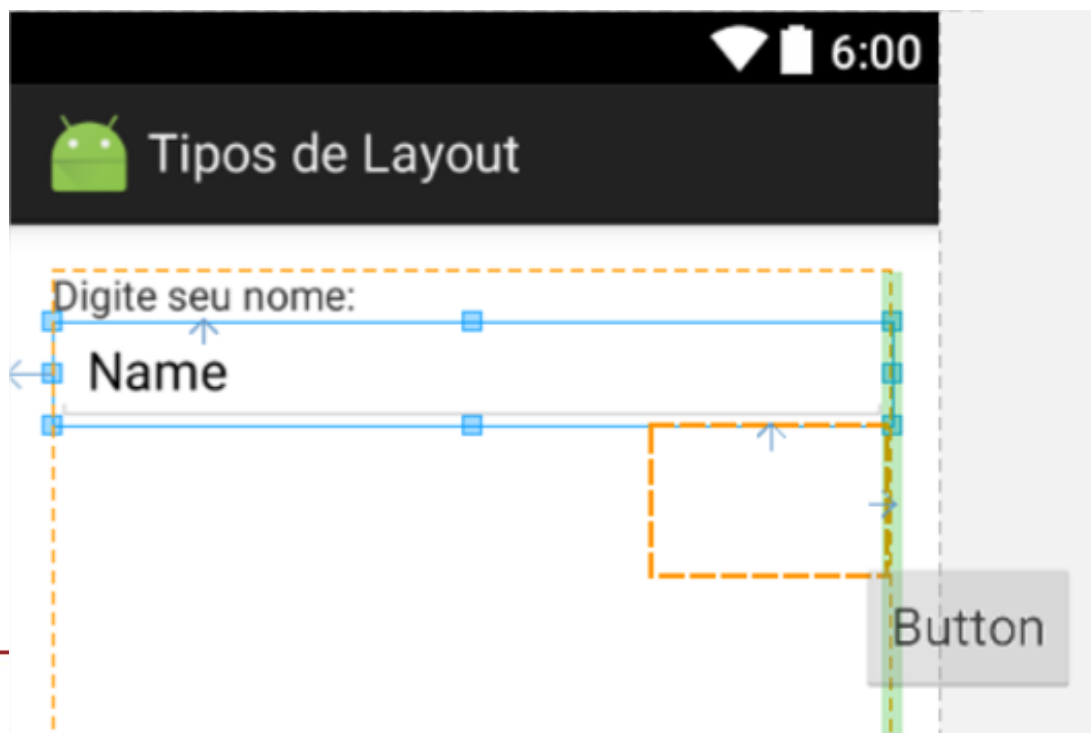
EditText

Propriedade	Valor
Text	(Deixar em branco)
layout:width	match_parent



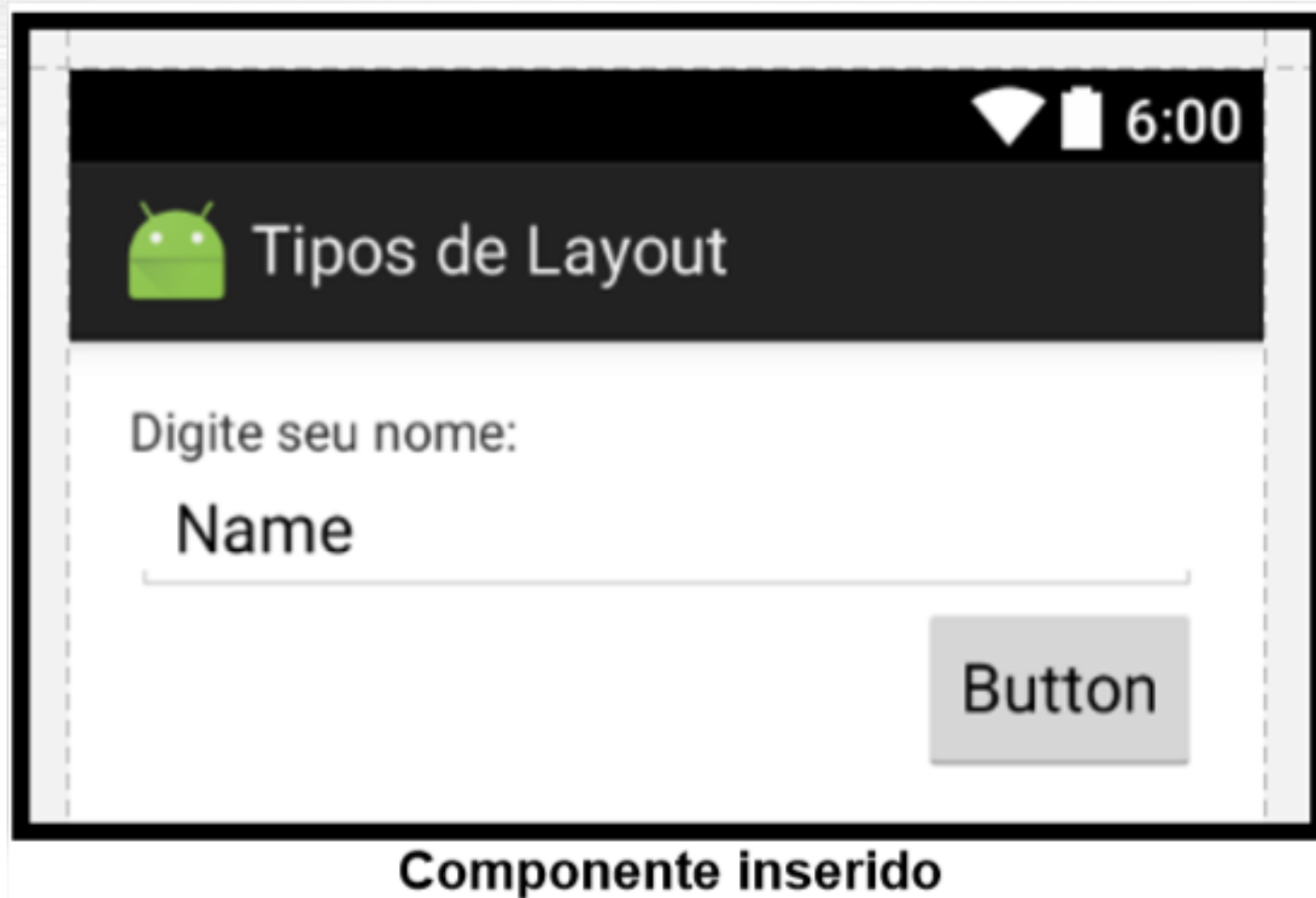
Estruturas de layout do Android

- Agora vamos adicionar um componente **Button** em nossa aplicação. Arraste o componente **Button** sobre o componente **EditText** e em seguida, coloque ele no local conforme mostra a figura abaixo:



Estruturas de layout do Android

- Veja como ficou o resultado na figura seguinte:



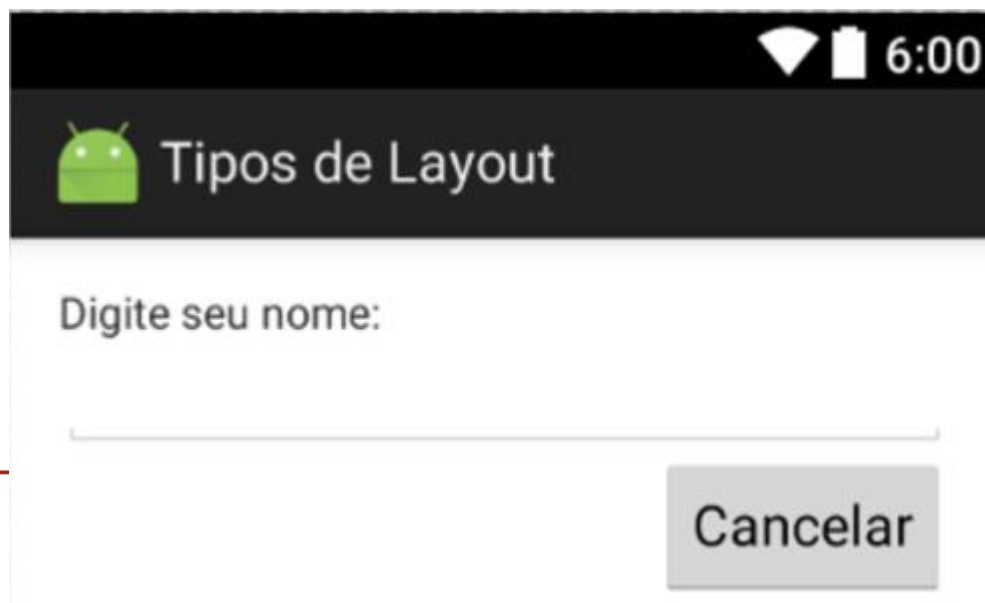
Estruturas de layout do Android

- Agora vamos alterar as propriedades do componente **Button** conforme a tabela abaixo:

Button

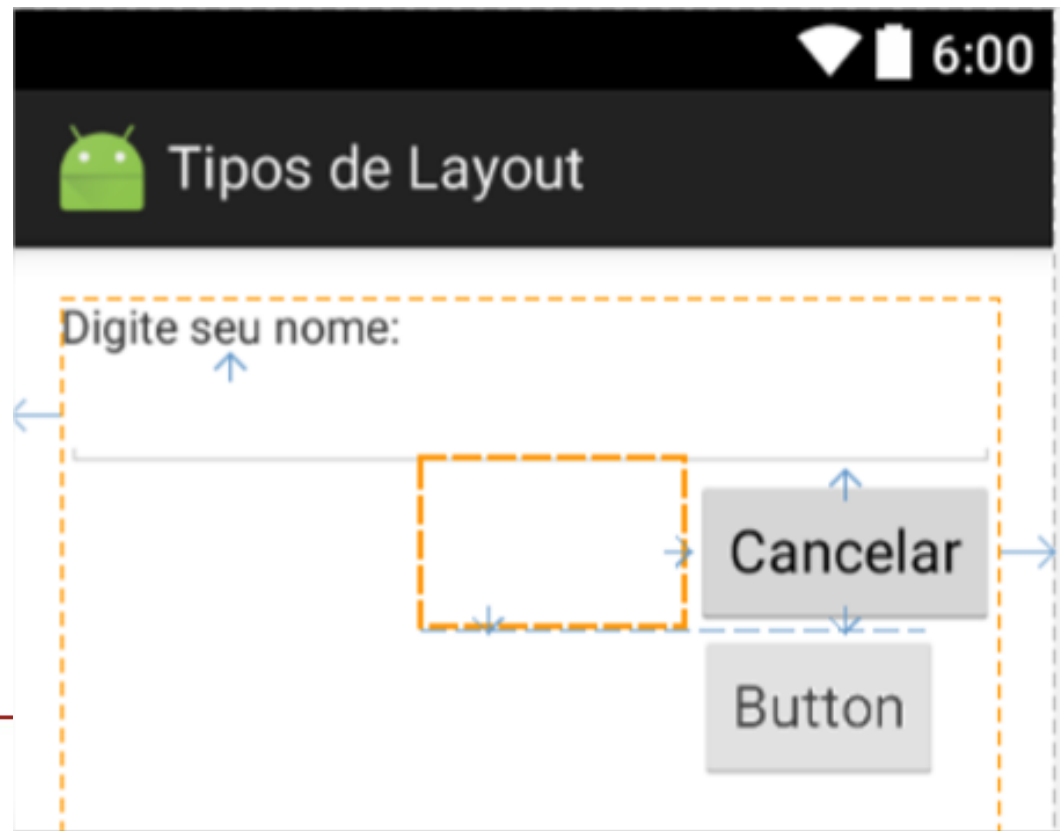
Propriedade	Valor
text	Cancelar

- Vejamos como ficou na figura seguinte :



Estruturas de layout do Android

- Agora vamos arrastar um outro componente **Button** sobre o componente anteriormente inserido, de forma que ele fique do lado esquerdo dele, conforme demonstra a figura seguinte:



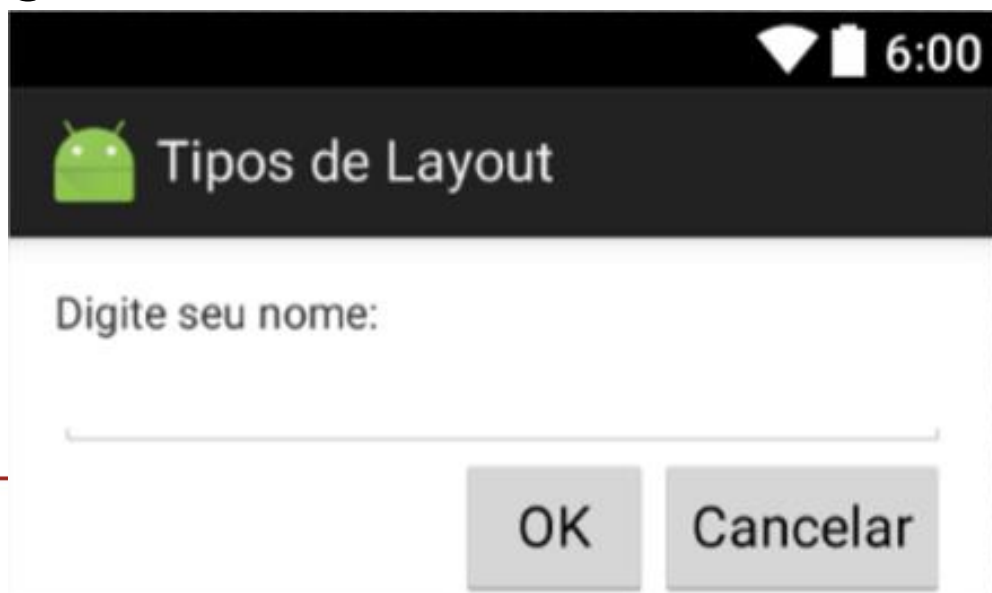
Estruturas de layout do Android

- Agora vamos alterar as propriedades do componente **Button** conforme a tabela abaixo:

Button

Propriedade	Valor
text	OK

- Veja como ficou a tela da nossa aplicação na figura seguinte:



Estruturas de layout do Android

- Agora vamos conferir o código XML da tela da aplicação:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:text="Digite seu nome:" />

    <EditText
        android:id="@+id/editText1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView1" >

        <requestFocus />
    </EditText>
```

Estruturas de layout do Android

#continuação do XML da aplicação:

```
<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentRight="true"
    android:layout_below="@+id/editText1"
    android:text="Cancelar" />
```

```
<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/editText1"
    android:layout_toLeftOf="@+id/button1"
    android:text="OK" />
```

```
</RelativeLayout>
```



TableLayout



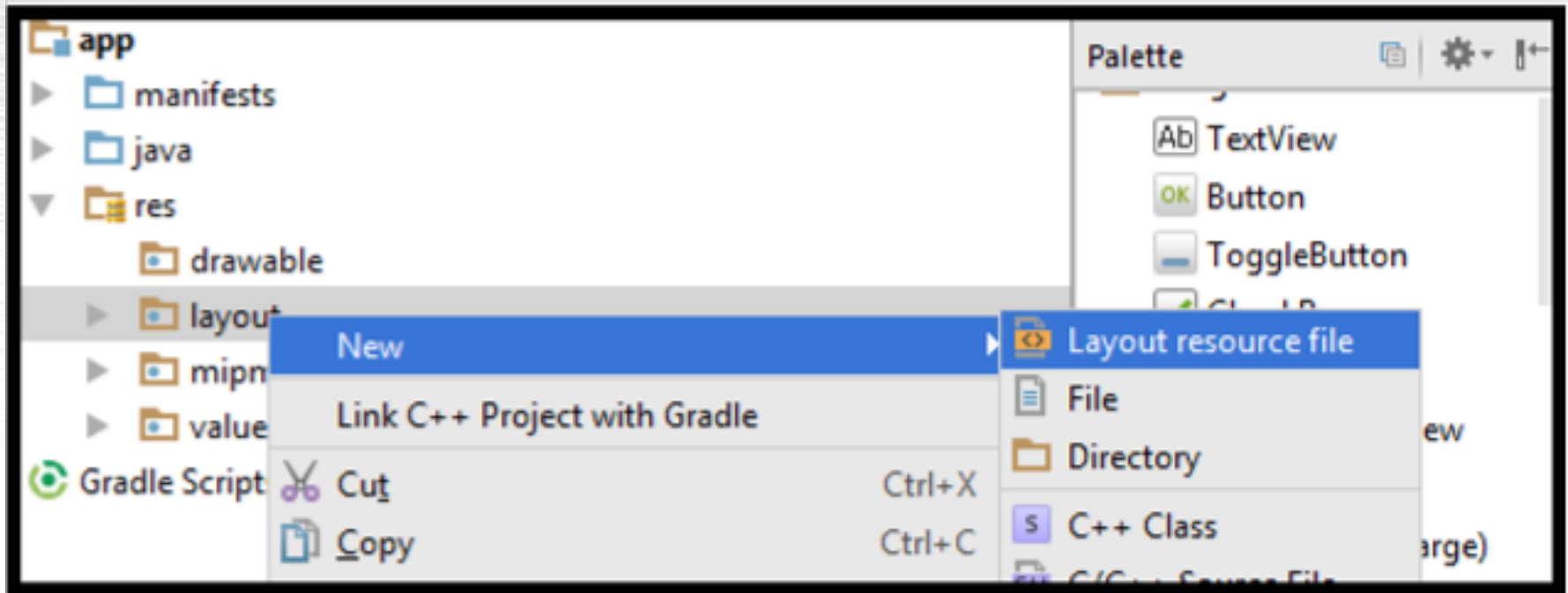
Estruturas de layout do Android



- **TableLayout**
 - A estrutura **TableLayout** “comporta” os componentes nele inseridos como se eles estivessem em uma tabela (**Table**), organizados em linhas e colunas.
 - Vamos demonstrar o uso dessa estrutura. Dentro do projeto que criamos, na pasta “**layout**”, vamos criar um arquivo **XML** chamado “**table_layout**” (cuja estrutura de layout será **TableLayout**). Para isso, clique com o botão direito sobre a pasta “**layout**” e em seguida selecione “**New**” / “**Layout Resource File**”:
-

Estruturas de layout do Android

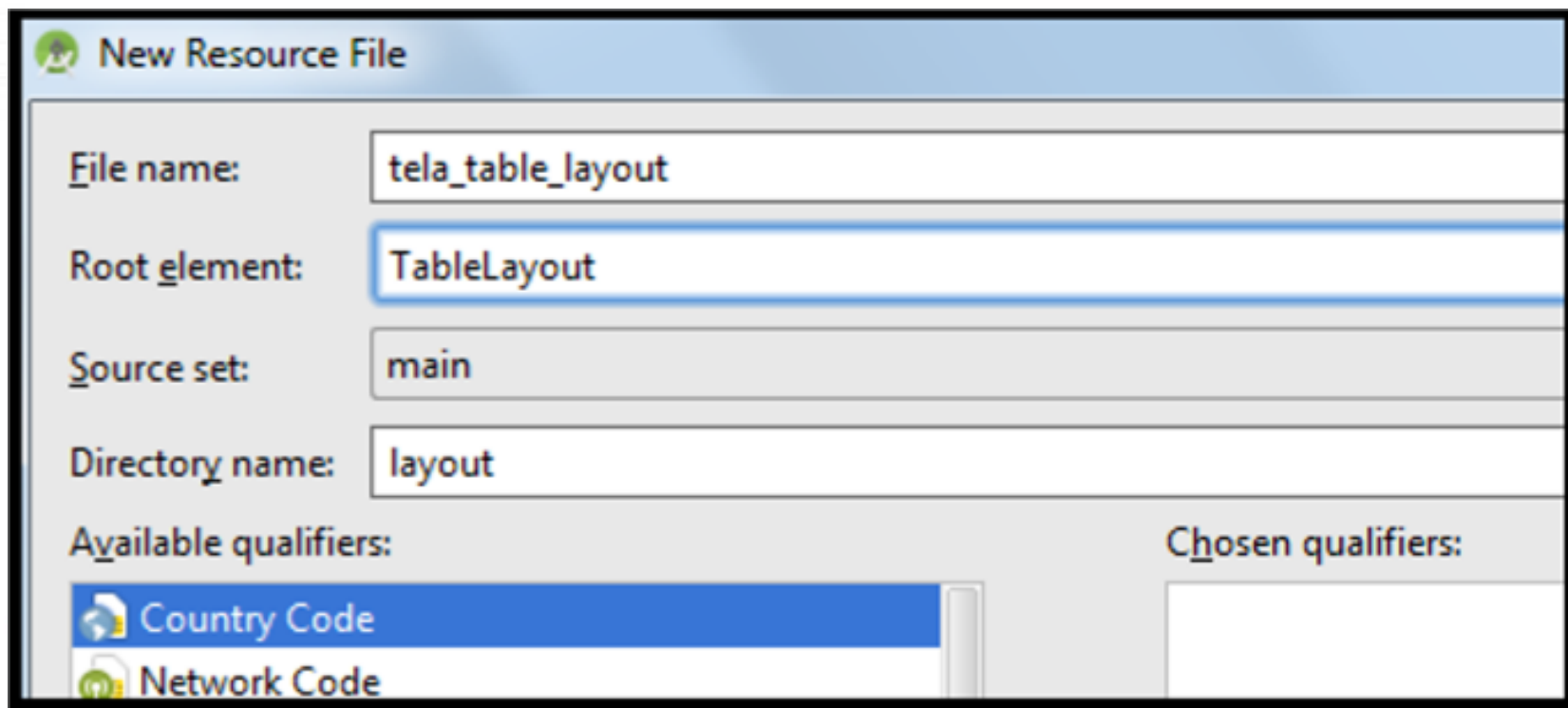
- Vamos fazer conforme a figura abaixo:



Criando um novo arquivo de Layout

Estruturas de layout do Android

- Na caixa de diálogo que se abra, insira no campo “**File name**” o nome do nosso arquivo, que se chamará “**tela_table_layout**”. No campo “**Root element**” vamos digitar “**TableLayout**”, que será a nossa estrutura do nosso arquivo.



New Resource File

File name: tela_table_layout

Root element: TableLayout

Source set: main

Directory name: layout

Available qualifiers:

- Country Code
- Network Code

Chosen qualifiers:



Estruturas de layout do Android



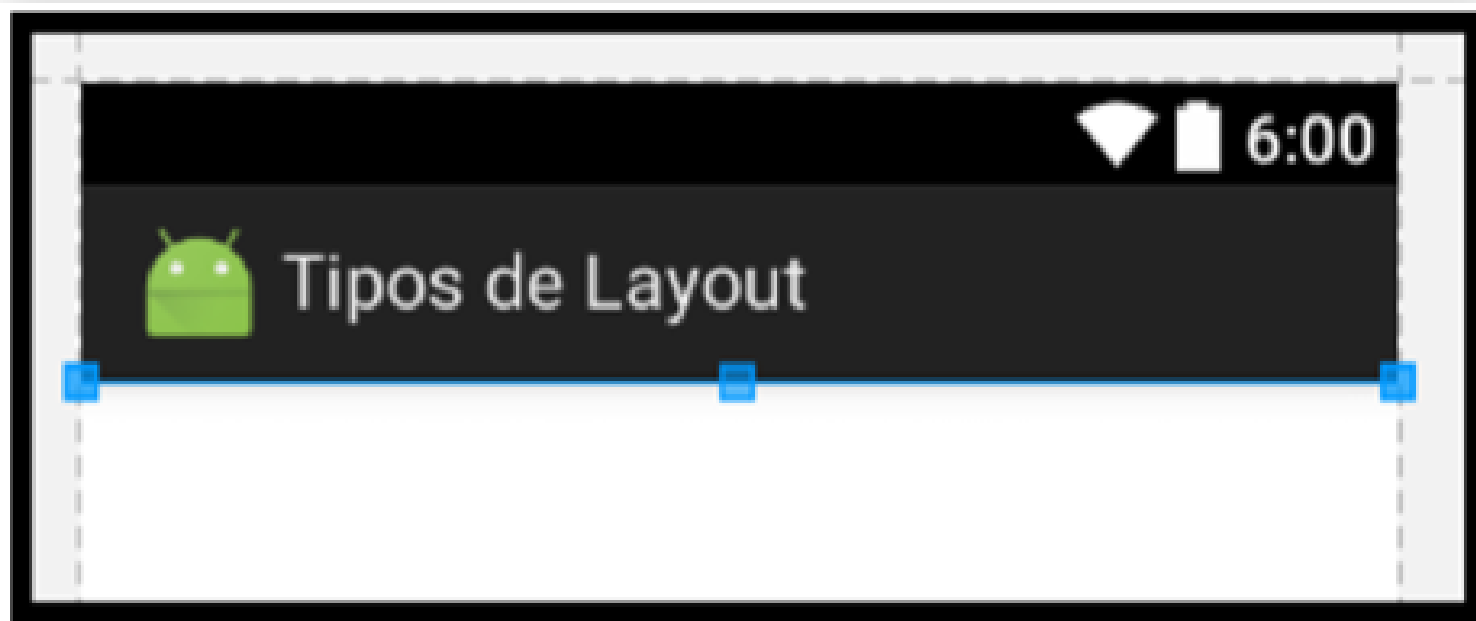
- Clique em “OK” para gerarmos o nosso arquivo.
 - Nesse tipo de estrutura, os componentes são organizados em linhas e colunas.
 - Para adicionarmos um componente nesse estrutura, é preciso antes adicionar uma outra estrutura do tipo **TableRow** (presente em “**Layouts**”).
 - Clique e arraste esse componente na tela, antes de qualquer coisa. Vejamos na figura a seguir no próximo slide.
-



Estruturas de layout do Android



- **TableRow** inserido



TableRow inserido na tela

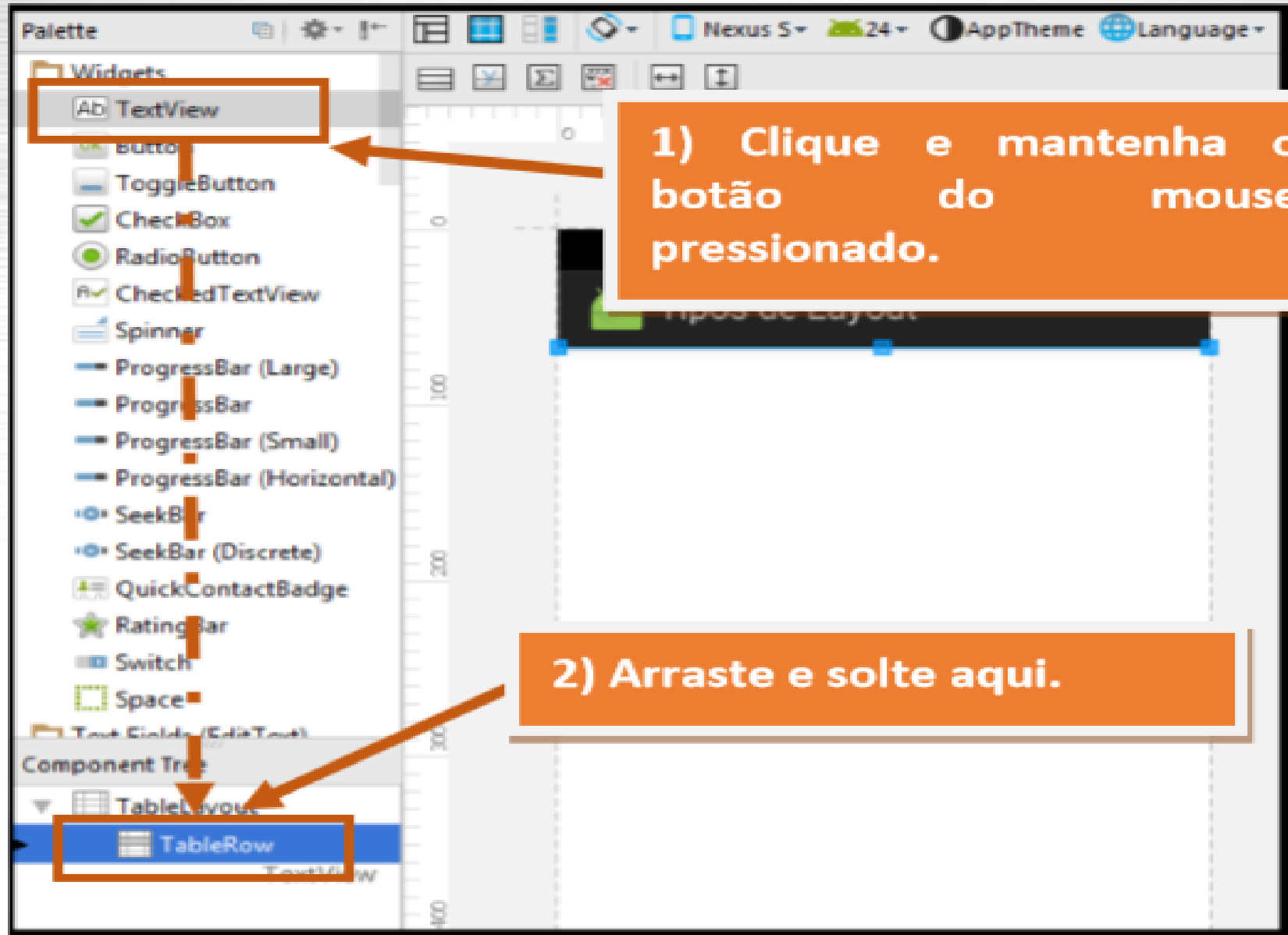


Estruturas de layout do Android



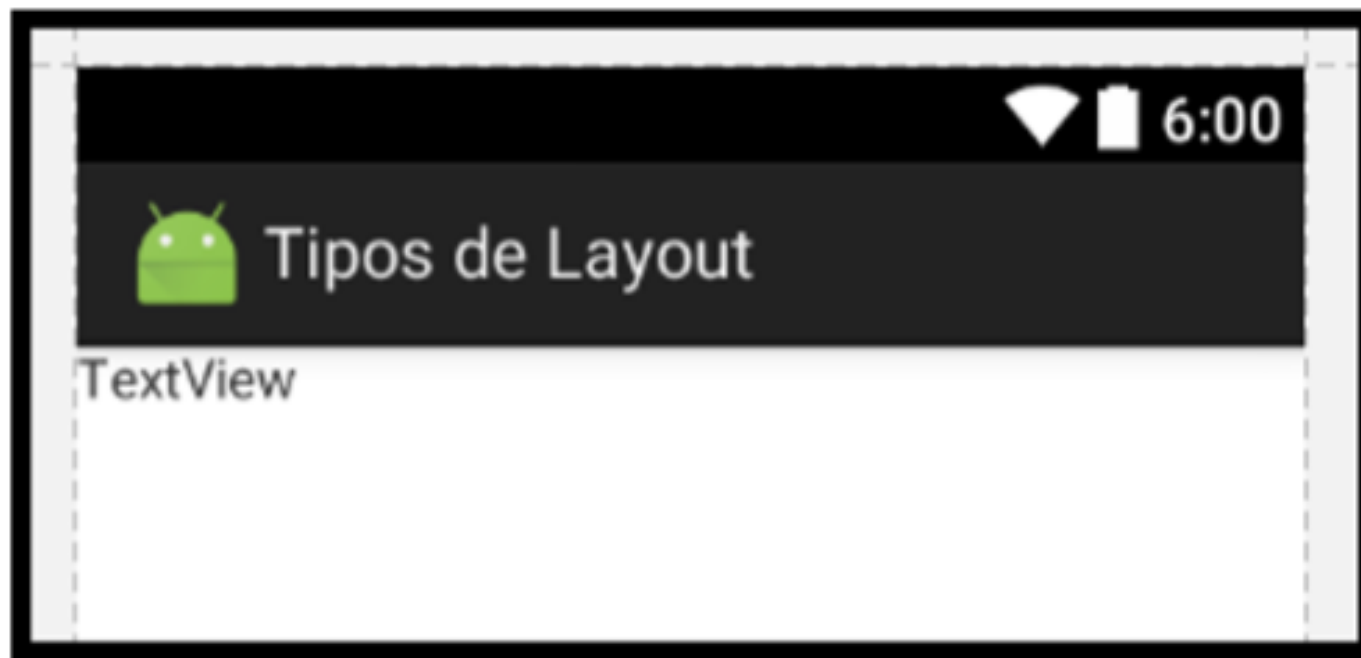
- Pelo que podemos ver, não conseguimos visualizar o **TableRow** preenchendo adequadamente a tela da nossa aplicação, porém, precisamos adicionar os elementos dentro dele.
 - Para começar, vamos adicionar um **TextView** no **TableRow** inserido. Clique no componente **TextView** e arraste até o **TableRow** presente no **Component Tree**. Veja no próximo slide.
-

Estruturas de layout do Android



Estruturas de layout do Android

- Feito isso nosso componente será inserido na tela, conforme podemos conferir em seguida:



Componente inserido na tela

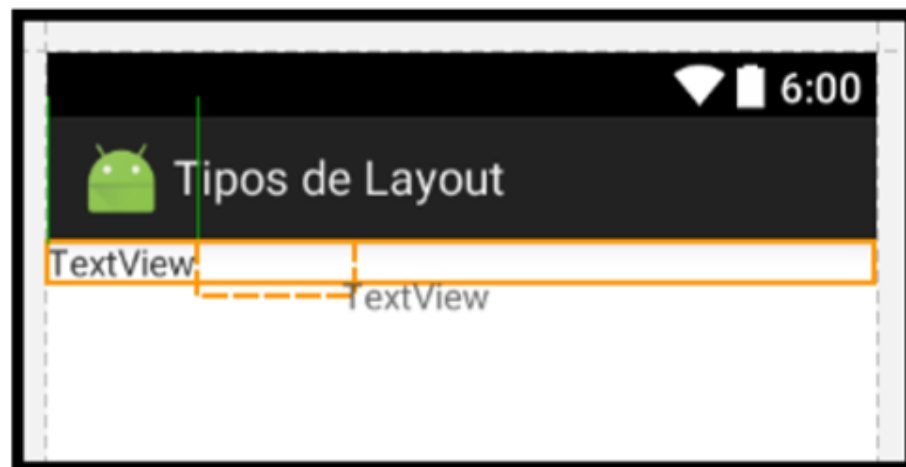


Estruturas de layout do Android



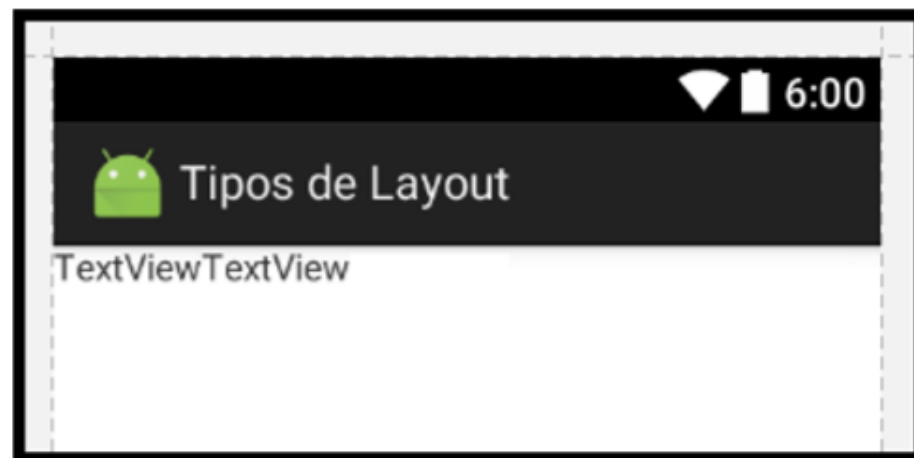
- Todo e qualquer componente presente dentro de um **TableLayout** deve estar organizado dentro de uma estrutura **TableRow** (que distribui os componentes em colunas, igual a uma tabela).
 - Vamos adicionar agora mais um componente do tipo **TextView**. Para isso, nesta situação agora, basta arrastar o componente ao lado do outro objeto, conforme mostra a figura a seguir no próximo slide.
-

Estruturas de layout do Android



Colocando o componente ao lado do objeto TextView

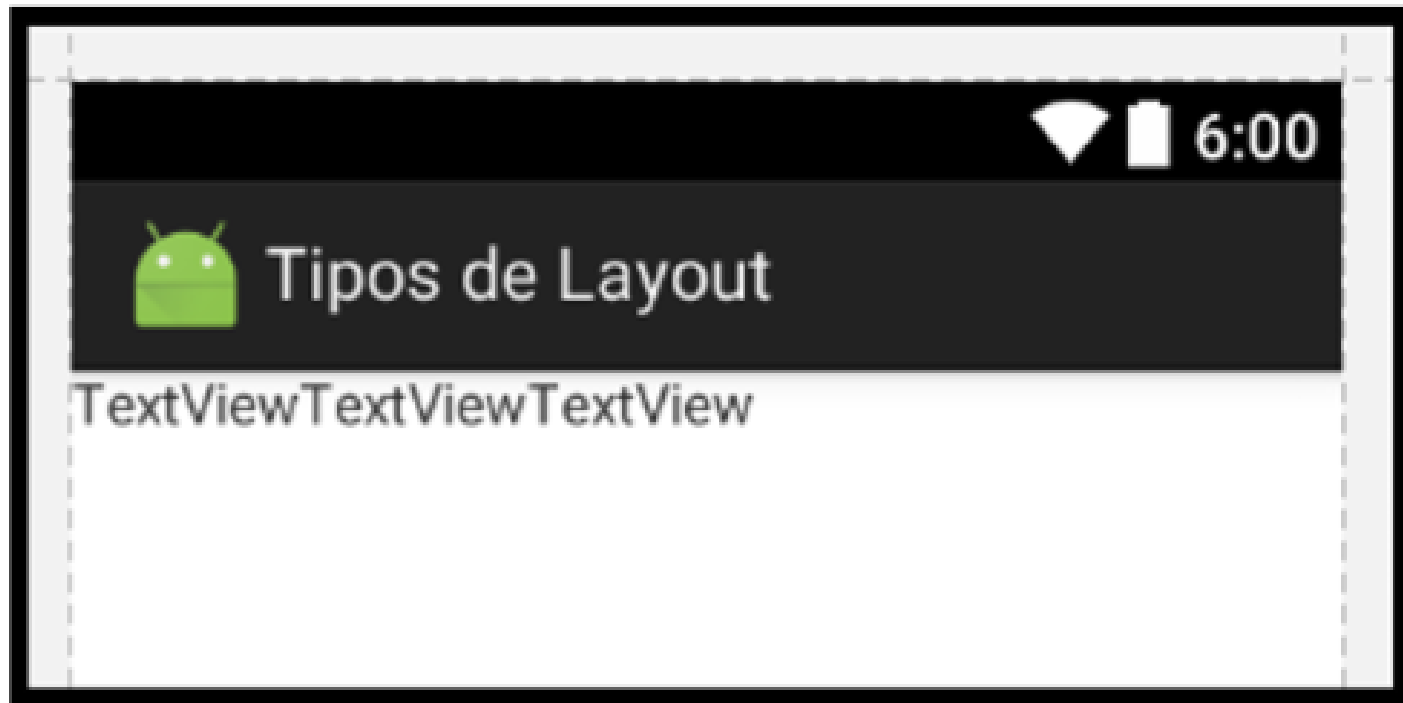
- Vejamos o resultado:



Componente inserido

Estruturas de layout do Android

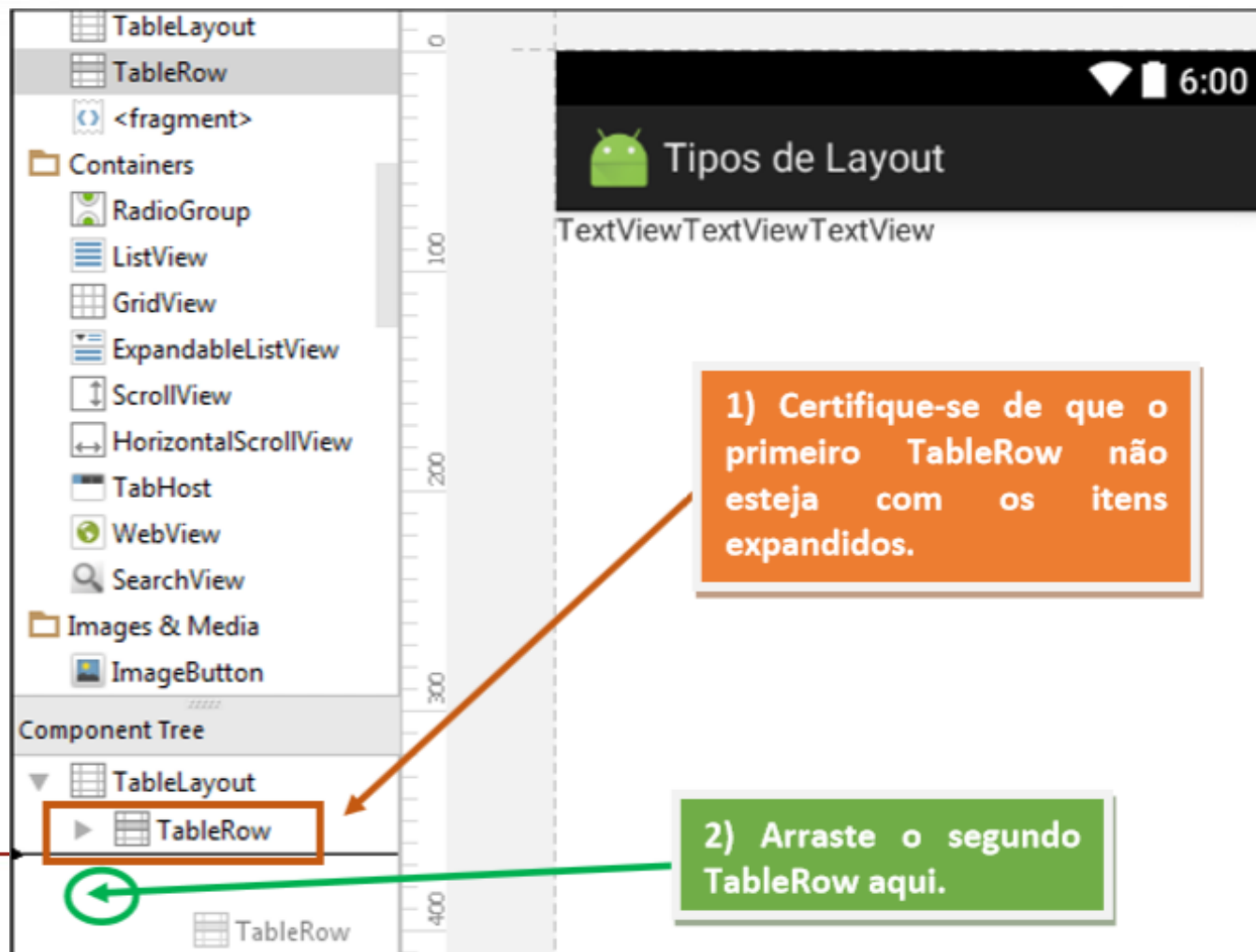
- Insira mais um componente do tipo **TextView** ao lado do objeto **TextView** já inserido anteriormente. Vejamos o resultado:



Componente inserido

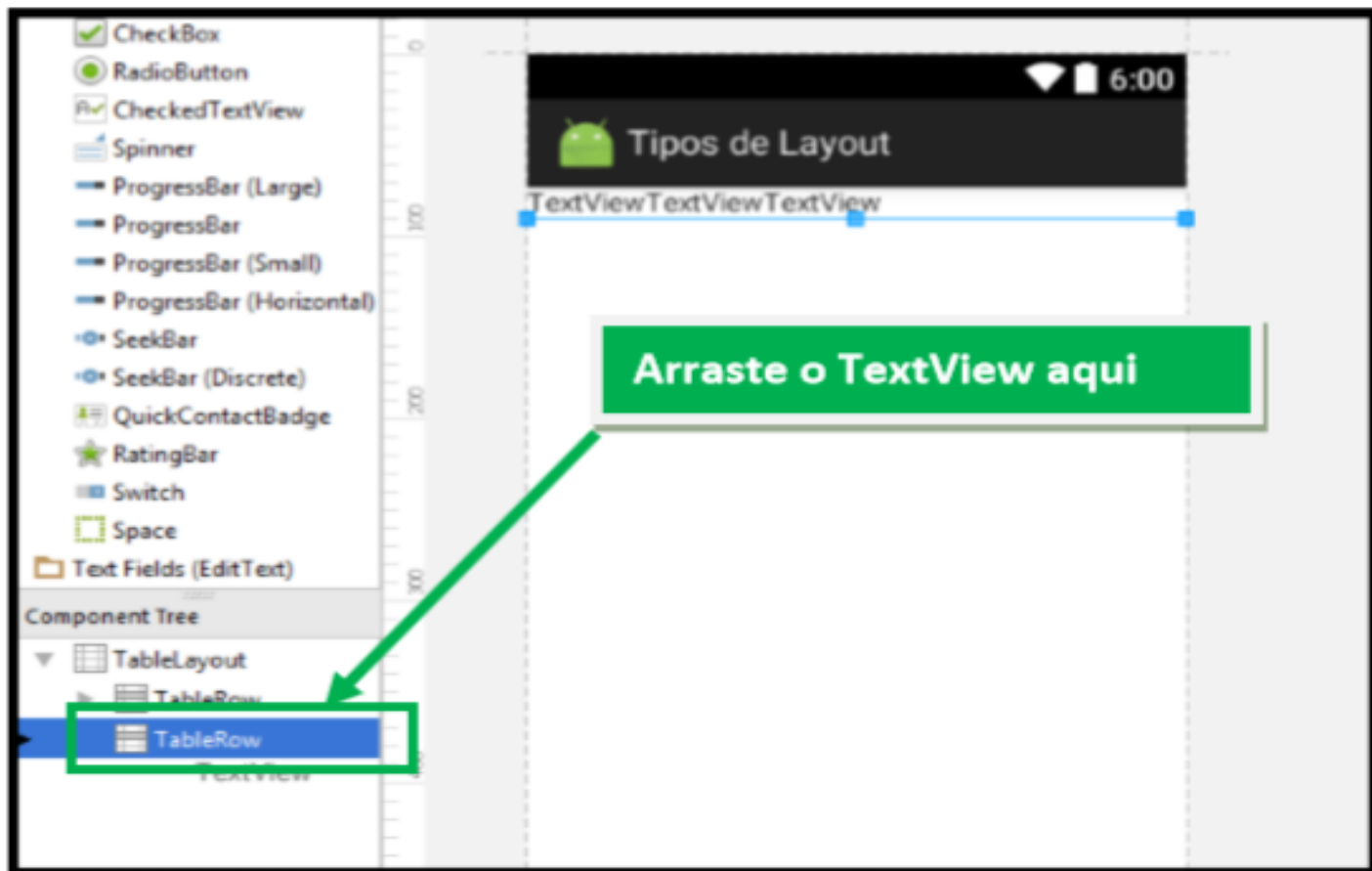
Estruturas de layout do Android

- Vamos agora colocar mais um componente do tipo **TableRow**, conforme orientação da imagem a seguir:



Estruturas de layout do Android

- Agora insira mais um componente **TextView** na estrutura **TableRow** que inserimos agora (não pela tela, mas sim pelo **Component Tree**).



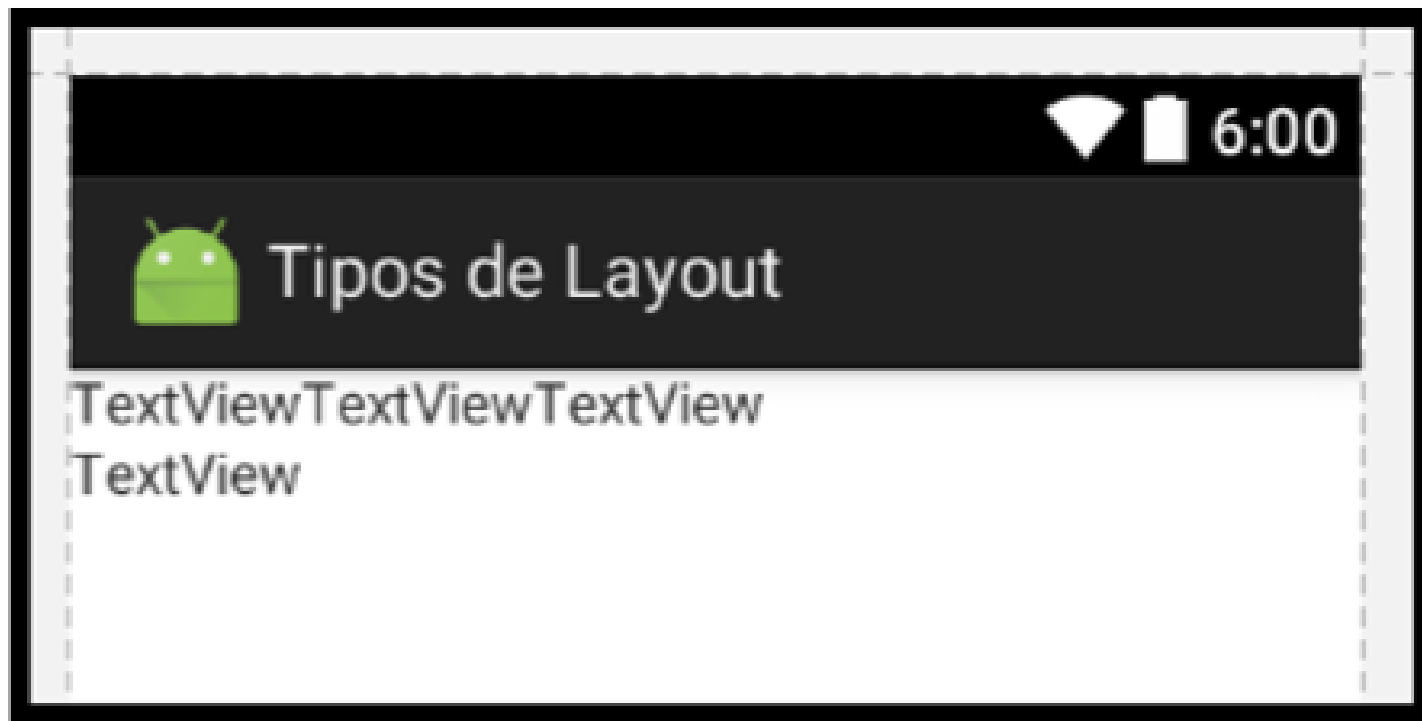
Arrastando o TextView



Estruturas de layout do Android



- Feito isso teremos o nosso resultado:



Componente inserido

Estruturas de layout do Android

- Vamos ver agora como ficou o código em **XML** da tela da aplicação, conforme você pode conferir abaixo e nos próximos slides:

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <TableRow
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">
```

Estruturas de layout do Android

#Continuação do XML da nossa aplicação.

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="New Text"
    android:id="@+id/textView2"
    android:layout_column="0" />
```

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="New Text"
    android:id="@+id/textView3"
    android:layout_column="1" />
```

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="New Text"
    android:id="@+id/textView4"
    android:layout_column="2" />
```

```
</TableRow>
```

Estruturas de layout do Android

#Continuação do XML da nossa aplicação.

```
<TableRow
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Text"
        android:id="@+id/textView5"
        android:layout_column="0" />

</TableRow>

</TableLayout>
```



LinearLayout



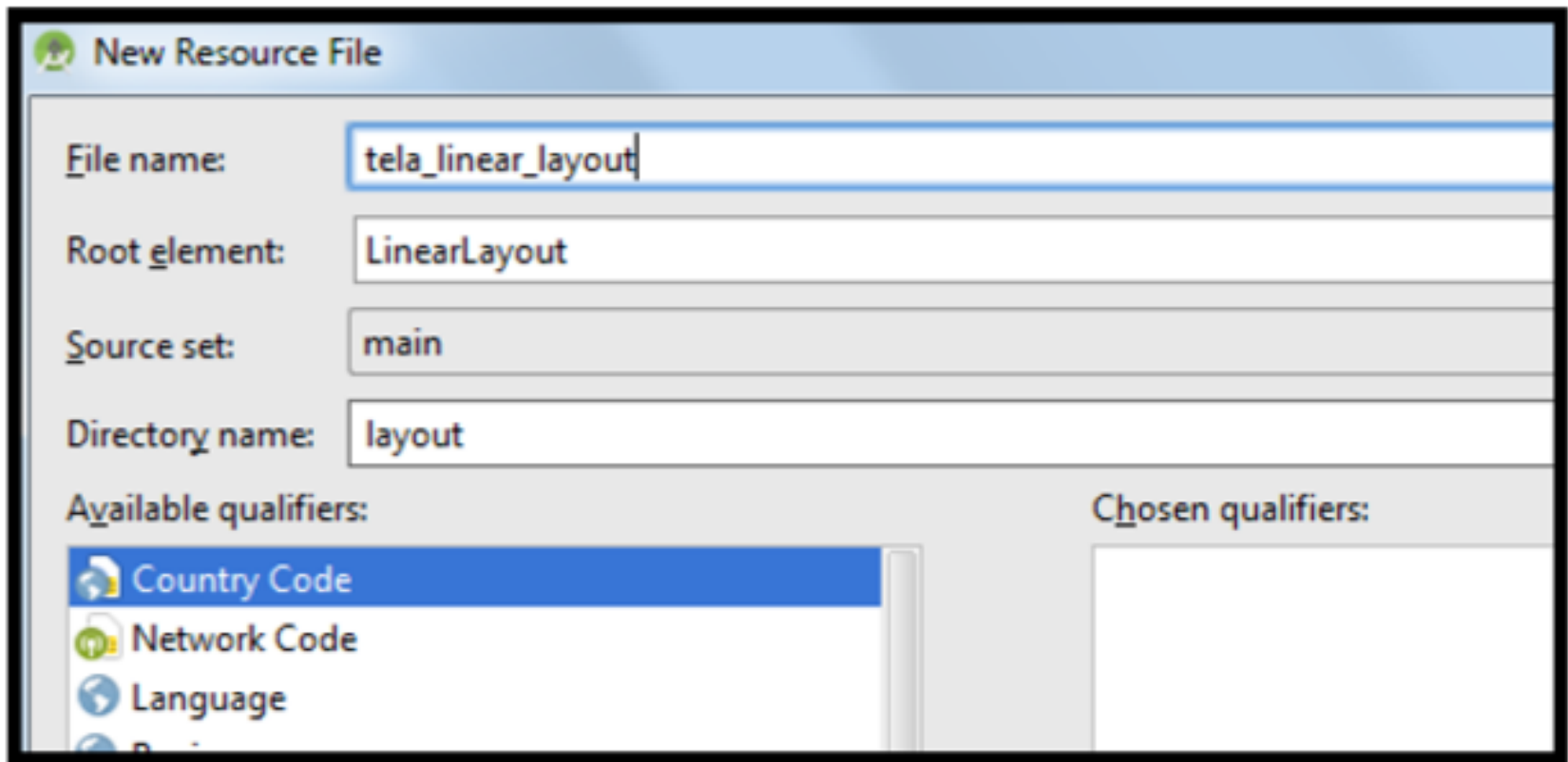
Estruturas de layout do Android



- **LinearLayout**
 - A estrutura **LinearLayout** organiza os componentes de forma que eles possam ser distribuídos de forma horizontal e vertical (ele foi uma estrutura padrão de distribuição de componentes nas primeiras versões do **Android Developer Tools**). Vamos criar agora um arquivo de layout chamado **“tela_linear_layout”**, através do **“Layout resource file”**. Veja abaixo como ficará as configurações do nosso arquivo a seguir.
-

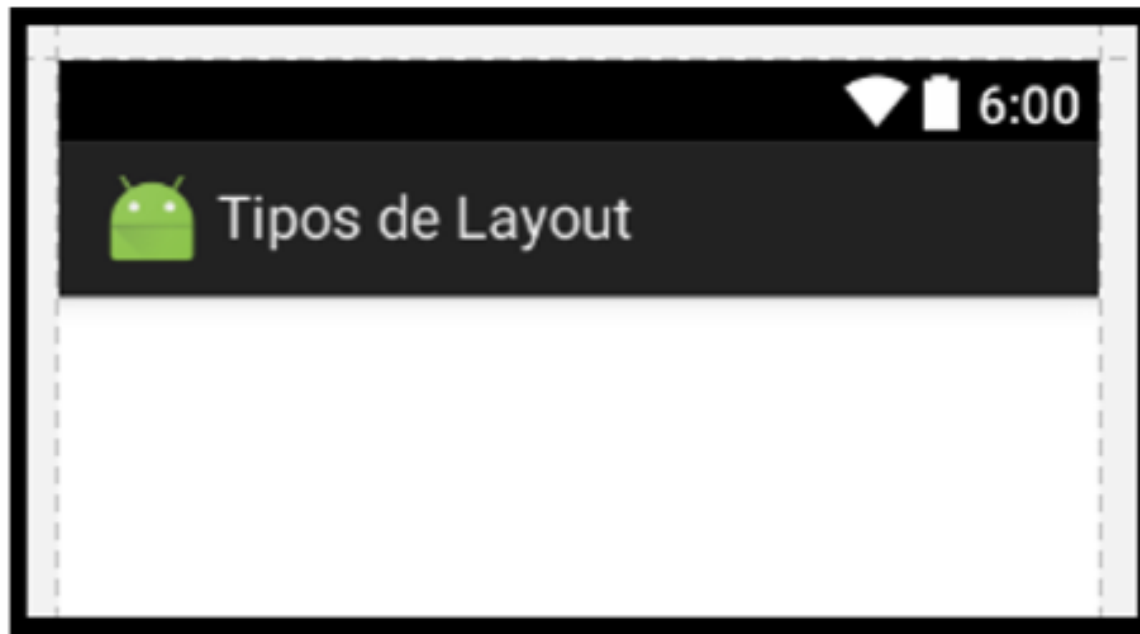
Estruturas de layout do Android

- Configurando **File name** e **Root element**:



Estruturas de layout do Android

- Clique em “**OK**” para que o arquivo possa ser criado. Vejamos a tela criada abaixo:



Tela criada

Estruturas de layout do Android

- Vamos adicionar um componente do **TextView** e em seguida, modifique a seguinte propriedade abaixo:

TextView

Propriedade	Valor
text	Texto 1

- Agora adicione mais dois componentes **TextView** na sequência, conforme as tabelas a apresentadas no próximo slide.
-



Estruturas de layout do Android



- Os demais componentes **TextView** da nossa aplicação:

TextView

Propriedade	Valor
text	Texto 2

TextView

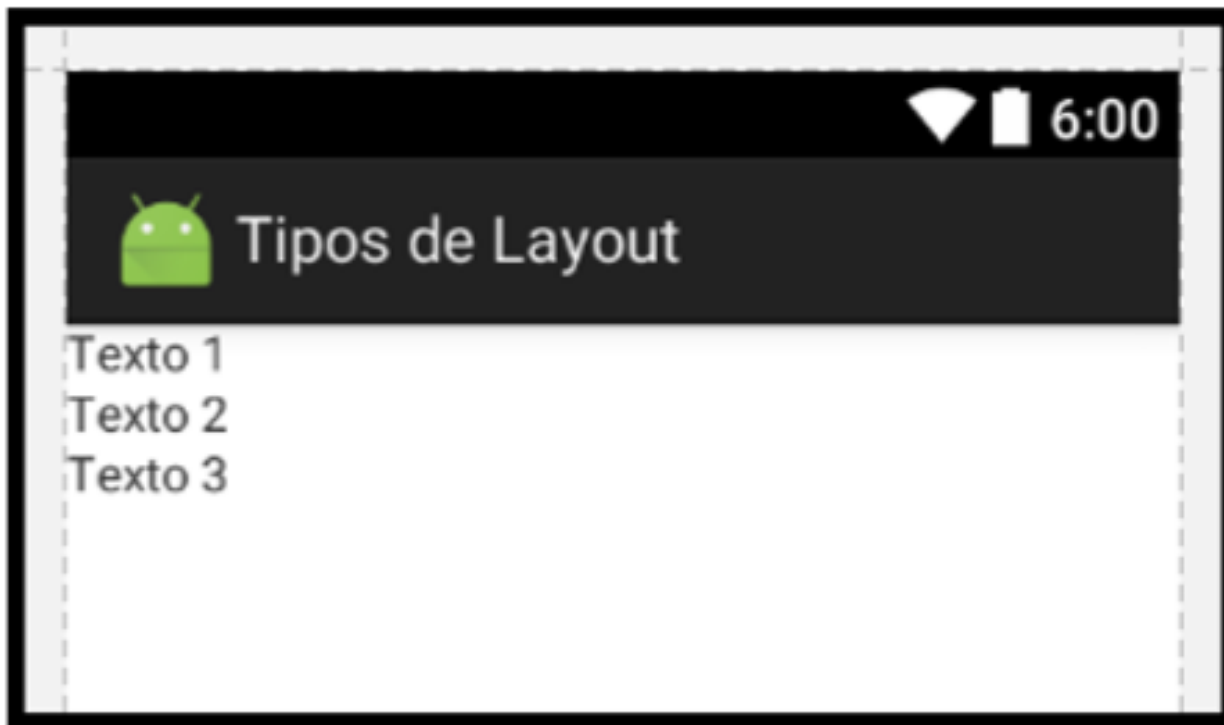
Propriedade	Valor
text	Texto 3



Estruturas de layout do Android



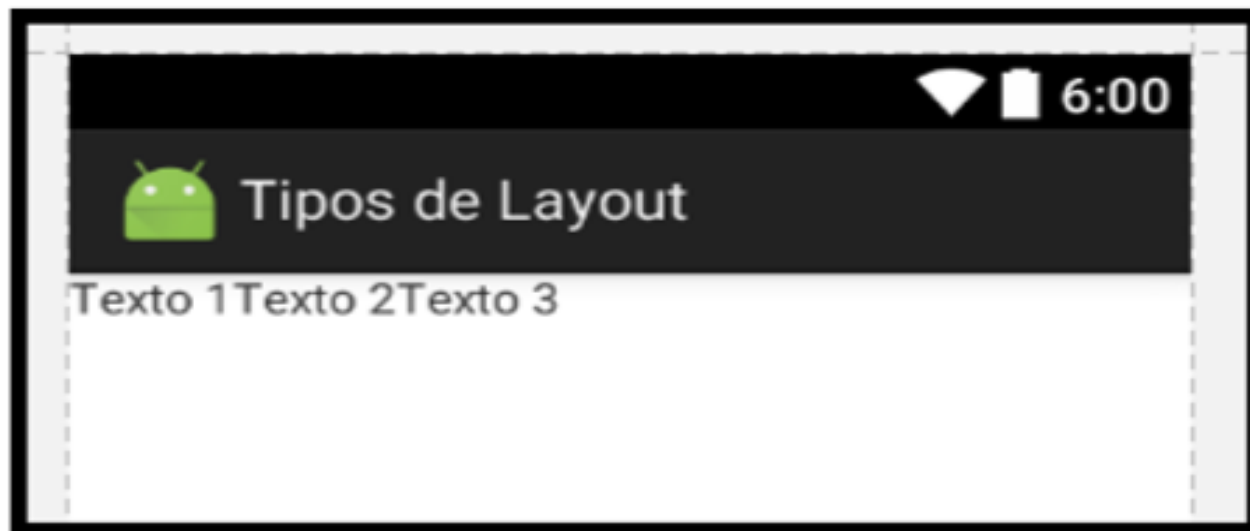
- Vejamos o resultado:



Componentes distribuídos com o LinearLayout (Vertical)

Estruturas de layout do Android

- A estrutura **LinearLayout** que definimos acima faz a distribuição vertical dos componentes (um embaixo do outro). Vamos agora na propriedade “**orientation**” e troque de “**vertical**” para “**horizontal**” (distribuição um ao lado do outro) e confira o resultado:



Componentes distribuídos com o **LinearLayout** (Horizontal)



ConstraintLayout



Estruturas de layout do Android



Como usar o Constraint Layout no Android?

Para conhecer sobre o Constraint Layout recomendo a leitura do artigo da Imaster disponível no link abaixo:

<https://imasters.com.br/android/como-usar-o-constraint-layout-no-android>



Exercício - vamos desenvolver????



Aplicação simples de compras na versão 3 do Android Studio

The screenshot shows a mobile application interface titled "Sistema de Compras". It features a list of products with checkboxes for selection. The products are Arroz (R\$ 2,69), Leite (R\$ 5,00), Carne (R\$ 10,90), and Feijão (R\$ 2,30). At the bottom, there is a button labeled "TOTAL DAS COMPRAS".

Sistema de Compras

Escolha o seu produto:

- ☐ Arroz (R\$ 2,69)
- ☐ Leite (R\$ 5,00)
- ☐ Carne (R\$ 10,90)
- ☐ Feijão (R\$ 2,30)

TOTAL DAS COMPRAS

APP simples de compras

- Agora para aprimorarmos os nossos conhecimentos no desenvolvimento de aplicações para Android, vamos criar um outro aplicativo que consiste em um sistema de compras, bem simples, onde enfatizaremos aqui o uso do componente **CheckBox**.
 - Em nossa aplicação teremos disponíveis **quatro** produtos: Arroz (R\$ 2,69) , Leite (R\$ 5,00) , Carne (R\$ 10,90) e Feijão (R\$ 2,30).
 - Nessa aplicação o usuário marca os itens que quer comprar e no final o sistema mostra o valor total das compras.
-

APP simples de compras

- Na aplicação que iremos desenvolver vamos utilizar os seguintes widgets: **TextView**, **CheckBox** e **Button**.
- Vamos criar um novo **projeto** no Android Studio chamado “**Sistema de Compras**”. Siga os dados do projeto abaixo:

Application Name : Sistema de Compras

Company Domain : app.usuario

Project location : (Fica ao seu critério onde salvar)

APP simples de compras

- Outras configurações do Projeto:

Name
Sistema de Compras

Package name
com.example.sistemadecompras

Save location
C:\ProjetosMobile\SistemadeCompras

Language
Java

Minimum SDK
API 15: Android 4.0.3 (IceCreamSandwich)

i Your app will run on approximately **100%** of devices.
[Help me choose](#)

☐ Use legacy android.support libraries [?](#)

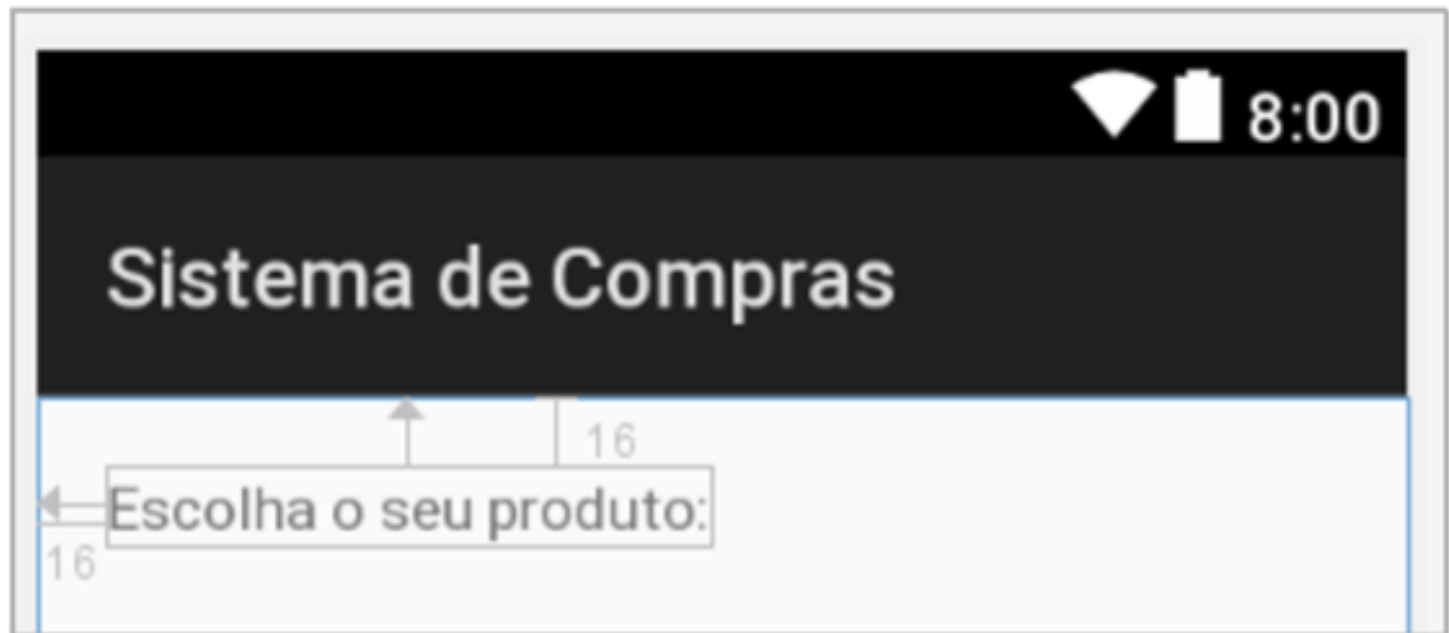
APP simples de compras

- Depois de carregado e criado o projeto modifique o componente **TextView** situado na tela, de acordo com a tabela abaixo:

Propriedade	Valor
text	Escolha o seu produto:

APP simples de compras

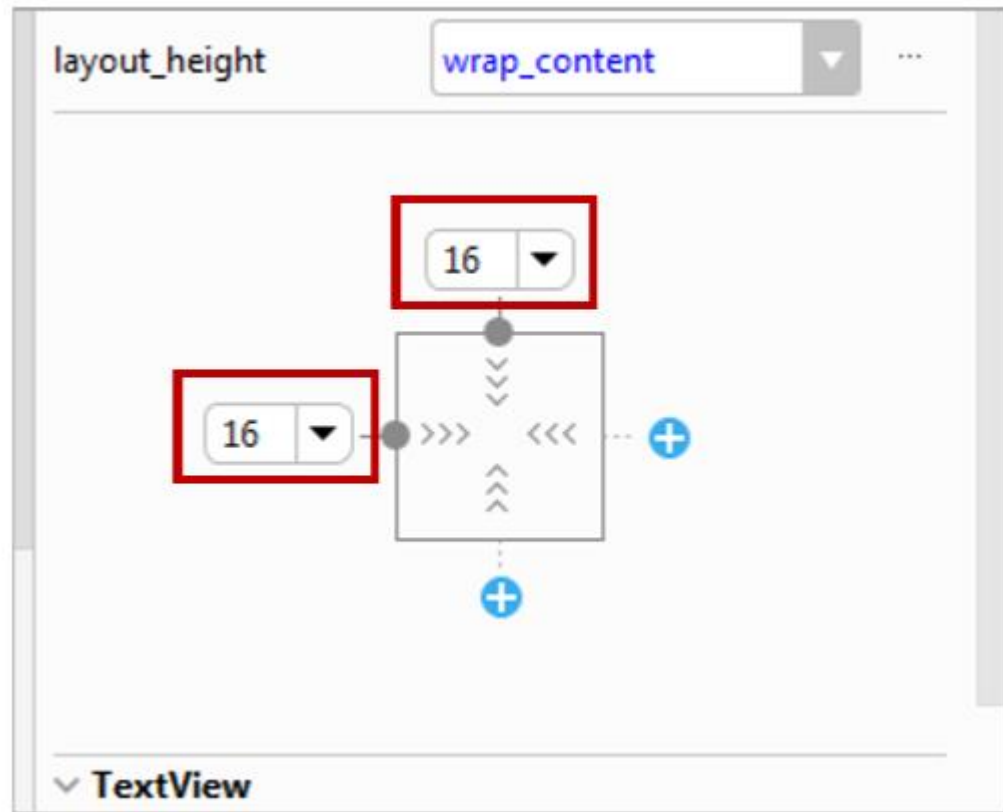
- Após efetuar a modificação solicitada, crie as conexões (influências) no topo e para o lado esquerdo da tela. Vejamos o resultado:



Componente TextView inserido

APP simples de compras

- **UMA DICA:** Para um melhor controle da distância das conexões criadas, use as propriedades do componente, conforme é mostrado abaixo:



APP simples de compras

- Vamos agora adicionar quatro componentes do tipo **CheckBox** (na seção “**Buttons**”), realizando as seguintes modificações conforme a tabela abaixo, na sequência:

CheckBox

Propriedade	Valor
id	chkarroz
text	Arroz (R\$ 2,69)

CheckBox

Propriedade	Valor
id	chkleite
text	Leite (R\$ 5,00)

APP simples de compras

CheckBox

Propriedade	Valor
id	chkcarne
text	Carne (R\$ 10,90)

CheckBox

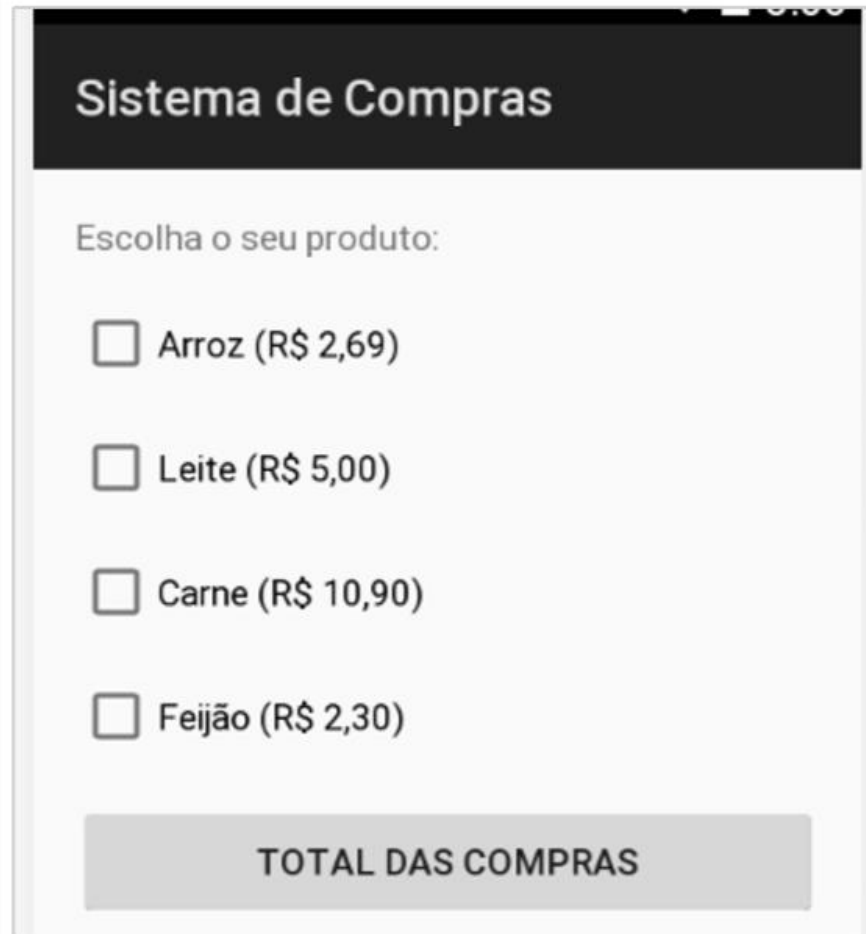
Propriedade	Valor
id	chkfeijao
text	Feijão (R\$ 2,30)

Button

Propriedade	Valor
id	bttotal
text	Total das compras
layout_width	0dp (match_constraint)

APP simples de compras

- Ao final, o **layout** da nossa aplicação deve estar de acordo com a figura seguinte:



The image shows a mobile app interface for a shopping system. It features a dark header with the title 'Sistema de Compras'. Below the header, there is a label 'Escolha o seu produto:' followed by four list items, each with an unchecked checkbox and a product name with its price in parentheses: 'Arroz (R\$ 2,69)', 'Leite (R\$ 5,00)', 'Carne (R\$ 10,90)', and 'Feijão (R\$ 2,30)'. At the bottom of the interface is a large, light gray button labeled 'TOTAL DAS COMPRAS'.

Sistema de Compras

Escolha o seu produto:

☐ Arroz (R\$ 2,69)

☐ Leite (R\$ 5,00)

☐ Carne (R\$ 10,90)

☐ Feijão (R\$ 2,30)

TOTAL DAS COMPRAS

APP simples de compras

- Segue o XML do APP de Compras:

```
1 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
2   xmlns:tools="http://schemas.android.com/tools"
3   android:layout_width="match_parent"
4   android:layout_height="match_parent"
5   tools:context=".MainActivity">
6   <TextView android:text="@string/escolha_o_seu_produto"
7       android:layout_width="wrap_content"
8       android:layout_height="wrap_content"
9       android:id="@+id/textView" />
10  <CheckBox
11      android:layout_width="wrap_content"
12      android:layout_height="wrap_content"
13      android:text="@string/arroz_r_2_69"
14      android:id="@+id/chkarroz"
15      android:layout_below="@+id/textView"
16      android:layout_alignParentLeft="true"
17      android:layout_alignParentStart="true" />
```

APP simples de compras

- Continuação do XML do APP de Compras:

```
18 <CheckBox
19     android:layout_width="wrap_content"
20     android:layout_height="wrap_content"
21     android:text="@string/leite_r_5_00"
22     android:id="@+id/chkleite"
23     android:layout_below="@+id/chkarroz"
24     android:layout_alignParentLeft="true"
25     android:layout_alignParentStart="true" />
26
27 <CheckBox
28     android:layout_width="wrap_content"
29     android:layout_height="wrap_content"
30     android:text="@string/carne_r_9_70"
31     android:id="@+id/chkcarne"
32     android:layout_below="@+id/chkleite"
33     android:layout_alignParentLeft="true"
34     android:layout_alignParentStart="true" />
35
```

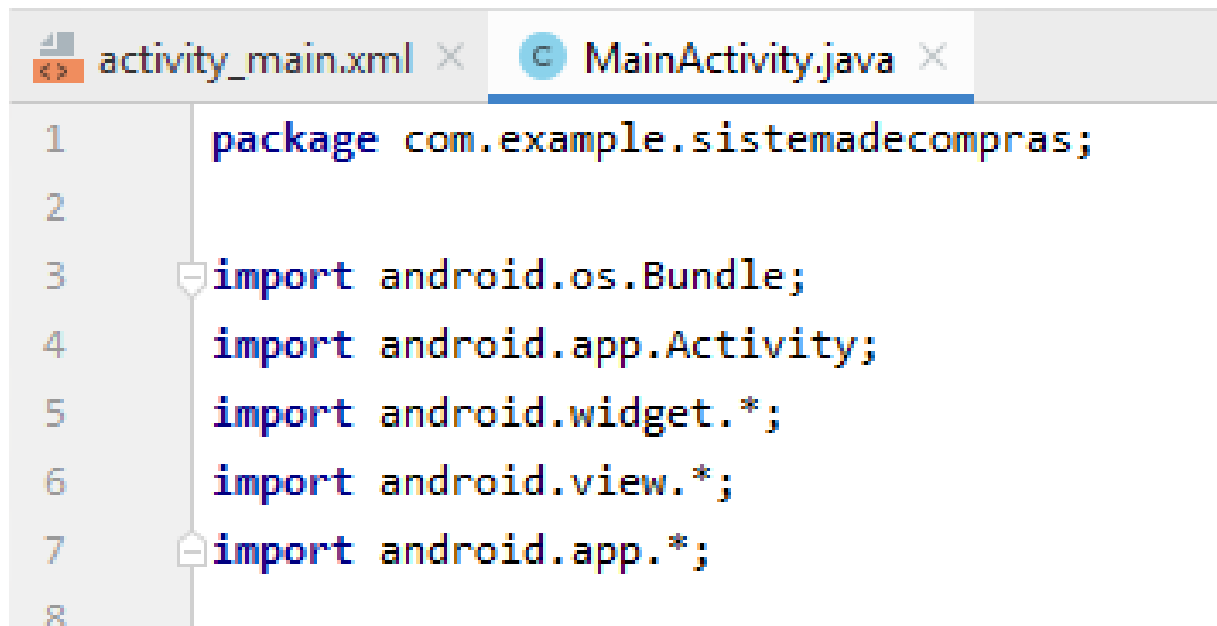
APP simples de compras

- Continuação do XML do APP de Compras:

```
36      <CheckBox
37          android:layout_width="wrap_content"
38          android:layout_height="wrap_content"
39          android:text="@string/feij_o_r_2_70"
40          android:id="@+id/chkfeijao"
41          android:layout_below="@+id/chkcarne"
42          android:layout_alignParentLeft="true"
43          android:layout_alignParentStart="true" />
44      <Button
45          android:layout_width="match_parent"
46          android:layout_height="wrap_content"
47          android:text="@string/total_das_compras"
48          android:id="@+id/btttotal"
49          android:layout_below="@+id/chkfeijao"
50          android:layout_alignParentLeft="true"
51          android:layout_alignParentStart="true" />
52  </RelativeLayout>
53
```

APP simples de compras

- Vamos abrir agora o arquivo “MainActivity.java” para iniciarmos a sua codificação.
- Primeiramente vamos realizar a importação dos seguintes pacotes conforme mostra a figura abaixo:



The screenshot shows an IDE window with two tabs: 'activity_main.xml' and 'MainActivity.java'. The 'MainActivity.java' tab is active, displaying the following Java code:

```
1 package com.example.sistemadecompras;
2
3 import android.os.Bundle;
4 import android.app.Activity;
5 import android.widget.*;
6 import android.view.*;
7 import android.app.*;
8
```

APP simples de compras

Na seção de declaração dos atributos da classe **ComprasActivity** digite o seguinte código mostrado abaixo:

```
9  public class MainActivity extends Activity {  
10  
11      CheckBox chkarroz, chkleite, chkcarne, chkfeijao;  
12      Button btttotal;  
13  
14      @Override  
15      public void onCreate(Bundle savedInstanceState) {  
16          super.onCreate(savedInstanceState);  
17
```



APP simples de compras

Continuação do código .java:

```
18      setContentView(R.layout.activity_main);
19
20      chkarroz = (CheckBox) findViewById(R.id.chkarroz);
21      chkleite = (CheckBox) findViewById(R.id.chkleite);
22      chkcarne = (CheckBox) findViewById(R.id.chkcarne);
23      chkfeijao = (CheckBox) findViewById(R.id.chkfeijao);
24      Button btttotal = (Button) findViewById(R.id.btttotal);
25
26      btttotal.setOnClickListener(new View.OnClickListener() {
27
```

APP simples de compras

Continuação do código .java:

```
28
29  
30
31
32
33
34
35
36
37
38
39
40

@Override
public void onClick(View arg0) {

    double total = 0;
    if (chkarroz.isChecked())
        total += 2.69;
    if (chkleite.isChecked())
        total += 5.00;
    if (chkcarne.isChecked())
        total += 9.7;
    if (chkfeijao.isChecked())
        total += 2.30;
```


APP simples de compras

Continuação do código .java:

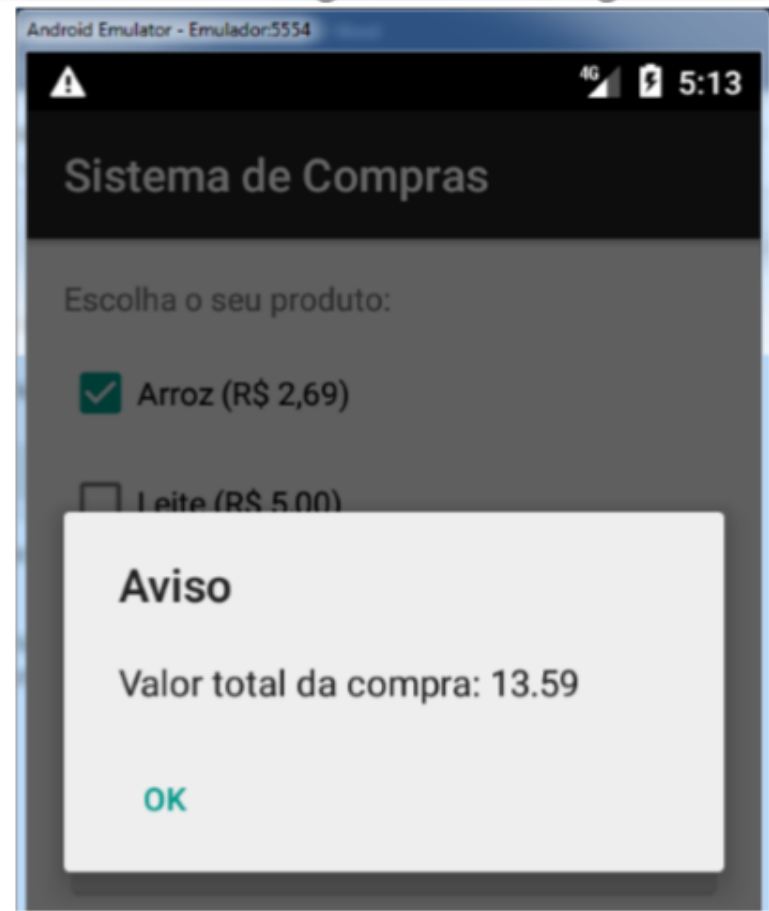
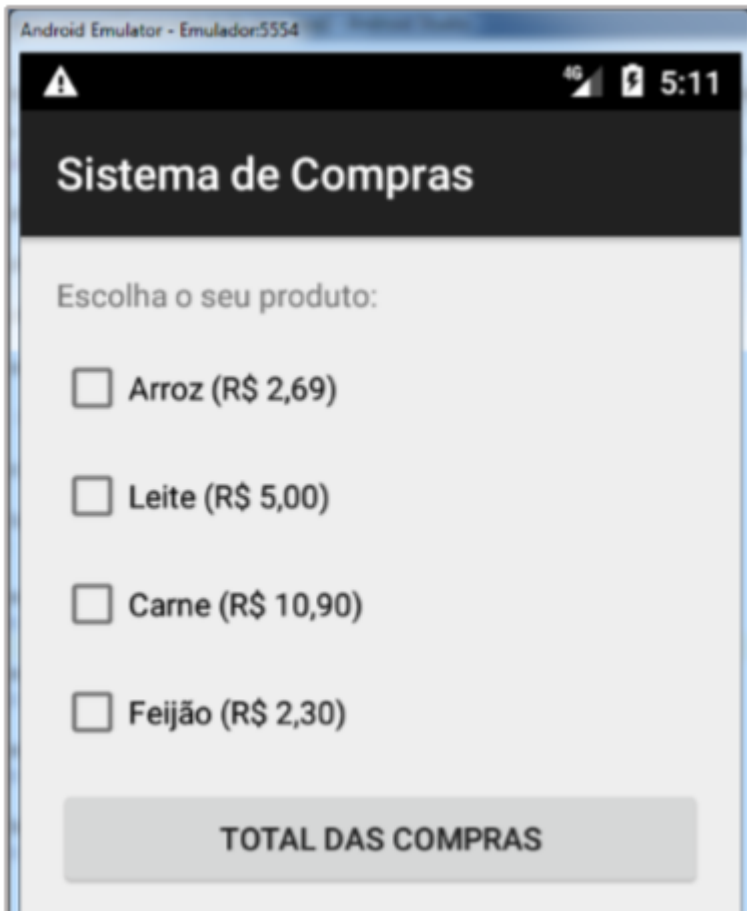
```
41 |
42 |     AlertDialog.Builder dialogo = new AlertDialog.Builder(
43 |         context: MainActivity.this);
44 |     dialogo.setTitle("Aviso");
45 |     dialogo.setMessage("Valor total da compra : " + String.valueOf(total));
46 |     dialogo.setNeutralButton( text: "OK", listener: null);
47 |     dialogo.show();
48 |
49 |     }
50 | });
51 |
52 | }
53 | }
```

APP simples de compras

- Uma pequena descrição sobre o código .java:
 - No método **onClick** é criado uma variável chamada **total** que armazena o valor total da compra. Observem que existem quatro estruturas **if's** onde cada uma verifica se um determinado item foi marcado, se foi, incrementa o valor do item na variável **total**. No final é exibido o valor total das compras na tela.
-

APP simples de compras

- Vamos rodar nossa aplicação? O resultado você confere na figura seguinte:





Por hoje é só!!!

Até a próxima aula...
