

Linguagem e Técnica de Programação Mobile

AULA 4 – Conhecendo as widgets do Android

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



Roteiro da Aula



- Conhecendo as widgets do Android
 - A seção “Common”
 - A seção “Text”
 - A seção “Buttons”
 - A seção “Widgets”
 - A seção “Layouts”
 - A seção “Containers”
 - A seção “Google”
 - A seção “Legacy”
 - Construindo nossas aplicações no Android
 - Desenvolvendo uma Calculadora Básica
-



Conhecendo as widgets do Android



Conhecendo as widgets do Android



- Toda aplicação Android normalmente é formado por um ou mais **widgets**, que são componentes gráficos que constituem uma aplicação.
 - A partir de agora iremos conhecer os widgets básicos disponíveis e oferecidos pela plataforma Android, para o desenvolvimento das aplicações.
 - De acordo com alguns widgets que fomos conhecendo, vamos desenvolver aplicações que demonstrem o uso deles.
-



Conhecendo as widgets do Android



- Na aula passada (**Aula 3 - Começando a programar no Google Android**) aprendemos a desenvolver nossa primeira aplicação em Android, algo bem simples, mas já desenvolvemos.
 - Agora vamos conhecer melhor a **paleta** de componentes onde nela estão disponíveis todos os **widgets** que podemos utilizar em uma aplicação.
-



Conhecendo as widgets do Android



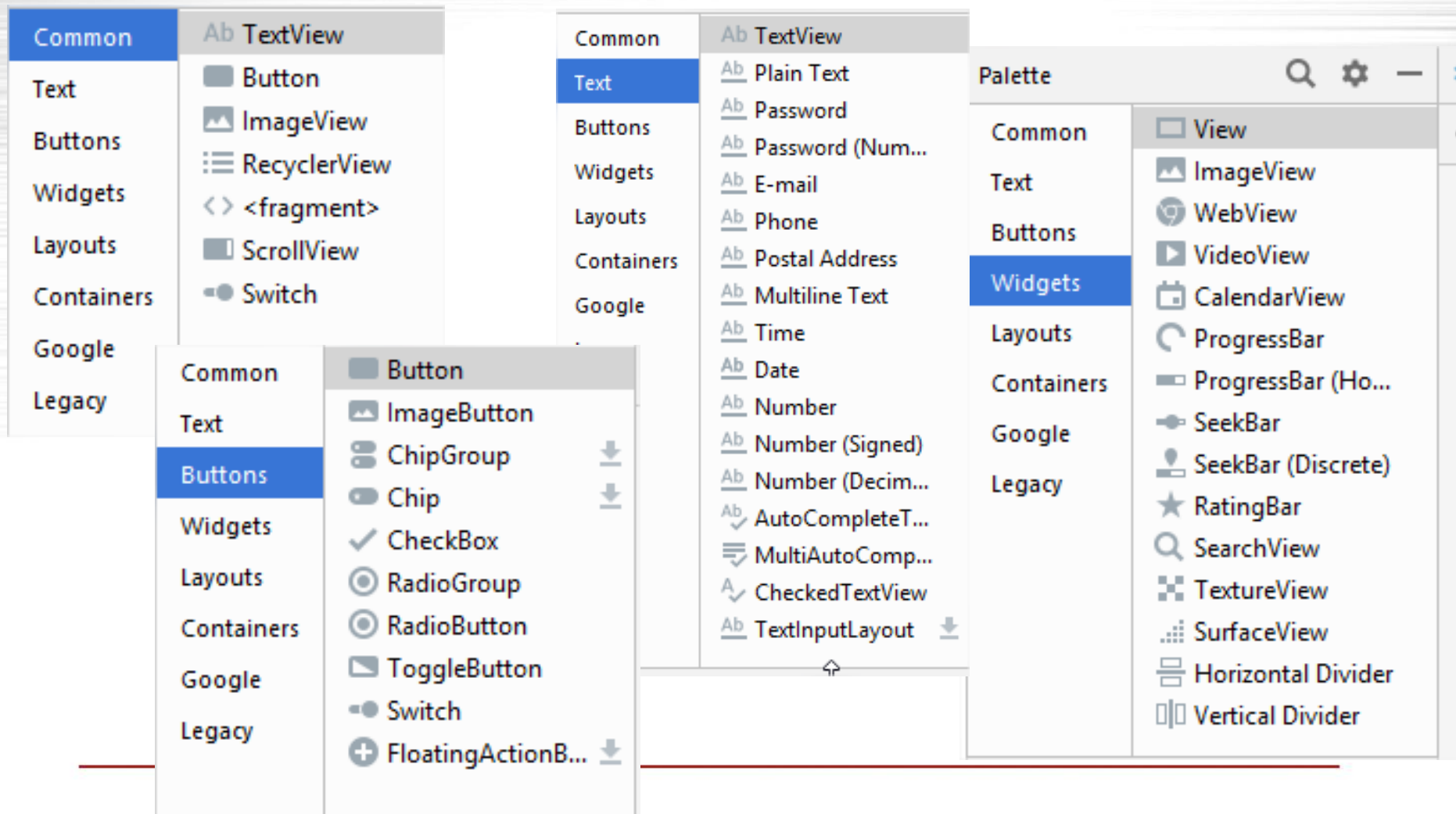
- A paleta de componentes e suas widgets
 - A ferramenta de desenvolvimento do **Android SDK** nos oferece uma gama de componentes (ou widgets, como preferirem) que podemos utilizar em uma aplicação a ser construída.
 - Podemos conferir esses widgets na paleta mostrada pelas figuras nos próximos slides.
-



Conhecendo as widgets do Android



- As paletas de Componentes Common, Buttons ... :

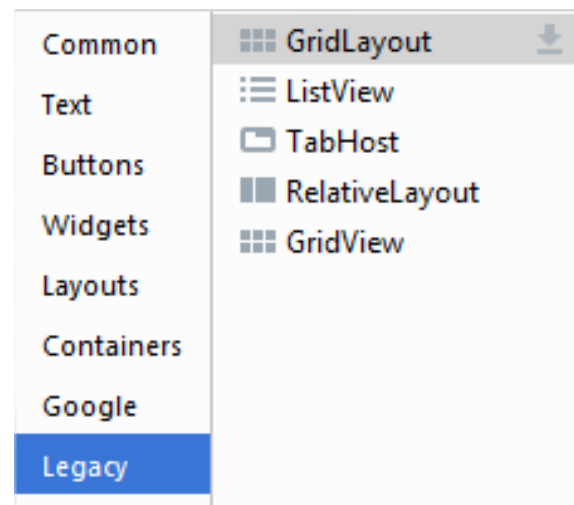
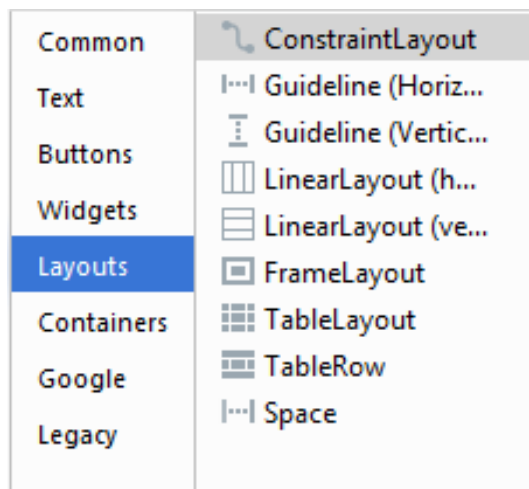




Conhecendo as widgets do Android



- Os componentes estão distribuídos nas mais diversas seções presentes na paleta de componentes.
- Vamos conhecer algumas das seções desta paleta e os componentes nela presentes.







Conhecendo as widgets do Android




- Vejamos alguns componentes mais básicos que podem ser utilizados em uma aplicação **Android**, são eles:

 TextView


Ícones do componente TextView

 Spinner


Ícone do componente Spinner

 Button

Ícones do componente Button

 CheckBox


Ícone do componente CheckBox

 RadioButton

Ícone do componente RadioButton

 ProgressBar

Ícone do componente ProgressBar

 RatingBar

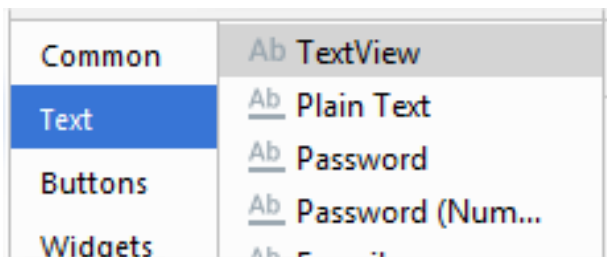
Ícone do componente RatingBar



Conhecendo as widgets do Android



- Vamos conhecer cada um deles?
- **TextView** : componente que funciona como se fosse uma **Label** (“rotulo”), onde nele podemos mostrar alguma informação, mensagem e etc.
- Na nossa primeira aplicação tivemos a oportunidade de usarmos esse componente. Veja abaixo o ícone dele:



Ícones do componente TextView

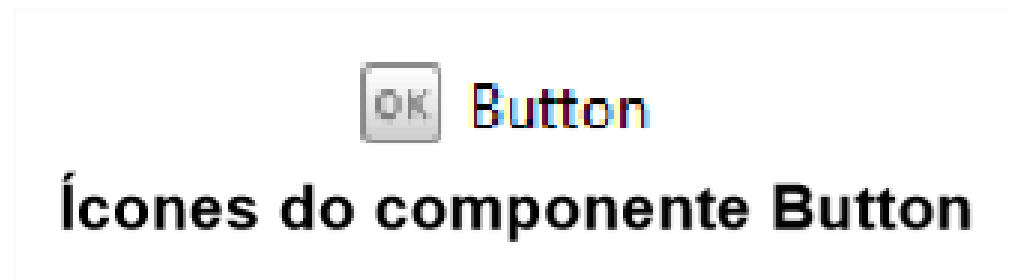
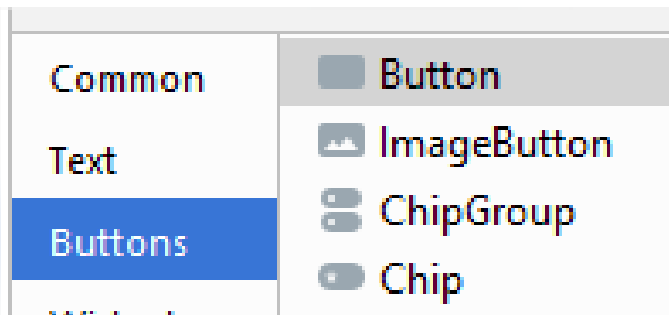




Conhecendo as widgets do Android



- **Button** : componente que representa um **botão** onde podemos clicar nele e também atribuir ações que podem ser executadas caso isso aconteça. Veja seu ícone na paleta de componentes:

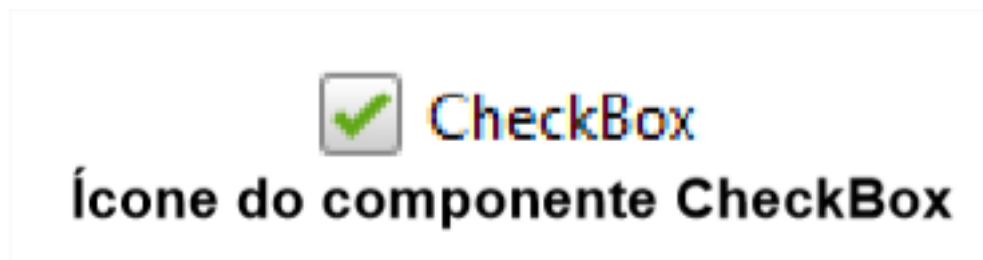
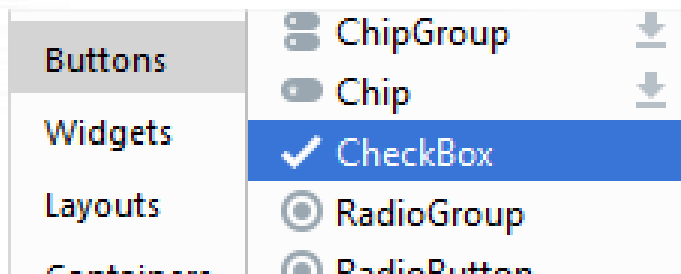




Conhecendo as widgets do Android



- **CheckBox** : esse componente funciona como **uma opção**, onde nele podemos marcá-lo e desmarcá-lo. Veja o ícone do componente abaixo:





Conhecendo as widgets do Android



- **RadioButton** : esse componente funciona como uma opção, normalmente utilizado quando temos uma situação onde devemos escolher uma entre várias opções (como numa prova de múltipla escolha), vamos precisar de um **RadioGroup**. Veja o ícone desse componente na paleta de componentes :

 RadioGroup

 RadioButton



RadioButton

Ícone do componente **RadioButton**

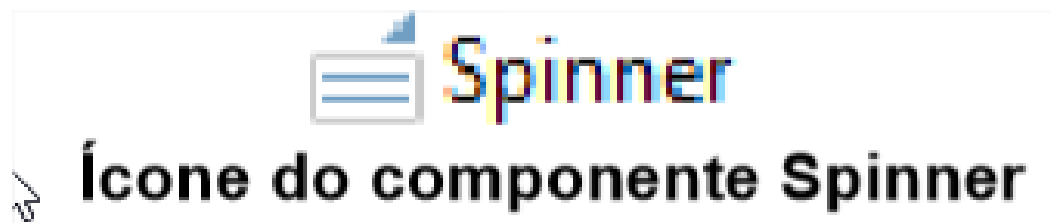
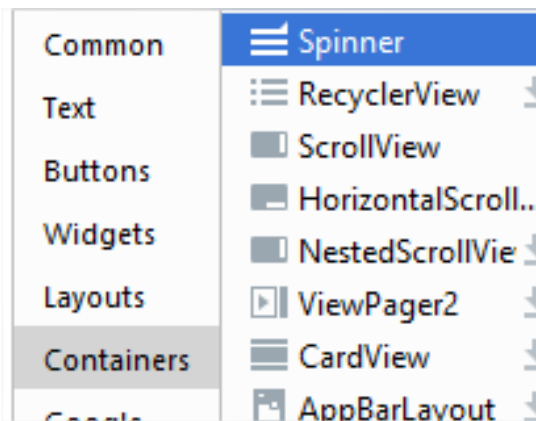




Conhecendo as widgets do Android



- **Spinner** : esse componente nada mais é do que uma caixa de combinação (também conhecido como **Combo Box**). Nesse componente podemos adicionar vários itens que poderão ser selecionados pelo usuário através do mesmo. Veja o ícone desse componente na figura seguinte:





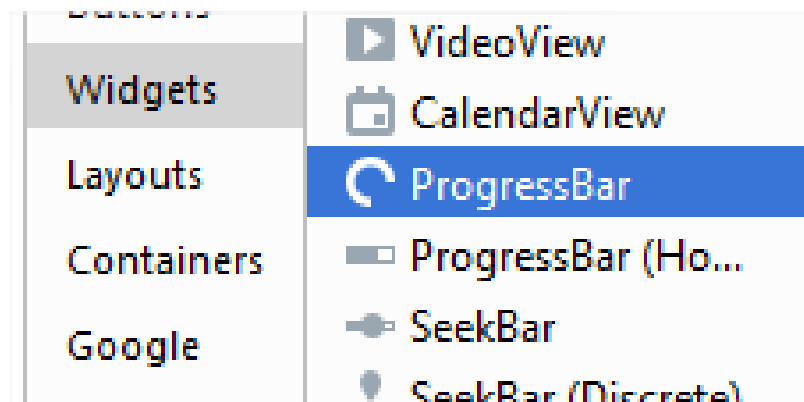
Conhecendo as widgets do Android



- **ProgressBar** : esse componente exibe uma barra de progresso na tela. Veja seus ícones abaixo:



Ícone do componente **ProgressBar**

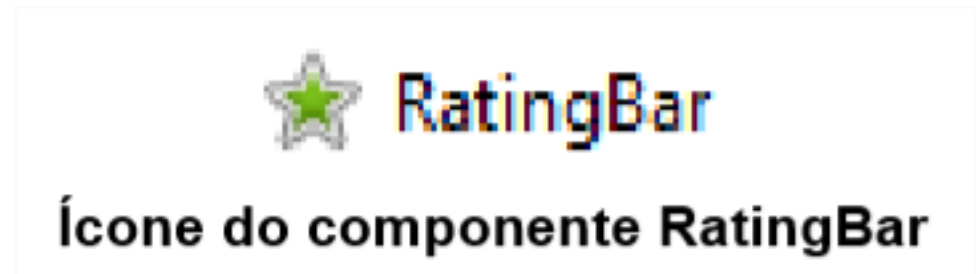
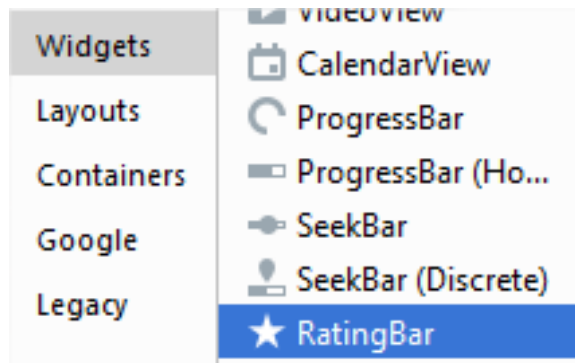




Conhecendo as widgets do Android



- **RatingBar** : esse componente é bastante utilizado para fazer sistemas de votações e classificações (aqueles sistemas em que você define se uma coisa é **ruim**, **regular**, **boa**, **ótima** e etc). Veja o ícone do componente na figura seguinte:

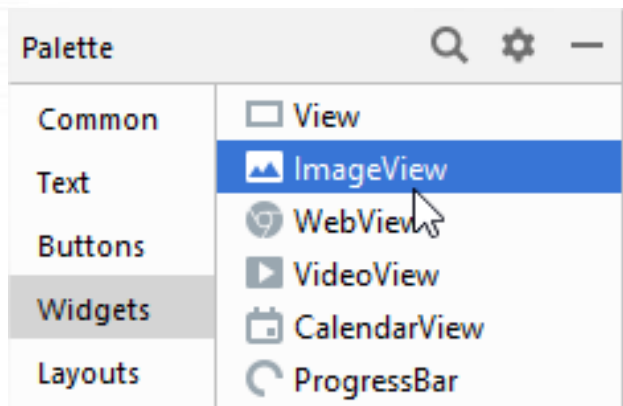




Conhecendo as widgets do Android



- **ImageView** : esse componente simplesmente serve para exibir imagens que se colocam nele. Os formatos de imagens suportados por esse componente são : **PNG, JPEG, BMP, GIF**. Veja o ícone desse componente em seguida:

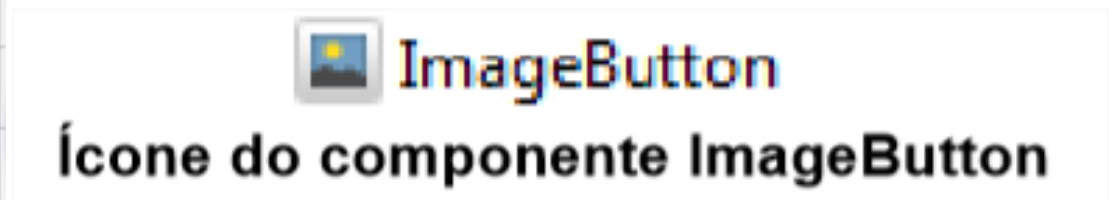
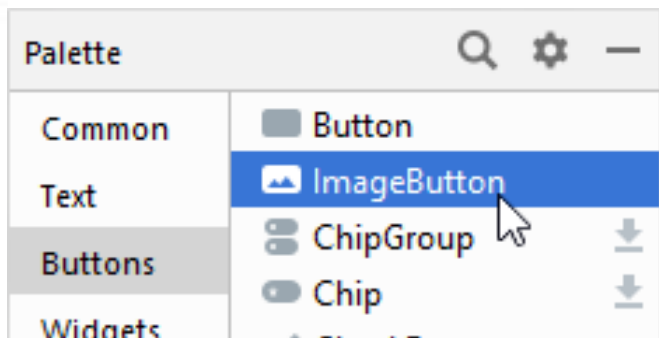




Conhecendo as widgets do Android



- **ImageButton** : esse componente é derivado do componente **Button** só que ao invés de exibir um texto dentro dele, exibe uma imagem. Os formatos de imagens suportados por esse componente são : **PNG, JPEG, BMP, GIF**. Veja o ícone desse componente abaixo:

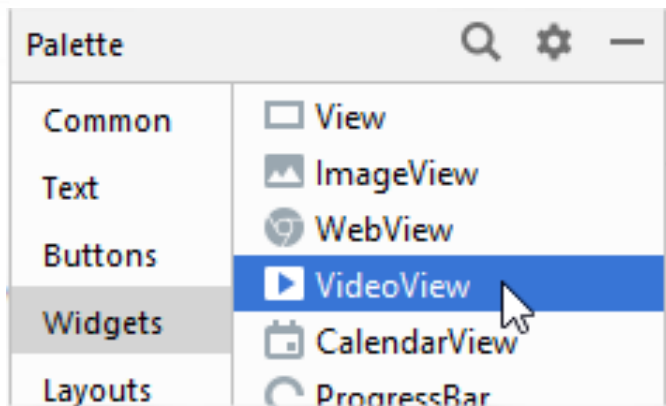




Conhecendo as widgets do Android



- **VideoView** : esse componente é destinado a exibição e reprodução de vídeos nos formatos mais comuns (**mp4**, **3gp** e **etc**). Esse componente também é capaz de reproduzir áudios (nos formatos **mp3**, **ogg**, **midi** e **etc**). Veja o ícone desse componente abaixo:






Conhecendo as widgets do Android




- A seção “Text”
- Nesta seção estão disponíveis todos os componentes baseados em **caixas de texto**, e todos eles são baseados no componente **EditText**. Vamos ver alguns desses componentes abaixo:

 Plain Text

Ícone do componente Plain Text

 E-mail

Ícone do componente E-mail

 Phone

Ícone do componente Phone

 Password  Password (Numeric)

Ícones do componente Password (na versão normal e numérica)

 Multiline Text

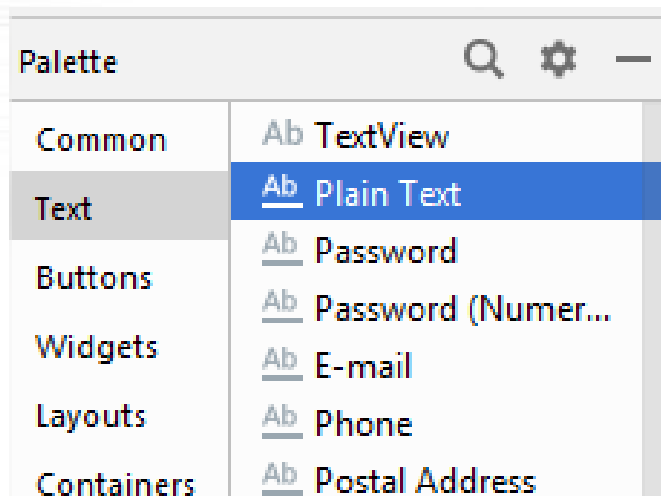
Ícone do componente Multiline Text



Conhecendo as widgets do Android



- **Plain Text:** esse é o modelo de caixa de texto “**padrão**”, que permite a digitação de qualquer tipo de caractere. Veja seu ícone abaixo:



Plain Text

Ícone do componente Plain Text



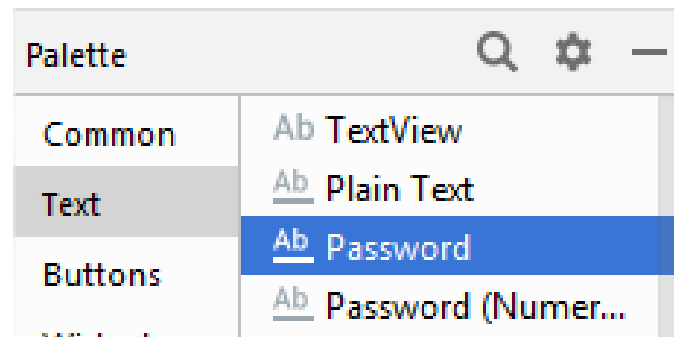
Conhecendo as widgets do Android



- **Password:** esse modelo de caixa de texto permite a digitação de senhas e está disponível tanto na versão **alfanumérica** quanto na **numérica (Numeric)**. Veja o ícone do componente abaixo:



Ícones do componente Password (na versão normal e numérica)

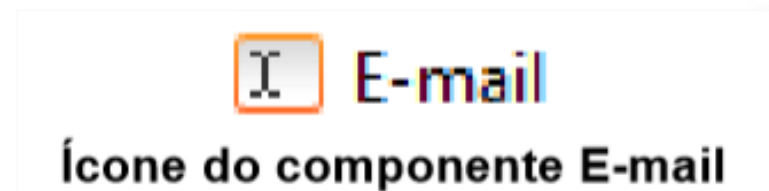
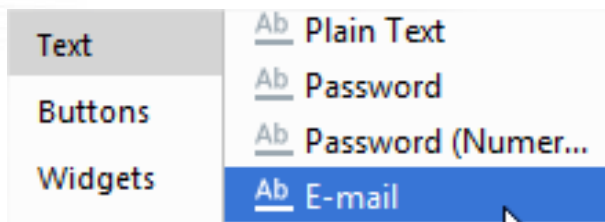




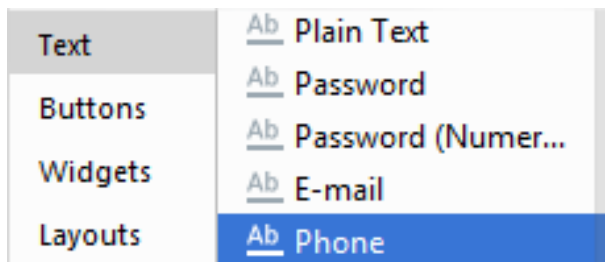
Conhecendo as widgets do Android



- **E-mail:** esse modelo de caixa de texto permite a digitação de e-mail. Veja o ícone do componente abaixo:



- **Phone:** esse modelo de caixa de texto permite a digitação de telefones. Veja o ícone do componente abaixo:

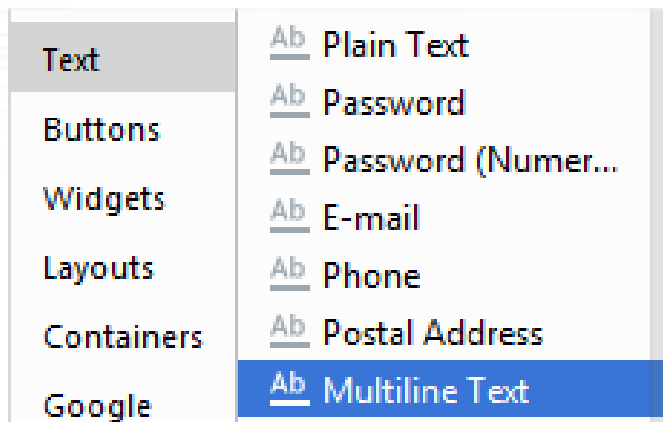




Conhecendo as widgets do Android



- **Multiline Text** : esse modelo de caixa de texto permite várias linhas de texto, de acordo com a nossa necessidade. Veja o ícone desse componente abaixo:



Multiline Text

Ícone do componente Multiline Text



Conhecendo as widgets do Android



- A seção “Layouts”
- Nesta seção estão disponíveis estruturas de layouts que podemos utilizar em nossas aplicações para organizar a disponibilidade dos componentes dentro da tela, no dispositivo. Vejamos esses componentes:



LinearLayout (Vertical)



LinearLayout (Horizontal)

Ícones do componente LinearLayout (horizontal e vertical)



RelativeLayout

Ícone do componente RelativeLayout



TableLayout

Ícone do componente TableLayout





Conhecendo as widgets do Android



- **LinearLayout** : essa estrutura organiza os componentes dentro dela de forma que os mesmos sejam distribuídos de forma horizontal (um ao lado do outro) ou vertical (um abaixo do outro), de acordo com a necessidade. Veja os ícones desse componente:

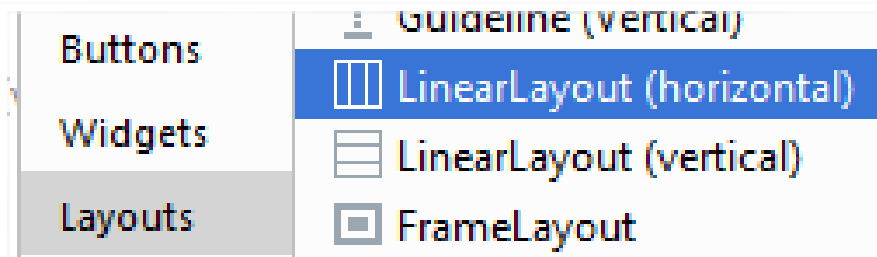


LinearLayout (Vertical)



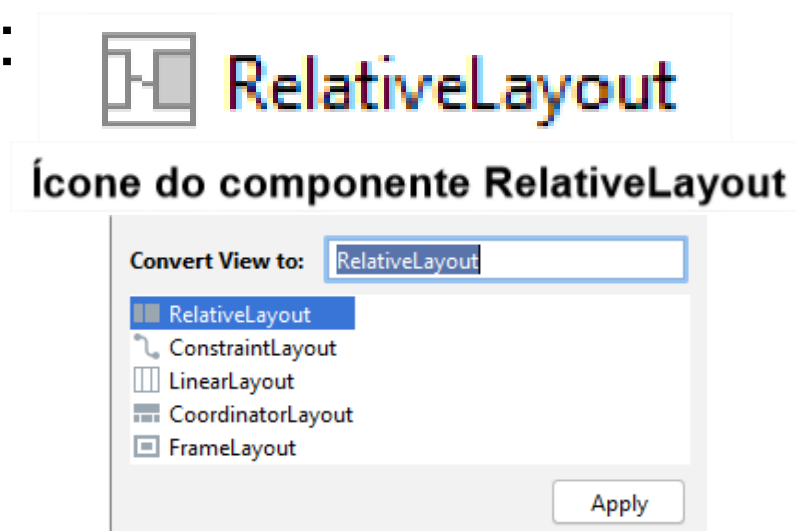
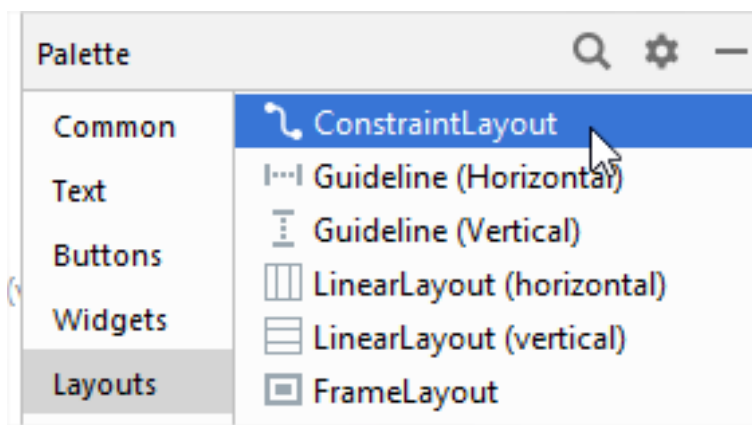
LinearLayout (Horizontal)

Ícones do componente LinearLayout (horizontal e vertical)



Conhecendo as widgets do Android

- **RelativeLayout (ConstraintLayout):** essa estrutura organiza os componentes dentro dela de forma que os mesmos sejam distribuídos livremente na tela (em qualquer ponto em que você desejar, relativo a outros componentes que, possivelmente, estejam na tela). Veja o ícone desse componente:





Conhecendo as widgets do Android

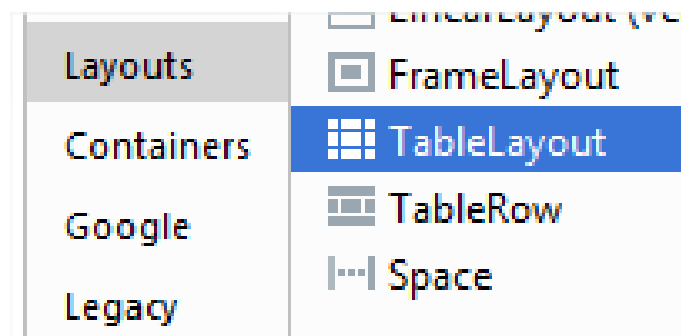


- **TableLayout**: essa estrutura organiza os componentes dentro dela de forma como se estivessem em uma tabela (com o auxílio de um componente útil, o **TableRow**, também presente nesta seção). Veja o ícone desse componente abaixo:



TableLayout

Ícone do componente TableLayout

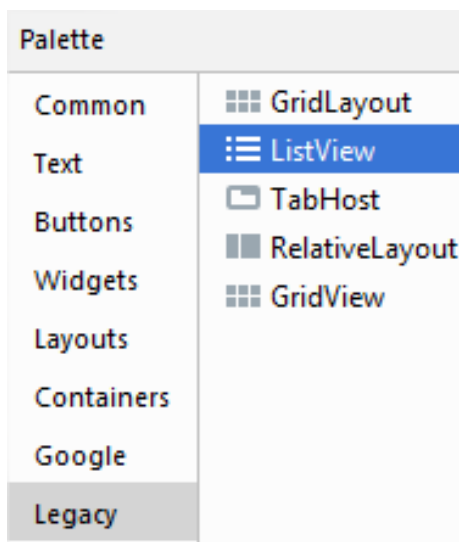




Conhecendo as widgets do Android



- A seção “Legacy”
- Nesta seção estão disponíveis componentes **legados** do Android Studio.
- Vejamos como exemplo o componente **ListView**, e até mesmo o **RelativeLayout**.

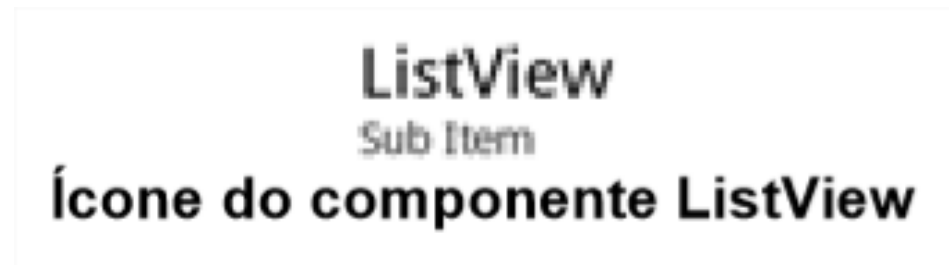
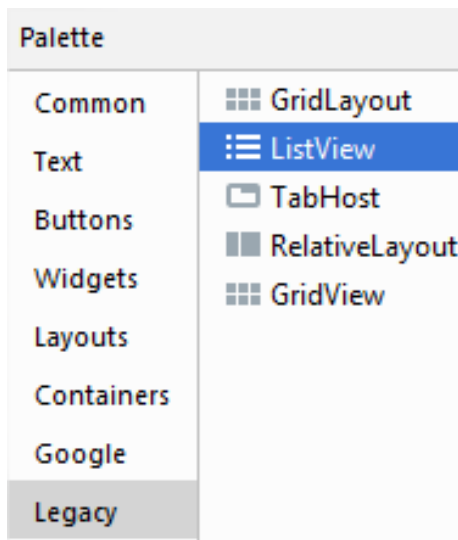




Conhecendo as widgets do Android



- **ListView** : esse componente funciona como uma lista onde nele podemos adicionar itens e visualizar os mesmos (conhecido em algumas ferramentas de desenvolvimento como “**ListBox**”). Vejamos o ícone desse componente:



Exemplo dos componentes

Component Tree

RelativeLayout

- Ab TextView "@string/estou_aprendendo_android"
- Ab textView3 "@string/vou_atualizar_meu_curr_culo"
- Ab textView "TextView"
- button "Button"
- checkbox "CheckBox"
- RadioGroup (vertical)
 - radioButton "RadioButton"
 - radioButton2 "RadioButton"
 - radioButton3 "RadioButton"
 - radioButton4 "RadioButton"
- progressBar (Large)
- ratingBar
- imageView
- imageButton2
- videoView
- Ab editTextTime (Time)

Estou aprendendo Android

☐ CheckBox

Vou atualizar meu currículo!

TextView

BUTTON

- ☐ RadioButton
- ☐ RadioButton
- ☐ RadioButton
- ☐ RadioButton





Vamos desenvolver um APP????



- **Construindo nossas aplicações no Android**
 - Vamos colocar a mão na massa? A partir de agora iremos começar a desenvolver as nossas aplicações no Android Studio utilizando os componentes apresentados durante as aulas.
 - Começaremos com aplicações simples e aos poucos iremos evoluir, criando aplicações mais ricas nas próximas aulas. **Estão prontos????**
-

Desenvolvendo uma Calculadora Básica

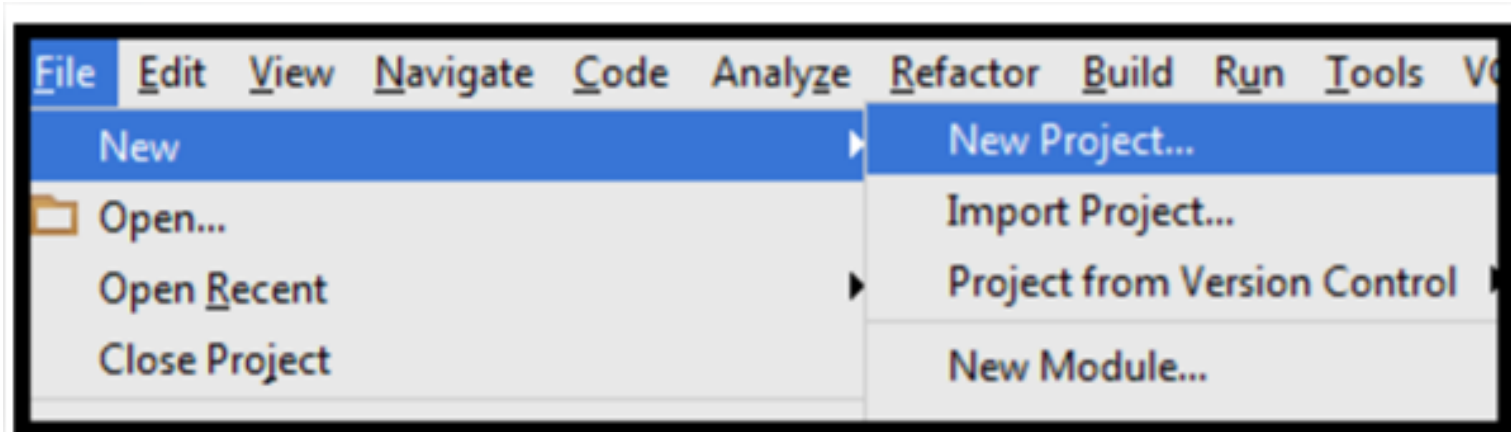




Desenvolvendo uma Calculadora Básica



- Vamos construir a nossa primeira aplicação que vai consistir em uma **calculadora básica** com **as quatro operações aritméticas**. Para criar um projeto no **Android Studio** vamos no menu **“File”/“New”/“New Project”**. Confira na figura seguinte:





New Project (pelo menu)






Desenvolvendo uma Calculadora Básica



 Create New Project 

 Select a Project Template


Phone and Tablet

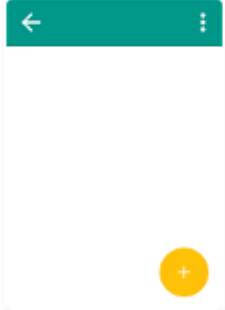
Wear OS


TV

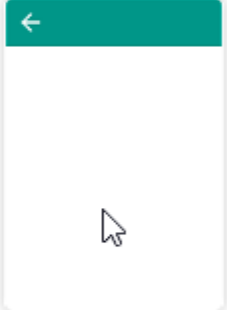
Automotive


Android Things

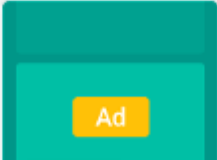

No Activity

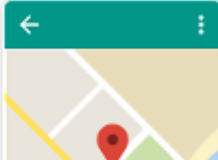

Basic Activity

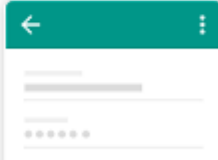

Bottom Navigation Activity


Empty Activity




Ad





Empty Activity
Creates a new empty activity.

Previous

Next

Cancel

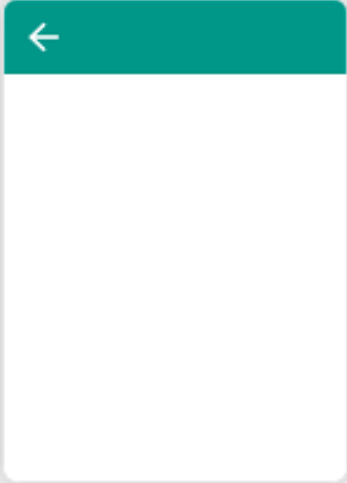
Finish



Desenvolvendo uma Calculadora Básica



- Após selecionar “**Empty Activity**” e clicar em “**Next**”, irá se abrir a caixa de diálogo abaixo:



Empty Activity
Creates a new empty activity.

Name
CalculadoraBasica

Package name
com.example.calculadorabasica

Save location
C:\ProjetosMobile\CalculadoraBasica

Language
Java

Minimum SDK
API 23: Android 6.0 (Marshmallow)

i Your app will run on approximately **84,9%** of devices.
[Help me choose](#)

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



Desenvolvendo uma Calculadora Básica



- Após digitar “**CalculadoraBasica**” para Name do Projeto, **Java** para Language, **API 23: Android 6** para Minimum SDK, clique em “**Finish**”:

Name
CalculadoraBasica

Package name
com.example.calculadorabasica

Save location
C:\ProjetosMobile\CalculadoraBasica

Language
Java

Minimum SDK
API 23: Android 6.0 (Marshmallow)

i Your app will run on approximately 84,9% of devices.
[Help me choose](#)

☐ Use legacy android.support libraries ?

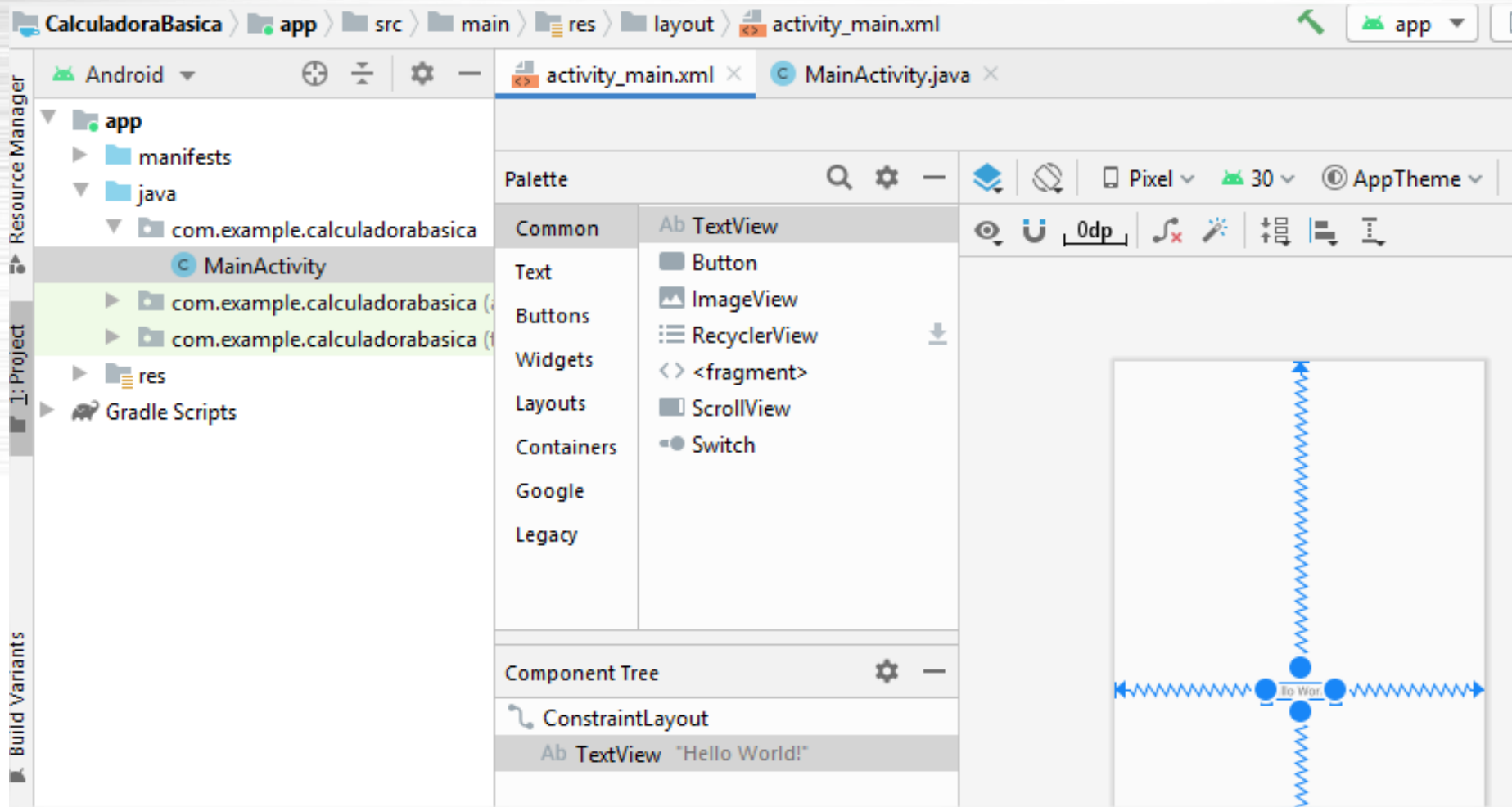
Previous **Next** **Cancel** **Finish**



Desenvolvendo uma Calculadora Básica



- Tela do Projeto “CalculadoraBasica”:

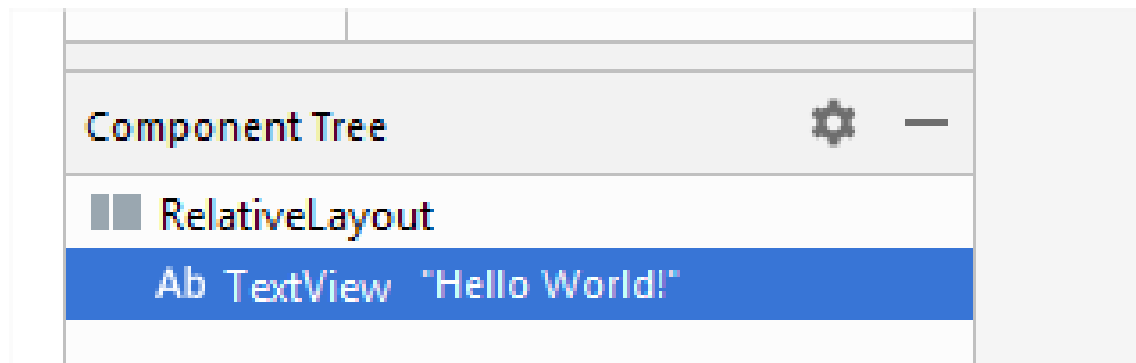
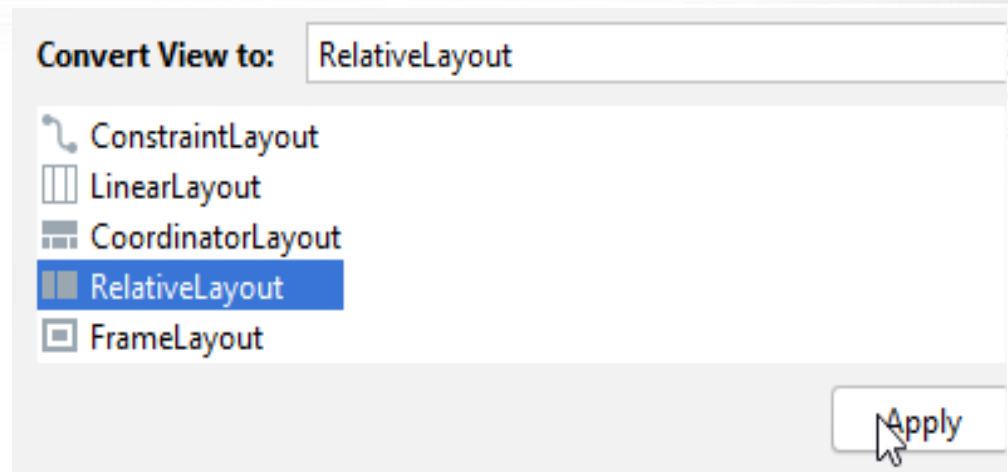
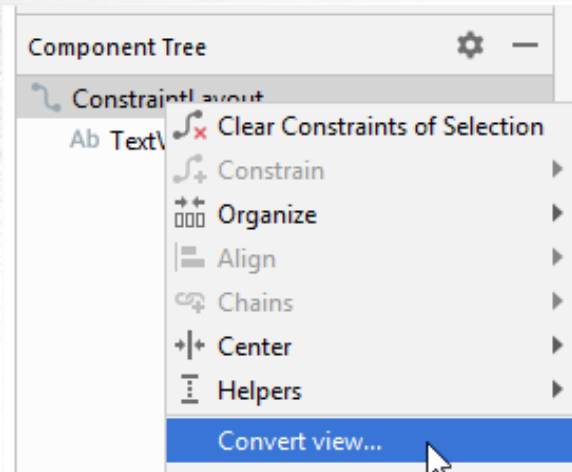




Desenvolvendo uma Calculadora Básica



- Vamos converter para **RelativeLayout**:



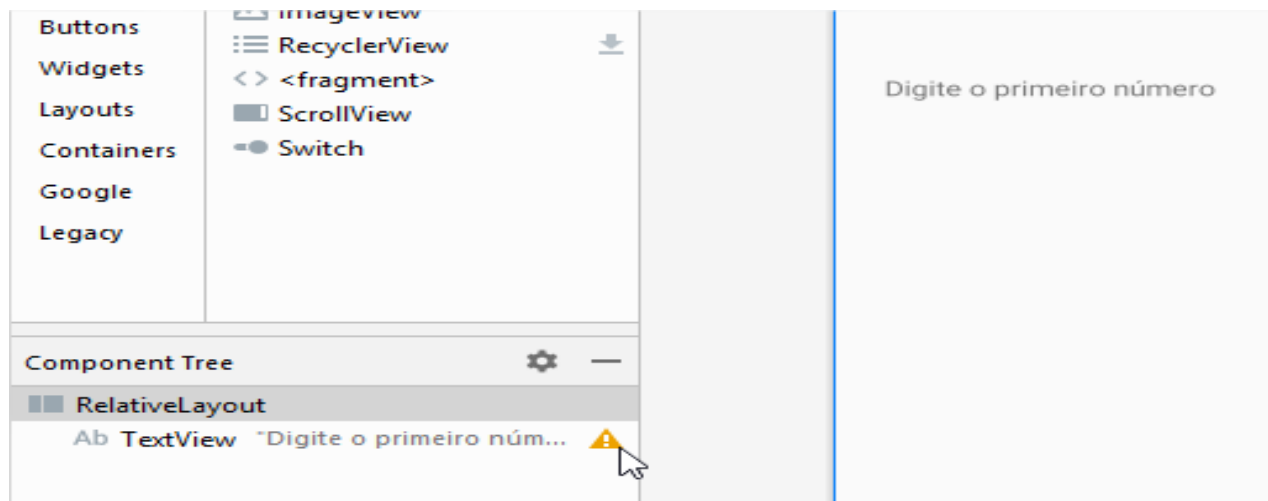


Desenvolvendo uma Calculadora Básica



- Na tela da aplicação selecione o componente **TextView** na tela (cuja frase está escrito “**Hello world**”) e vamos alterar as seguintes propriedades, como segue:

Propriedade	Valor
text	Digite o primeiro número

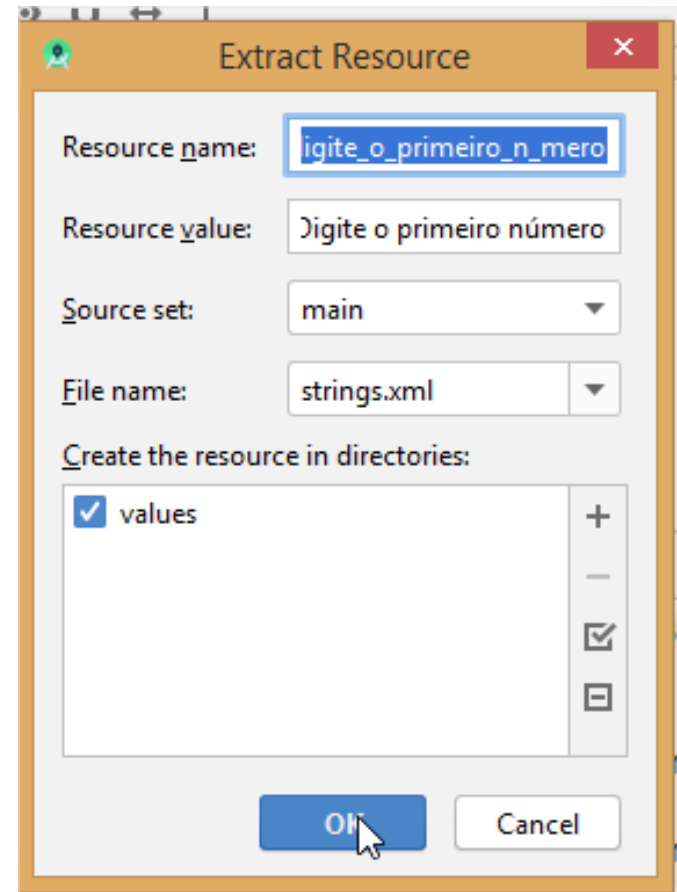
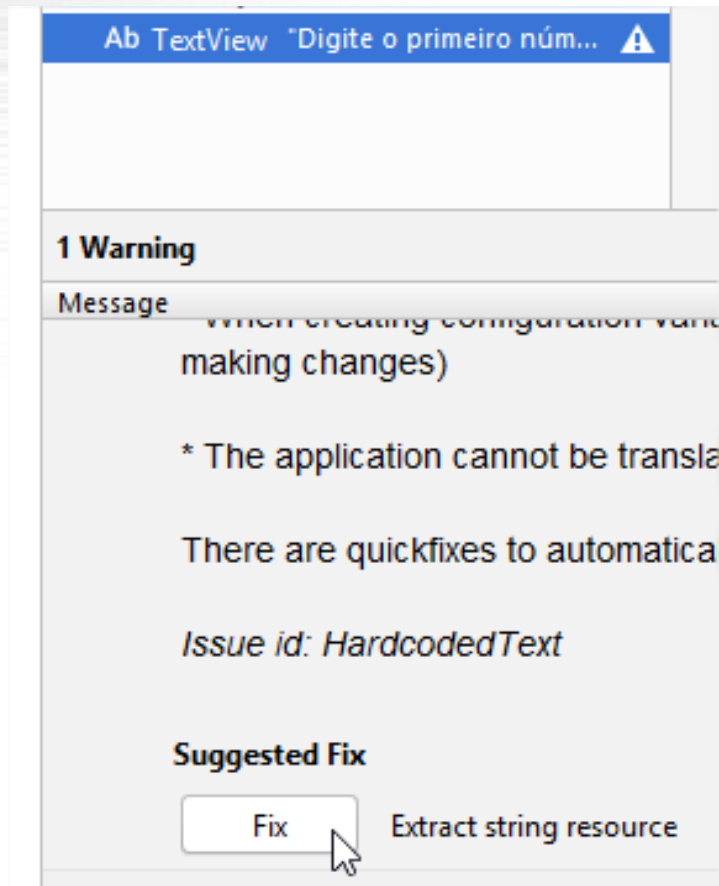




Desenvolvendo uma Calculadora Básica



- Incluir os textos em **strings.xml** conforme já foi ensinado:

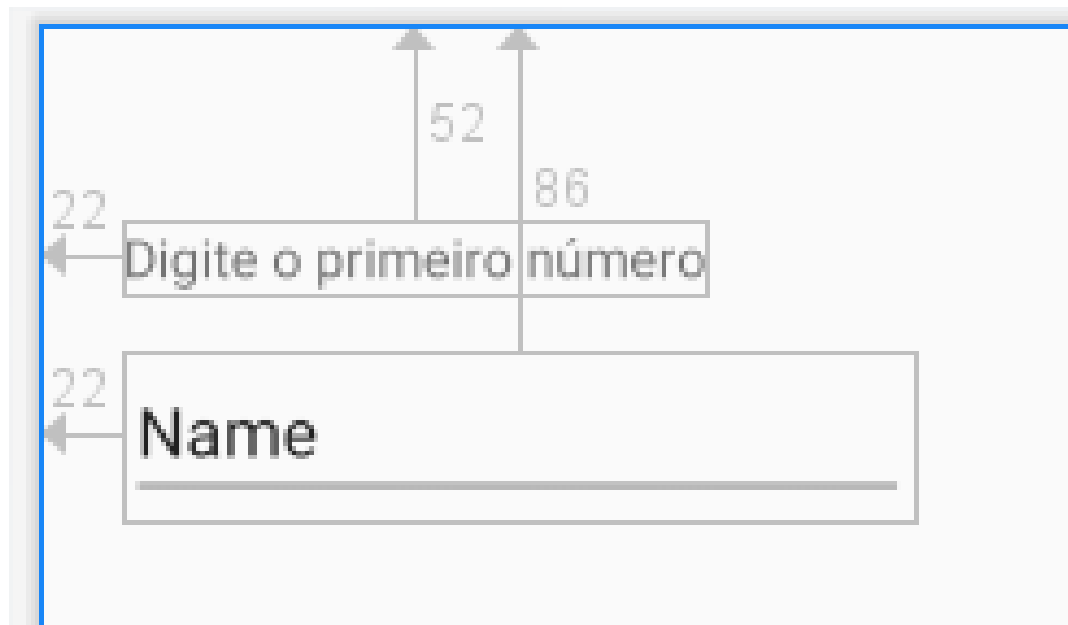




Desenvolvendo uma Calculadora Básica



- Agora arraste e solte um componente “**Plain Text**” (**EditText**) presente dentro da seção “**Text**” abaixo do título. Veja como ficou em seguida:





Desenvolvendo uma Calculadora Básica



- Com o componente selecionado, vamos em suas propriedades para alterar os seguintes valores, como segue:

EditText (Plan Text)

Propriedade	Valor
layout:width	match_parent
id	ednumero1
Text	DEIXAR EM BRANCO

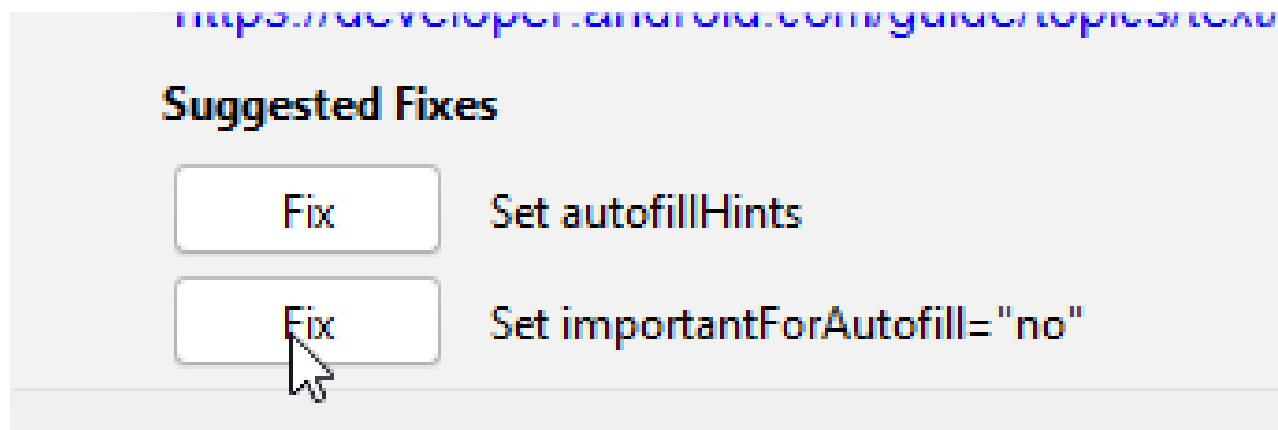
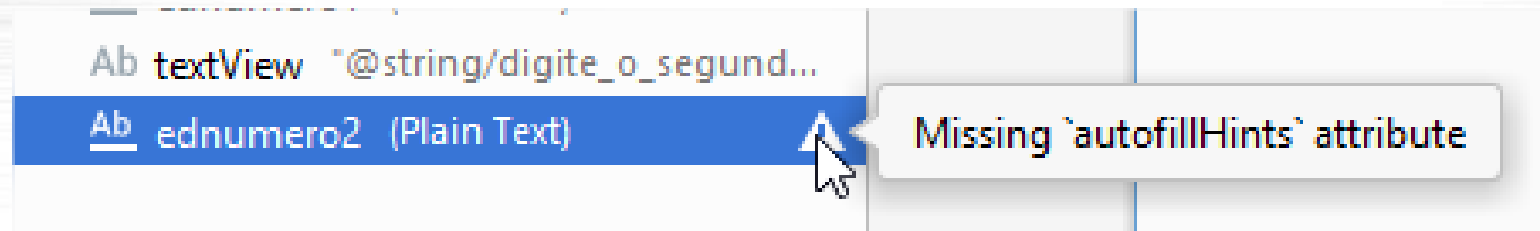




Desenvolvendo uma Calculadora Básica



- Na alerta “**autofillHints**”, clique na sugestão **Set importantForAutofill=“no”**, conforme abaixo:

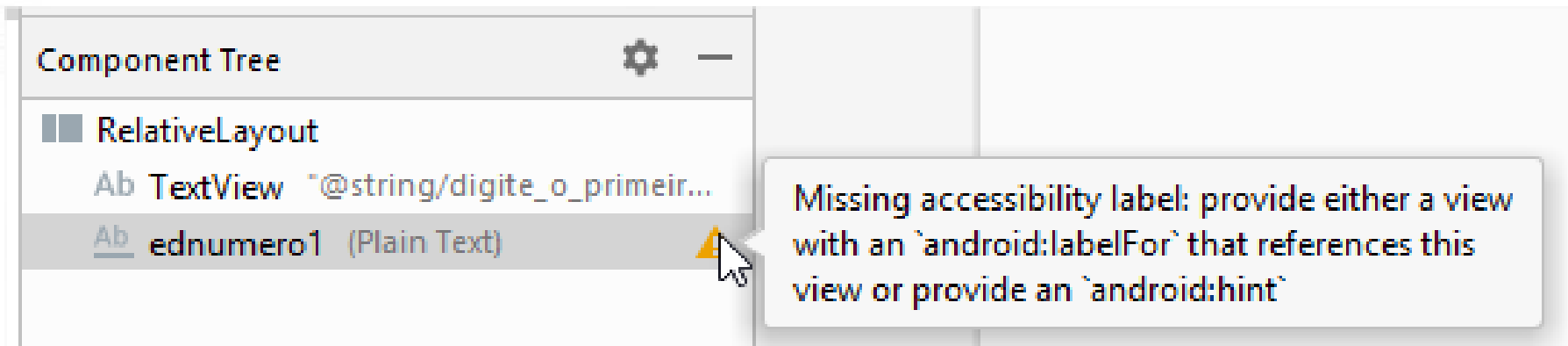




Desenvolvendo uma Calculadora Básica



- Após as configurações ainda é observado uma alerta sobre o **android:hint** que é para especificar um texto padrão no fundo do campo de texto, mas quero deixar em branco, com isso, vamos ao código do **strings.xml** e acrescentar a linha **<string name="embranco"></string>**

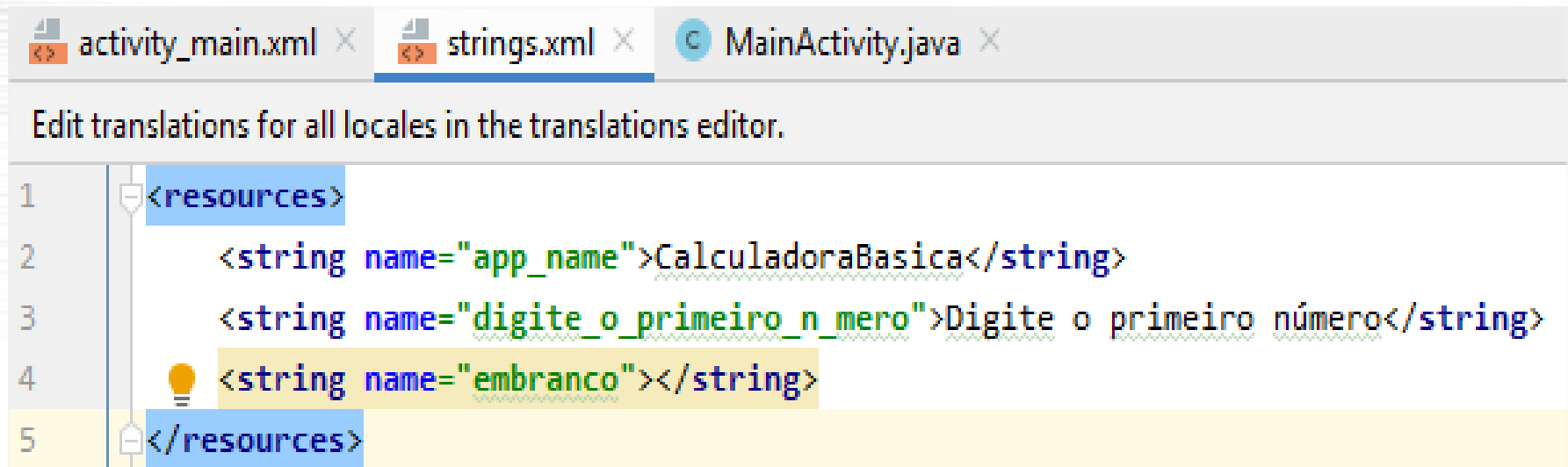




Desenvolvendo uma Calculadora Básica



- Para textos em branco, acrescentamos ao código **strings.xml** a linha abaixo:



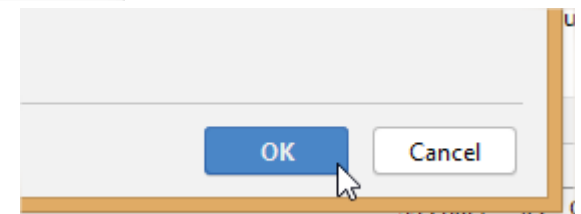
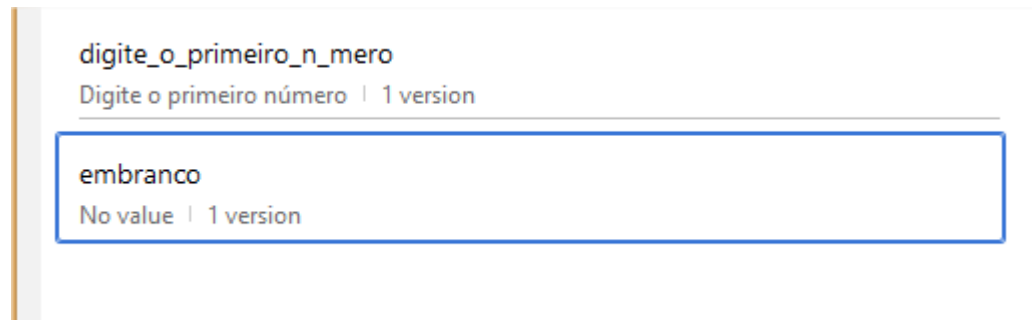
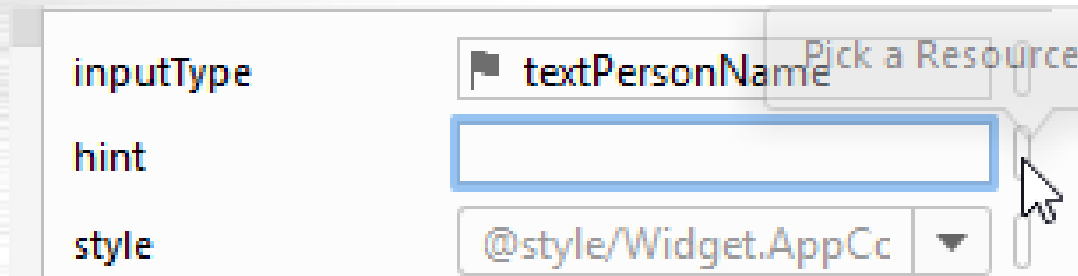
```
activity_main.xml x strings.xml x MainActivity.java x
Edit translations for all locales in the translations editor.
1 <resources>
2     <string name="app_name">CalculadoraBasica</string>
3     <string name="digite_o_primeiro_n_mero">Digite o primeiro número</string>
4     <string name="embranco"></string>
5 </resources>
```



Desenvolvendo uma Calculadora Básica



- Configurando o “**embranco**” ao **ednumero1**, na propriedade **hint**:

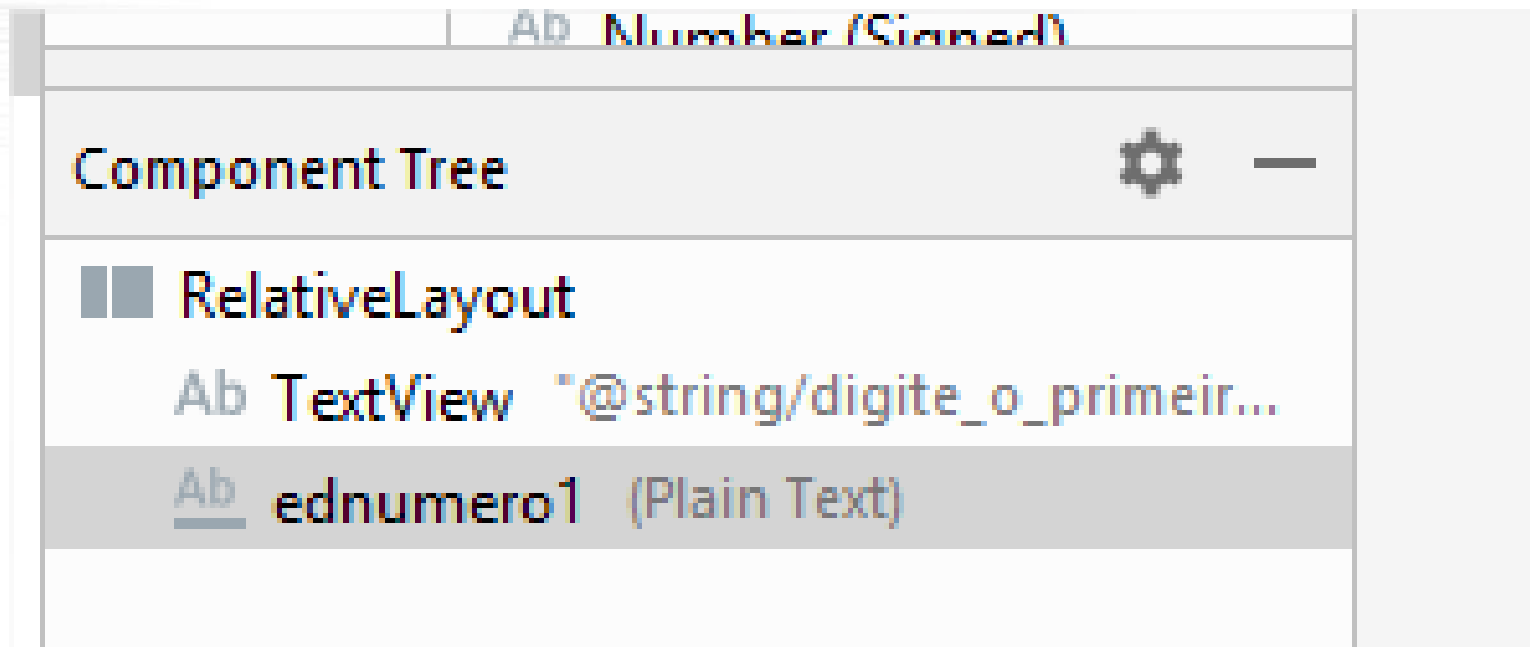




Desenvolvendo uma Calculadora Básica



- Observe que não existe nenhum erro, e nenhuma alerta:





Desenvolvendo uma Calculadora Básica



- Na propriedade “**layout:width**” especificamos a largura do nosso componente, que no caso de nossa aplicação possuirá (ocupará) a mesma largura da tela do dispositivo (“**match_parent**”).
 - Em “**id**” especificamos o nome do nosso componente, que será tratado (e identificado) na programação ao utilizarmos a linguagem Java.
-



Desenvolvendo uma Calculadora Básica



- Porque alterar a sua **ID** ? Isso é necessário pois vamos “manipular” esse componente através do código Java, então nada mais justo do que trabalhar com componentes cujos nomes estejam de forma clara e organizada.
- Agora arraste e solte um componente **TextView** abaixo da caixa de texto que inserimos, e em seguida altere as seguintes propriedades e os demais ajustes necessários:

TextView

Propriedade	Valor
text	Digite o segundo número



Desenvolvendo uma Calculadora Básica



- Logo após , arraste e solte um componente **Plain Text** (EditText), situado na guia “Text Fields”, abaixo do componente acima inserido, e altere seu nome (ID) para “**ednumero2**” e sua largura (layout:width) para “**match_parent**”. (conforme já foi mostrado).

Digite o primeiro número

Digite o segundo número



Desenvolvendo uma Calculadora Básica



- Agora vamos adicionar um componente **Button** abaixo da **caixa de texto**, que vai ser o nosso botão de “**somar**” os números. Depois de adicionar, vamos alterar as suas propriedades, conforme é mostrado abaixo:

Button

Propriedade	Valor
text	Somar
layout:width	match_parent
id	btsomar



Desenvolvendo uma Calculadora Básica



- Vejamos o resultado abaixo:

Digite o primeiro número

Digite o segundo número

SOMAR

Desenvolvendo uma Calculadora Básica

- Conforme os outros componentes, para os botões também teremos que acatar a sugestão para inclusão no **strings.xml**:

The screenshot shows an IDE interface with a 'Component Tree' on the left and a visual editor on the right. In the 'Component Tree', the 'RelativeLayout' is expanded, showing several 'TextView' components. The 'bt_somar' button is selected, which has the text 'Somar'. A warning icon is next to it, and a tooltip explains that the hardcoded string 'Somar' should be replaced with a string resource. Below the tree, a 'Suggested Fix' panel offers a 'Fix' button to 'Extract string resource'.

Component Tree

- RelativeLayout
 - TextView "@string/digite_o_primeir..."
 - ednumero1 (Plain Text)
 - textView "@string/digite_o_segund..."
 - ednumero2 (Plain Text)
 - bt_somar "Somar"**

Hardcoded string "Somar", should use `@string` resource

Issue id: HardcodedText

Suggested Fix


Fix Extract string resource



Desenvolvendo uma Calculadora Básica



- Nosso arquivo strings.xml está assim, por enquanto:

```
1 <resources>
2     <string name="app_name">CalculadoraBasica</string>
3     <string name="digite_o_primeiro_n_mero">Digite o primeiro número</string>
4     <string name="embranco"></string>
5     <string name="digite_o_segundo_n_mero">Digite o segundo número</string>
6      <string name="somar">Somar</string>
7 </resources>
```



Desenvolvendo uma Calculadora Básica



- Para começarmos, vamos fazer o teste da nossa aplicação realizando somente soma dos números.
- Vamos inserir o código em nossa classe (o arquivo “**MainActivity.java**”), basta seguir os procedimentos da figura no próximo slide.





Desenvolvendo uma Calculadora Básica



- Vamos para o arquivo **MainActivity.java**

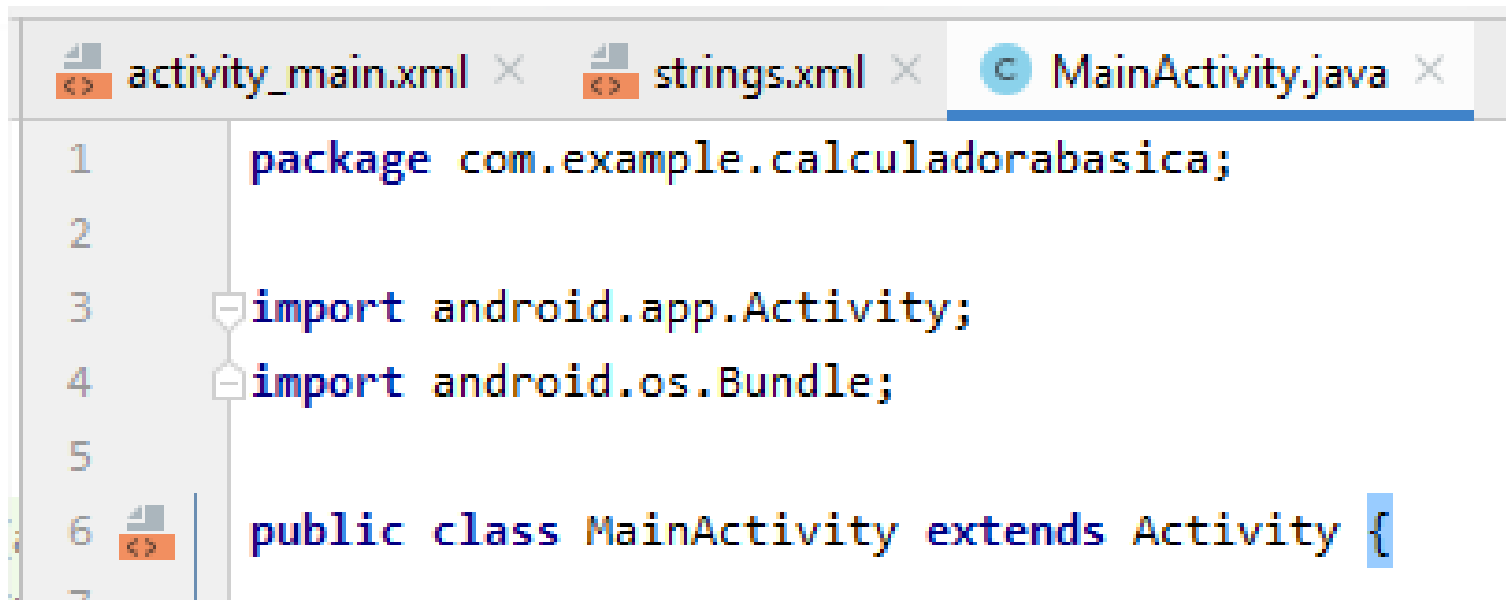
```
1 package com.example.calculadorabasica;
2
3 import androidx.appcompat.app.AppCompatActivity;
4
5 import android.os.Bundle;
6
7 public class MainActivity extends AppCompatActivity {
8
9     @Override
10    protected void onCreate(Bundle savedInstanceState) {
11        super.onCreate(savedInstanceState);
12        setContentView(R.layout.activity_main);
13    }
14 }
```



Desenvolvendo uma Calculadora Básica



- Vamos alterar a biblioteca de import **androidx.appcompat.app.AppCompatActivity;** para **import android.app.Activity;**
- Alterar também no lugar de extends **AppCompatActivity**, para extends **Activity**



```
activity_main.xml x strings.xml x MainActivity.java x
1 package com.example.calculadorabasica;
2
3 import android.app.Activity;
4 import android.os.Bundle;
5
6 public class MainActivity extends Activity {
7
```



Desenvolvendo uma Calculadora Básica



- Vamos importar alguns pacotes da plataforma Android que serão necessários para o desenvolvimento da nossa aplicação.

```
import android.widget.*;  
import android.view.*;  
import android.app.*;
```





Desenvolvendo uma Calculadora Básica



- Vejamos como está o código até agora:

```
activity_main.xml × strings.xml × MainActivity.java ×
1 package com.example.calculadorabasica;
2
3 import android.app.Activity;
4 import android.os.Bundle;
5 import android.widget.*;
6 import android.view.*;
7 import android.app.*;
8
9 public class MainActivity extends Activity {
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_main);
15     }
16 }
```



Desenvolvendo uma Calculadora Básica



- Agora no código do nosso programa, antes da linha:

```
@Override
```

- Digite:

```
EditText ednumero1,ednumero2;  
Button btsomar;
```



Desenvolvendo uma Calculadora Básica



- Vejamos como ficou na figura seguinte:

```
1      package com.example.calculadorabasica;
2
3      import android.app.Activity;
4      import android.os.Bundle;
5      import android.widget.*;
6      import android.view.*;
7      import android.app.*;
8
9      public class MainActivity extends Activity {
10
11          EditText ednumero1, ednumero2;
12          Button btsonmar;
13
14          @Override
15          protected void onCreate(Bundle savedInstanceState) {
16              super.onCreate(savedInstanceState);
17              setContentView(R.layout.activity_main);
18          }
19      }
```



Desenvolvendo uma Calculadora Básica



- Explicando o código do slide anterior:
 - Os **widgets** vão ser usados no nosso código **Java**, com isso, no código **XML**, a cada **widget** criado na tela (**EditText** e **Button**) vamos criar um objeto **Java** para acessar esse componente através do uso da Classe **EditText** e **Button**.
 - Cada **widget** no **XML** possui o seu respectivo “em classe” **Java**, logo, se possui um **widget** **Button**, para acessá-lo devo fazer uso da classe **Button** e assim vai.
-



Desenvolvendo uma Calculadora Básica



- Agora dentro do método **onCreate** após a linha:

```
setContentView(R.layout.activity_main);
```

- Digite as seguintes linhas de código:

```
ednumero1 = (EditText) findViewById(R.id.ednumero1);  
ednumero2 = (EditText) findViewById(R.id.ednumero2);  
btsomar   = (Button) findViewById(R.id.btsomar);
```



Desenvolvendo uma Calculadora Básica



- Vejam como está ficando o código de nossa aplicação Calculadora:

```
9  public class MainActivity extends Activity {  
10  
11      EditText ednumero1, ednumero2;  
12      Button btsomar;  
13  
14      @Override  
15      protected void onCreate(Bundle savedInstanceState) {  
16          super.onCreate(savedInstanceState);  
17          setContentView(R.layout.activity_main);  
18          ednumero1 = (EditText) findViewById(R.id.ednumero1);  
19          ednumero2 = (EditText) findViewById(R.id.ednumero2);  
20          btsomar = (Button) findViewById(R.id.btsomar);  
21      }  
22  }
```



Desenvolvendo uma Calculadora Básica



- Explicando as linhas de comandos que foram adicionadas na nossa calculadora. A linha:

```
ednumero1 = (EditText) findViewById(R.id.ednumero1);
```

- Faz referência ao primeiro **EditText**, através do método **findViewById** com o parâmetro **“R.id.numero1”**.
-



Desenvolvendo uma Calculadora Básica



- Se lembra do nome da primeira **EditText** que está no código **XML**? Ela se chama **“ednumero1”**.
 - Vamos entender. Observe que para fazer referência ao **EditText** pelo método **findViewById** eu passei o parâmetro **“R.id.ednumero1”**.
 - Na segunda instrução que digitamos, para fazer referência à segunda **EditText**, cujo nome é **“ednumero2”**, pelo método **findViewById**, passei o parâmetro **“R.id.ednumero2”**.
-



Desenvolvendo uma Calculadora Básica



- Como você pode ver, estamos fazendo uso da **classe R** (uma classe “**interna**” do **Android**, onde todos os elementos e diretórios são estruturados em atributos da classe) que funciona como interface entre o código Java e o arquivo XML. O procedimento é o mesmo para o componente **Button**.



Tranquilo até
aqui?????





Desenvolvendo uma Calculadora Básica



- Agora iremos adicionar um evento em nosso componente **Button** que será responsável por “**detectar**” toda vez que ele for “**clicado**” (tocado na tela), executando um conjunto de instruções após o evento (que vai consistir na soma dos números e na exibição do resultado).
 - Para adicionarmos esse evento em nosso componente, basta escrevermos, após a última instrução que adicionamos, a linha de código destacado em azul que está no próximo slide.
-



Desenvolvendo uma Calculadora Básica



```
protected void onCreate(Bundle savedInstanceState) {  
  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
  
    ednumero1 = (EditText) findViewById(R.id.ednumero1);  
    ednumero2 = (EditText) findViewById(R.id.ednumero2);  
    btsomar = (Button) findViewById(R.id.btsomar);  
  
    btsomar.setOnClickListener(new View.OnClickListener() {  
  
        @Override  
        public void onClick(View v) {  
  
            double num1 = Double.parseDouble(  
                ednumero1.getText().toString());  
  
            double num2 = Double.parseDouble(  
                ednumero2.getText().toString());  
            double soma = num1 + num2;  
  
            AlertDialog.Builder dialogo = new  
                AlertDialog.Builder(MainActivity.this);  
  
            dialogo.setTitle("Resultado soma");  
  
            dialogo.setMessage("A soma é " + soma);  
        }  
    });  
}
```



Desenvolvendo uma Calculadora Básica



```
        dialogo.setNeutralButton("OK", null);  
        dialogo.show();  
    }  
});  
}
```

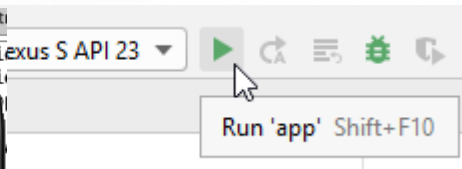
- Toda vez que eu clicar no botão ele irá mostrar o resultado da soma na tela através de uma caixa de mensagem.
-



Desenvolvendo uma Calculadora Básica



- Vamos executar a nossa aplicação?





Desenvolvendo uma Calculadora Básica



- **OBS:** provavelmente durante a execução da aplicação ao entrar com um número, deve ter surgido no dispositivo um teclado virtual, para ocultar ele é só pressionar **ESC**.
 - O método **setOnClickListener** serve para definir um evento de “**clique**” em um componente. Como parâmetro, criamos uma instância da interface **OnClickListener**, e dentro da mesma existe um método chamado **onClick**, que será disparado toda vez que o botão for clicado.
-



Desenvolvendo uma Calculadora Básica



```
double num1 = Double.parseDouble(ednumero1.getText().toString());
```

- Cria uma variável chamada **num1** e atribui a ela o valor que está contido dentro do componente identificado como **ednumero1**. Temos que utilizar o método **parseDouble** da classe **Double** pois o conteúdo é uma **String**. Observem que é chamado o método **getText** de **ednumero1** para retornar o conteúdo. Diferente de muitos métodos de retorno **String**, esse método **getText** não retorna uma **String**, mais sim um tipo chamado **Editable**.
-



Desenvolvendo uma Calculadora Básica



- Por isso chamamos o método **toString** de **getText** para que me retornasse uma **String**.
- Logo após a soma dos números que será armazenada na variável soma, vem o código em seguida:

```
AlertDialog.Builder dialogo = new  
    AlertDialog.Builder(context: MainActivity.this);  
dialogo.setTitle("Resultado da soma");  
dialogo.setMessage("A soma é: " + soma);  
dialogo.setNeutralButton(text: "OK", listener: null);  
dialogo.show();
```



Desenvolvendo uma Calculadora Básica



- Que mostra a soma dos números digitados na tela. Para conseguirmos exibir uma mensagem na tela, tivemos que fazer uso da classe **AlertDialog.Builder**, responsável por criar caixas de diálogo e exibi-las. Vamos aos comentários. A linha de comando:

```
AlertDialog.Builder dialogo = new  
    AlertDialog.Builder( context: MainActivity.this);
```



Desenvolvendo uma Calculadora Básica



- Cria a instância da classe **AlertDialog.Builder** que será representada e guardada dentro da variável **dialogo**. Na linha seguinte:

```
dialogo.setTitle("Resultado soma");
```

- Define o título da caixa de diálogo através do método **setTitle**. Na linha seguinte:

```
dialogo.setMessage("A soma é " + soma);
```

- Define a mensagem a ser exibida através do método **setMessage**.
-



Desenvolvendo uma Calculadora Básica



- Na linha seguinte:

```
dialogo.setNeutralButton("OK", null);
```

- Define o botão “**OK**” da caixa de texto através do método **setNeutralButton**. O parâmetro **null** indica que nenhuma ação será executada quando o botão for clicado (simplesmente a caixa será fechada e nada mais). E para finalizar:

```
dialogo.show();
```

- Responsável por “**exibir**” a mensagem na tela.
-



Desenvolvendo uma Calculadora Básica



- Agora vamos continuar as outras operações certo? Retornaremos então para a tela da nossa aplicação e vamos adicionar mais 3 botões referentes as operações restantes. Vamos adicionar na tela mais três botões como segue no próximo slide (um em baixo do outro).

Button

Propriedade	Valor
Id	btsubtrair
text	Subtrair
layout:width	match_parent



Desenvolvendo uma Calculadora Básica



Button

Propriedade	Valor
id	btmultiplicar
text	Multiplicar
layout:width	match_parent

Button

Propriedade	Valor
id	btdividir
text	Dividir
layout:width	match_parent



Desenvolvendo uma Calculadora Básica



- Depois de “**finalizado**” o que foi se pedido nos slides anteriores, vejam como ficou a tela da nossa aplicação:

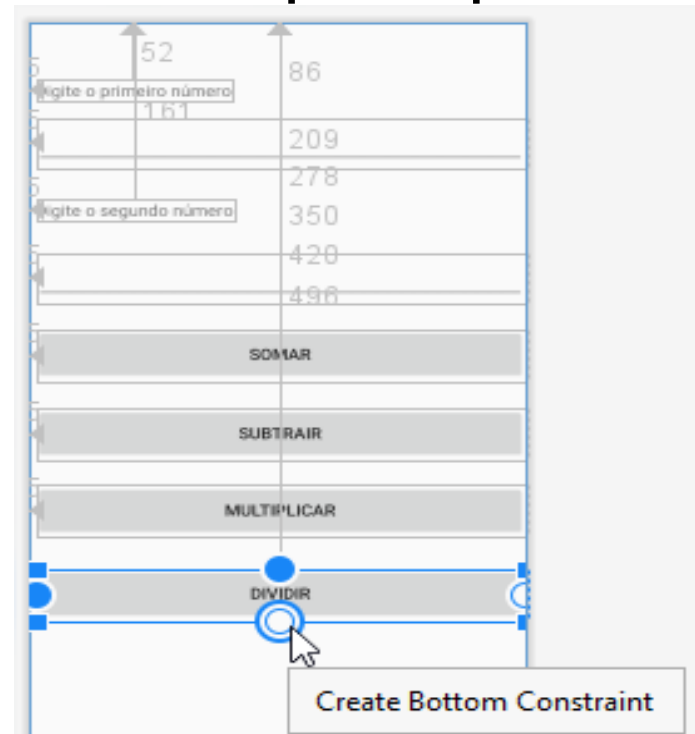
The image shows a mobile application interface for a basic calculator. It features two input fields for numbers, each with a label above it: "Digite o primeiro número" and "Digite o segundo número". Below the input fields are four large, light gray buttons with black text, stacked vertically: "SOMAR", "SUBTRAIR", "MULTIPLICAR", and "DIVIDIR". The interface is clean and minimalist, with a white background and light gray borders.



Desenvolvendo uma Calculadora Básica



- É importante observar se os componentes na tela estão somente com Top Constraint, ou seja, não pode ter o **Bottom Constraint**, para evitar que os componentes fiquem por cima dos outros na execução:





Desenvolvendo uma Calculadora Básica



- A tela da aplicação nada mais é do que uma estrutura XML. Vamos ver agora a estrutura XML que existe por trás dessa tela que desenhamos:

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout 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      tools:context=".MainActivity">
8
```



Desenvolvendo uma Calculadora Básica



```
9      <TextView
10          android:layout_width="wrap_content"
11          android:layout_height="wrap_content"
12          android:layout_alignParentStart="true"
13          android:layout_alignParentTop="true"
14          android:layout_marginStart="5dp"
15          android:layout_marginTop="52dp"
16          android:text="@string/digite_o_primeiro_n_mero" />
17
18      <EditText
19          android:id="@+id/ednumero1"
20          android:layout_width="match_parent"
21          android:layout_height="wrap_content"
22          android:layout_alignParentStart="true"
23          android:layout_alignParentTop="true"
24          android:layout_marginStart="5dp"
25          android:layout_marginTop="86dp"
26          android:ems="10"
27          android:hint="@string/embranco"
28          android:importantForAutofill="no"
29          android:inputType="textPersonName" />
```



Desenvolvendo uma Calculadora Básica



```
30
31 <TextView
32     android:id="@+id/textView"
33     android:layout_width="wrap_content"
34     android:layout_height="wrap_content"
35     android:layout_alignParentStart="true"
36     android:layout_alignParentTop="true"
37     android:layout_marginStart="5dp"
38     android:layout_marginTop="161dp"
39     android:text="@string/digite_o_segundo_n_mero" />
40
41 <EditText
42     android:id="@+id/ednumero2"
43     android:layout_width="match_parent"
44     android:layout_height="wrap_content"
45     android:layout_alignParentStart="true"
46     android:layout_alignParentTop="true"
47     android:layout_marginStart="5dp"
48     android:layout_marginTop="209dp"
49     android:ems="10"
50     android:hint="@string/embranco"
51     android:importantForAutofill="no"
52     android:inputType="textPersonName" />
53
```



Desenvolvendo uma Calculadora Básica



```
54 <Button
55     android:id="@+id/btsomar"
56     android:layout_width="match_parent"
57     android:layout_height="wrap_content"
58     android:layout_alignParentStart="true"
59     android:layout_alignParentTop="true"
60     android:layout_marginStart="5dp"
61     android:layout_marginTop="278dp"
62     android:text="@string/somar" />
63
64 <Button
65     android:id="@+id/btsubtrair"
66     android:layout_width="match_parent"
67     android:layout_height="wrap_content"
68     android:layout_alignParentStart="true"
69     android:layout_alignParentTop="true"
70     android:layout_marginStart="5dp"
71     android:layout_marginTop="350dp"
72     android:text="@string/subtrair" />
```



Desenvolvendo uma Calculadora Básica



```
73
74     <Button
75         android:id="@+id/btmultiplicar"
76         android:layout_width="match_parent"
77         android:layout_height="wrap_content"
78         android:layout_alignParentStart="true"
79         android:layout_alignParentTop="true"
80         android:layout_marginStart="5dp"
81         android:layout_marginTop="420dp"
82         android:text="@string/multiplicar" />
83
84     <Button
85         android:id="@+id/btdividir"
86         android:layout_width="match_parent"
87         android:layout_height="wrap_content"
88         android:layout_alignParentStart="true"
89         android:layout_alignParentTop="true"
90         android:layout_marginStart="5dp"
91         android:layout_marginTop="496dp"
92         android:text="@string/dividir" />
93
94 </RelativeLayout>
```



Desenvolvendo uma Calculadora Básica



- Agora retornando para o código do arquivo “**MainActivity.java**”, vamos declarar mais três atributos (variáveis) que vão corresponder aos botões que representam as operações restantes, conforme destaca a **linha em azul**:

```
:  
Button btsomar, btsubtrair,btmultiplicar, btdividir;  
:
```





Desenvolvendo uma Calculadora Básica



- Agora vamos atribuir para cada botão um evento de clique, fazendo com que eles efetuem a sua respectiva operação aritmética.
- Vamos continuar a codificação do método **onCreate**, digitando o código destacado em azul em no próximo slide.





Desenvolvendo uma Calculadora Básica



```
@Override
```

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);
```

```
    ednumero1 = (EditText) findViewById(R.id.ednumero1);  
    ednumero2 = (EditText) findViewById(R.id.ednumero2);  
    btsomar = (Button) findViewById(R.id.btsomar);
```

```
    btsubtrair = (Button) findViewById(R.id.btsubtrair);
```

```
    btmultiplicar=(Button)findViewById(R.id.btmultiplicar);
```

```
    btdividir = (Button) findViewById(R.id.btdividir);
```

```
    btsomar.setOnClickListener(new View.OnClickListener() {
```

```
        I        :
```

```
    });
```



Desenvolvendo uma Calculadora Básica



- Agora nos próximos slides vamos desenvolver os métodos **btsubtrair()**; **btmultiplicar()**; e **btdividir()**;



Digite o primeiro número

Digite o segundo número

SOMAR

SUBTRAIR

MULTIPLICAR

DIVIDIR

Método btsubtrair();

```
btsubtrair.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        double num1 = Double.parseDouble(ednumero1.getText().toString());  
        double num2 = Double.parseDouble(ednumero2.getText().toString());  
        double subtrai = num1 - num2;  
        AlertDialog.Builder dialogo = new  
            AlertDialog.Builder(context: MainActivity.this);  
        dialogo.setTitle("Resultado da subtração");  
        dialogo.setMessage("A subtração é: " + subtrai);  
        dialogo.setNeutralButton(text: "OK", listener: null);  
        dialogo.show();  
    }  
});
```

Método btmultiplicar();

```
btmultiplicar.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        double num1 = Double.parseDouble(ednumero1.getText().toString());  
        double num2 = Double.parseDouble(ednumero2.getText().toString());  
        double multiplica = num1 * num2;  
        AlertDialog.Builder dialogo = new  
            AlertDialog.Builder( context: MainActivity.this);  
        dialogo.setTitle("Resultado da multiplicação");  
        dialogo.setMessage("A multiplicação é: " + multiplica);  
        dialogo.setNeutralButton( text: "OK", listener: null);  
        dialogo.show();  
    }  
});
```

Método btdividir();

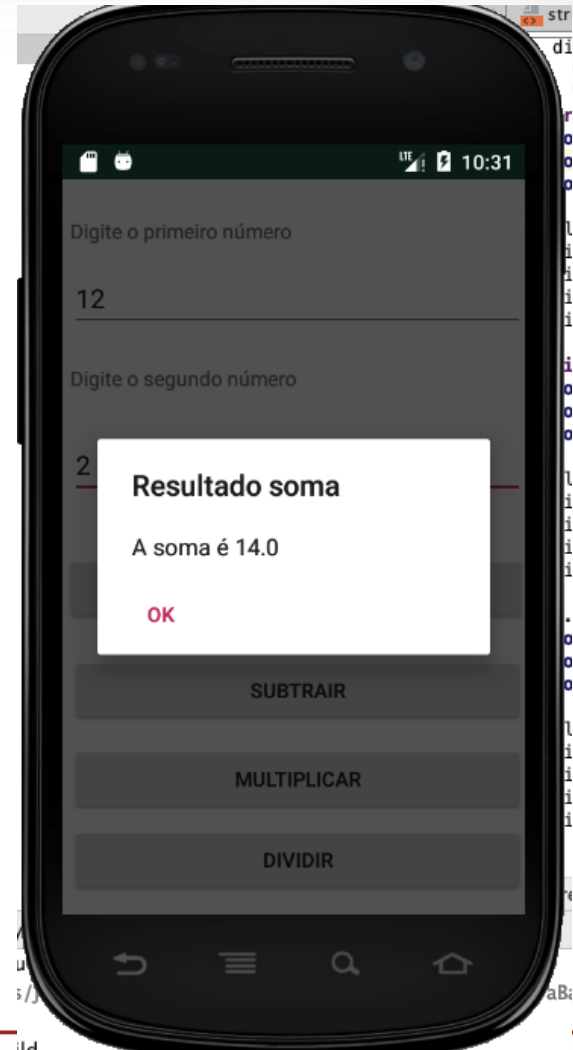
```
btdividir.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        double num1 = Double.parseDouble(ednumero1.getText().toString());  
        double num2 = Double.parseDouble(ednumero2.getText().toString());  
        double divide = num1 / num2;  
        AlertDialog.Builder dialogo = new  
            AlertDialog.Builder(context: MainActivity.this);  
        dialogo.setTitle("Resultado da divisão");  
        dialogo.setMessage("A divisão é: " + divide);  
        dialogo.setNeutralButton(text: "OK", listener: null);  
        dialogo.show();  
    }  
});
```



Desenvolvendo uma Calculadora Básica



- Testando a Soma:





Desenvolvendo uma Calculadora Básica



- Testando a Subtração:





Desenvolvendo uma Calculadora Básica



- Testando a Multiplicação:





Desenvolvendo uma Calculadora Básica



- Testando a Divisão:





Desenvolvendo uma Calculadora Básica



- Testando a Divisão por zero **sem tratamento**:



Exercício

- Tratar a divisão por zero, ou seja, se um número tentar ser dividido por ZERO, o programa terá que informar sobre a operação ilegal.
- Bom trabalho!!!!





Por hoje é só !!!!

Até a próxima aula...
