

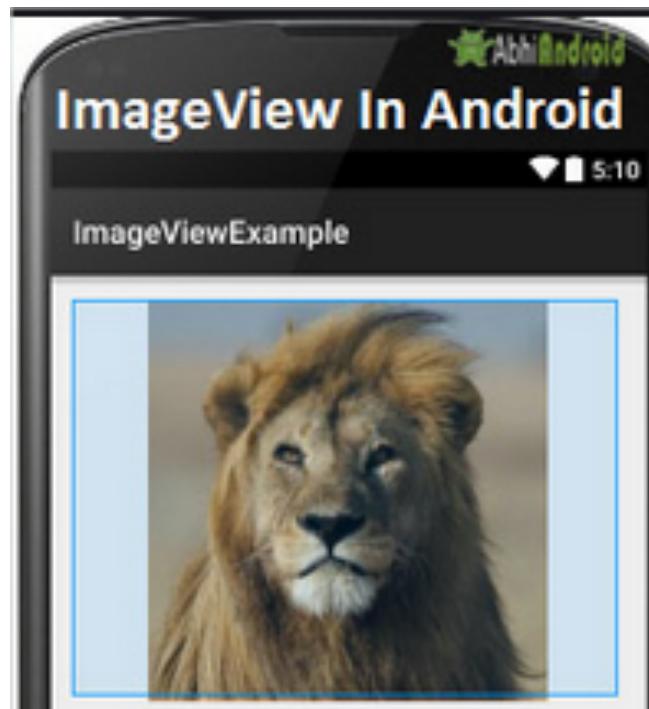
# Linguagem Técnica de Programação Mobile

**AULA 8 - Componentes de Imagens:  
ImageView e ImageSwitcher**

Prof. João Paulo Pimentel  
[joao.pimentel@projecao.br](mailto:joao.pimentel@projecao.br)

---

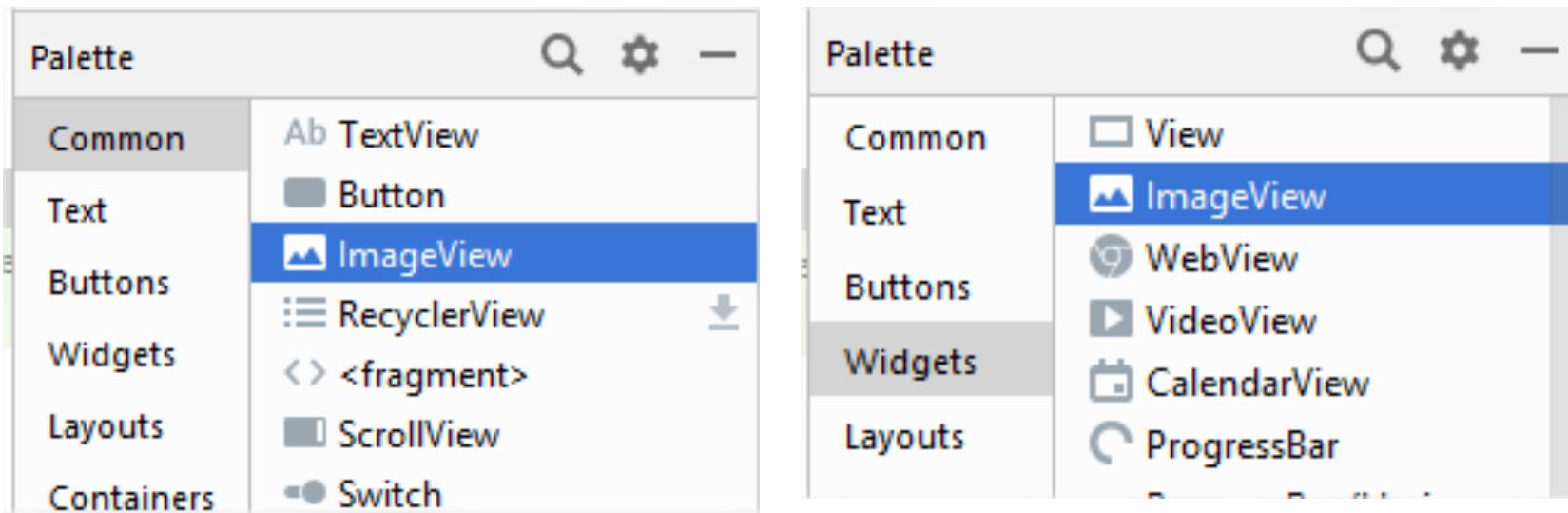
# Conhecendo o “ImageView”



---

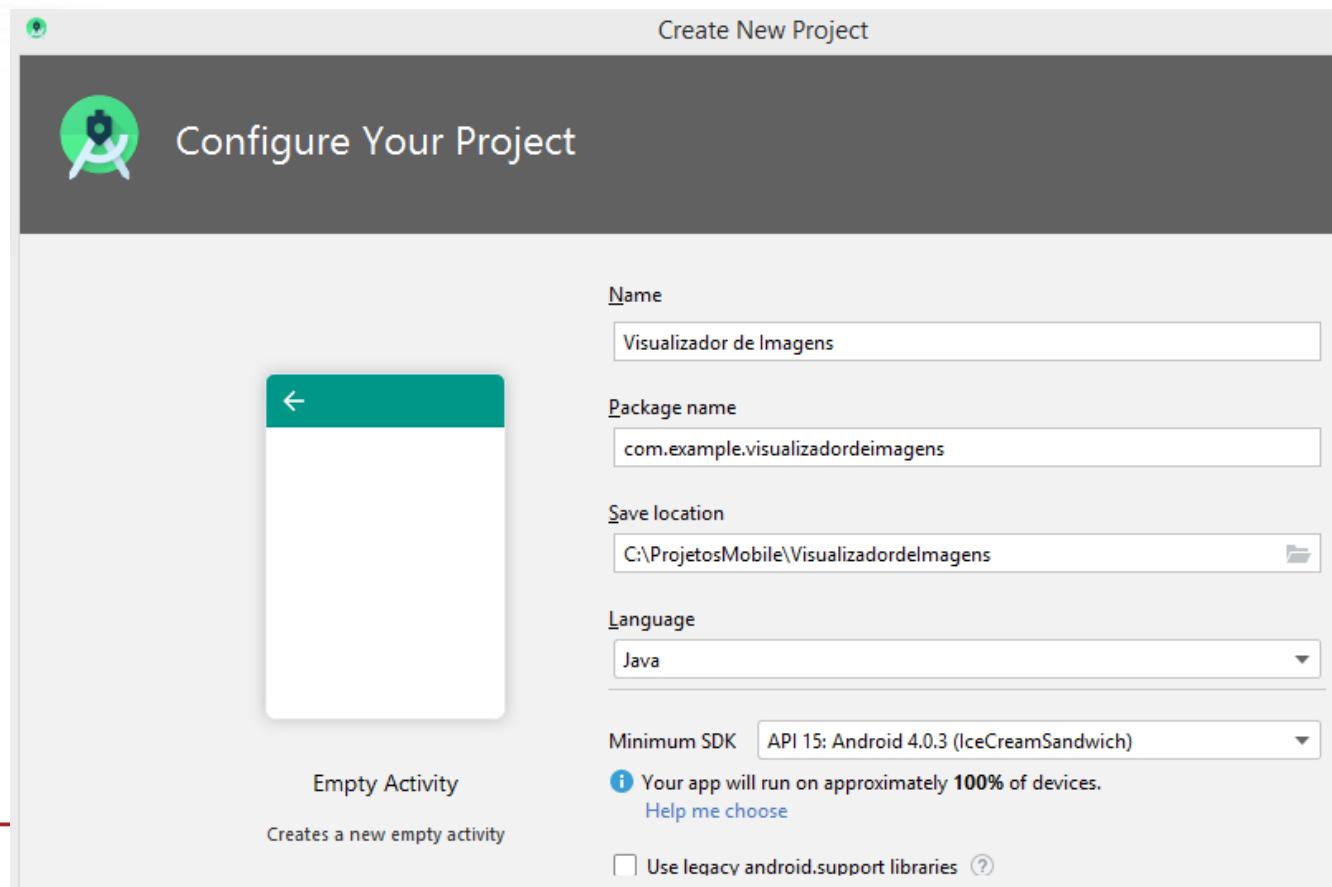
# Conhecendo o “ImageView”

- Esse componente simplesmente serve para exibir imagens que se colocam nele.
- Este componente se encontra na Palette **Common** e **também na Widgets**.
- Os formatos de imagens suportados por esse componente são : PNG, JPEG, BMP, GIF.



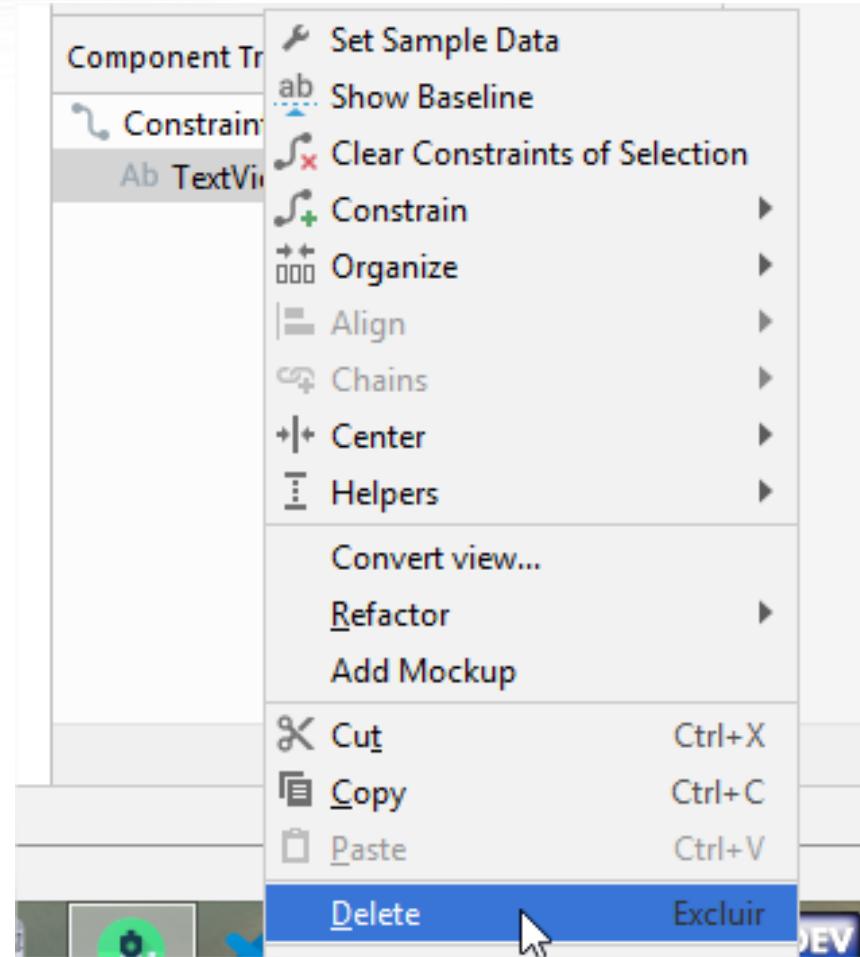
# App com o “ImageView”

- Desenvolvendo uma aplicação que visualiza imagens (com *ImageView*). Vamos criar um projeto com os seguintes dados abaixo:



# App com o “ImageView”

- Apague o componente “**TextView**” do *Hello World* do App:



# App com o “ImageView”

- Vamos adicionar duas imagens PNG (com a extensão .png) e uma JPG (.jpg), dentro da pasta “**drawable**”, presente dentro do diretório “**res**” (“doutor\_estrano.png”, “fundo\_tela.jpg” e “titulo\_avengers\_end\_game.png”), que já estão no **Blog da Disciplina**.



doutor\_estrano.png



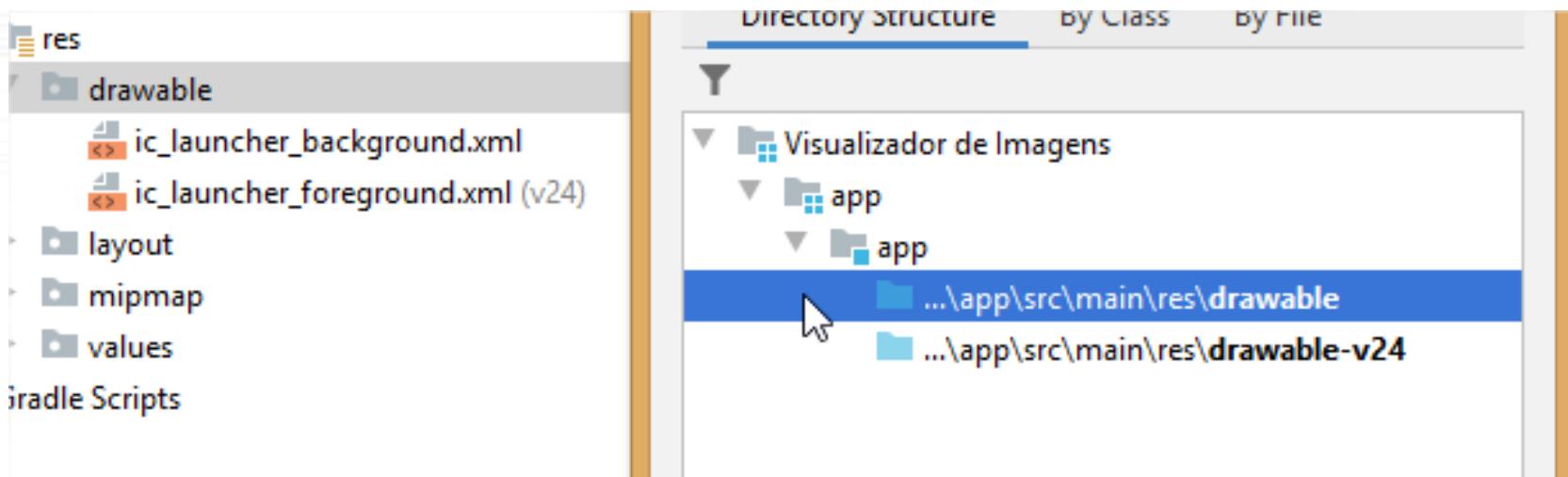
fundo\_tela.jpg



titulo\_avengers\_end\_game.png

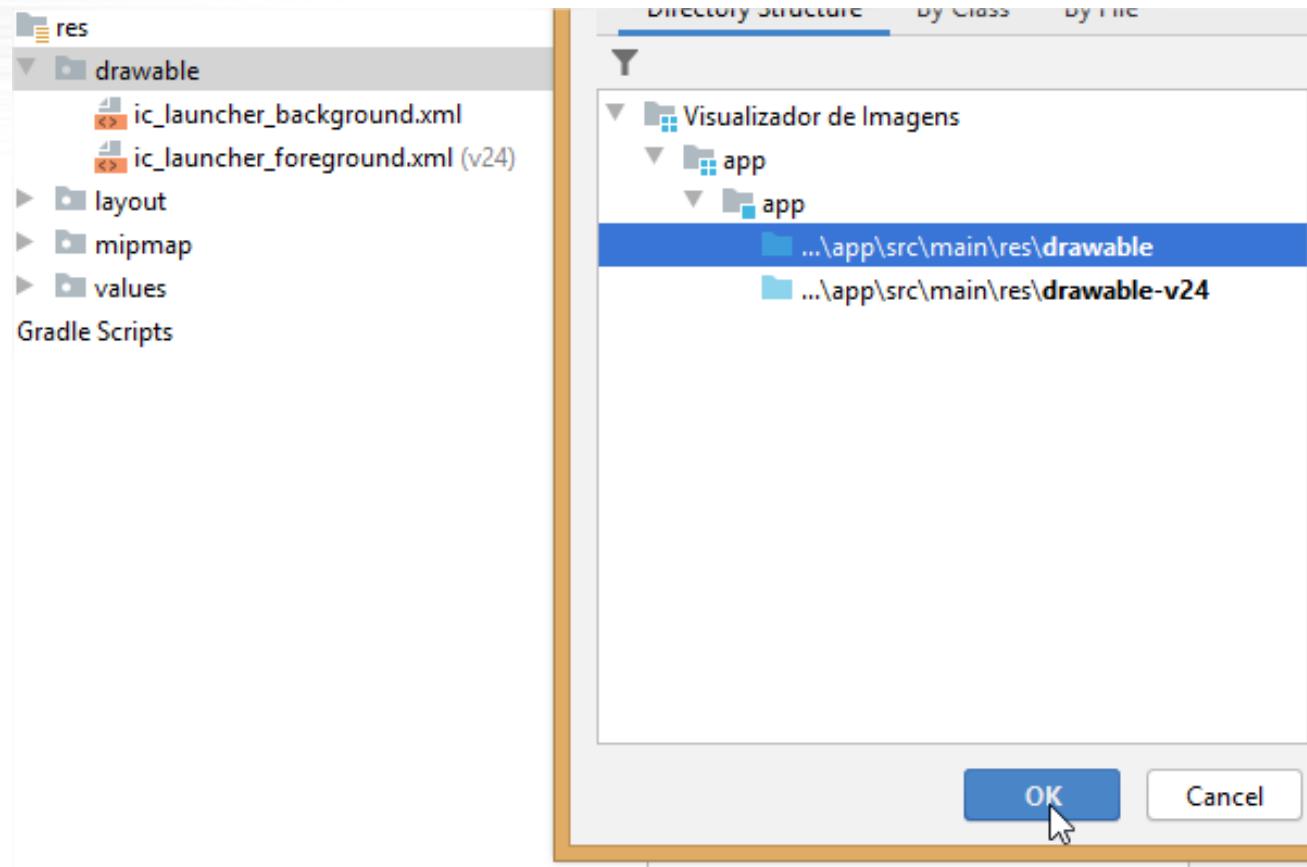
# App com o “ImageView”

- Quando colar em “drawable” é normal aparecer uma pergunta conforme abaixo (cuidado para não incluir no drawable-v24):



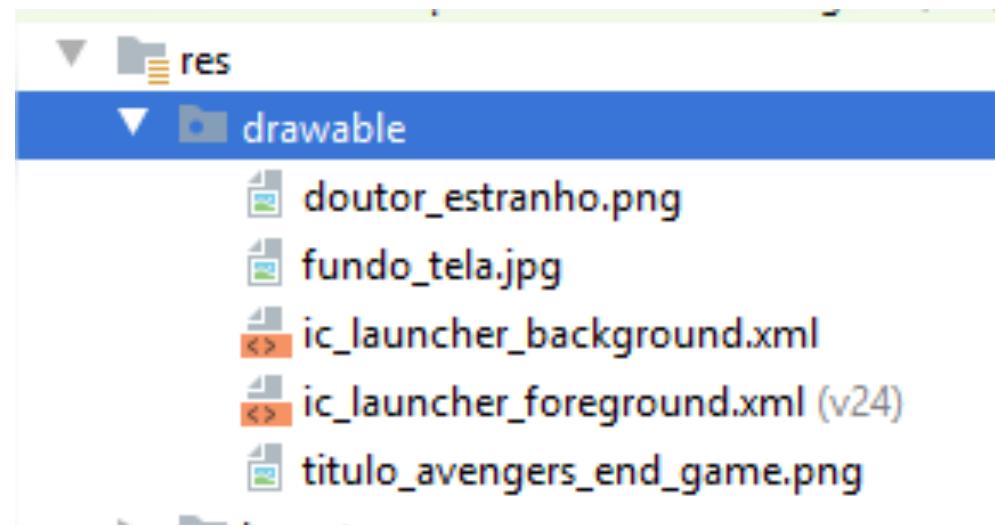
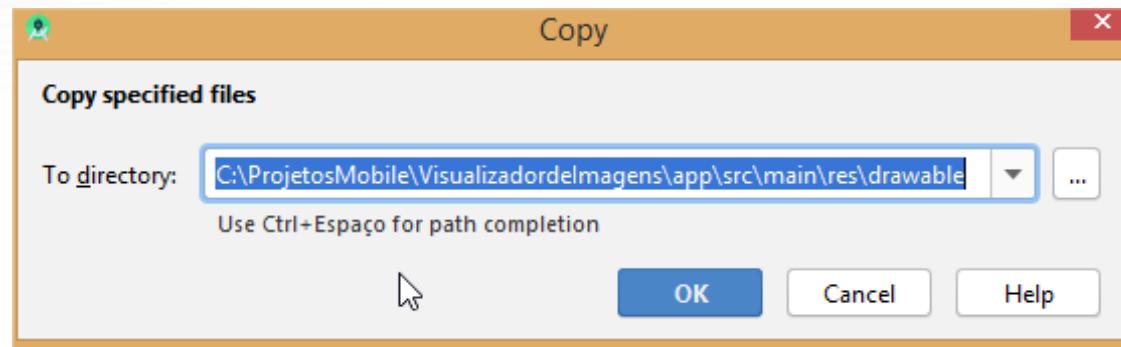
# App com o “ImageView”

- Selecionado a pasta **drawable**, basta clicar em **OK**:



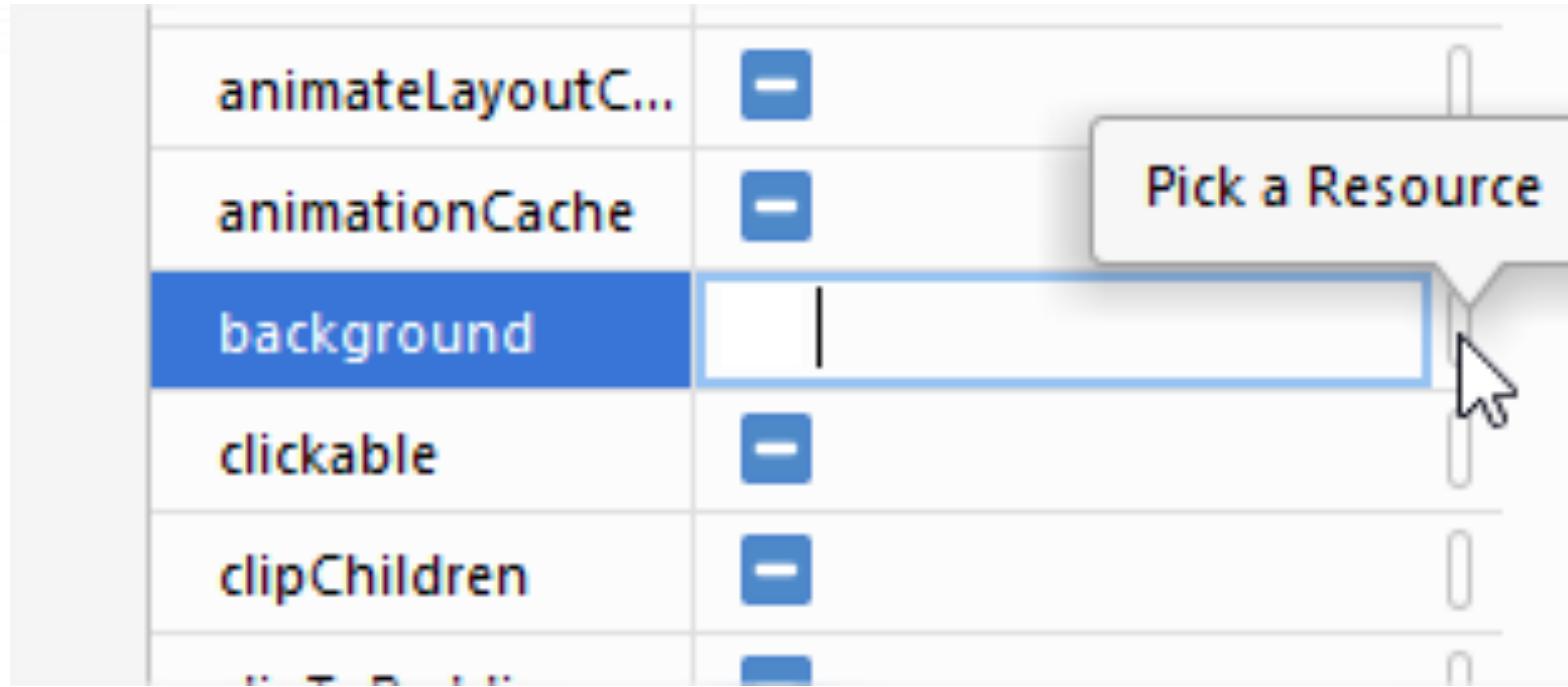
# App com o “ImageView”

- Imagens na pasta **drawable** após OK do **Copy**:



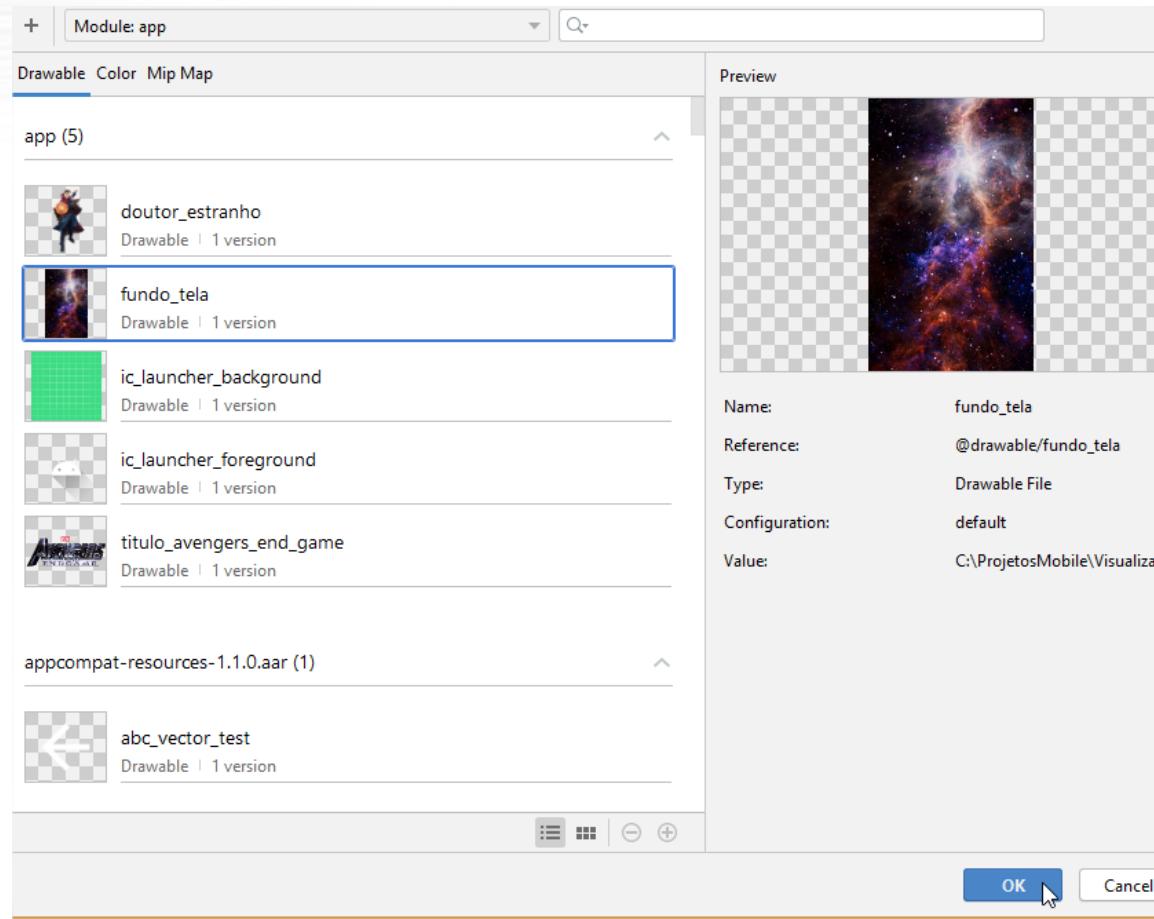
# App com o “ImageView”

- Voltando para o “Design” da tela dê um clique em qualquer parte da tela e na lista de atributos que aparece, clique em “***View all attributes***” e localize a propriedade “background”:



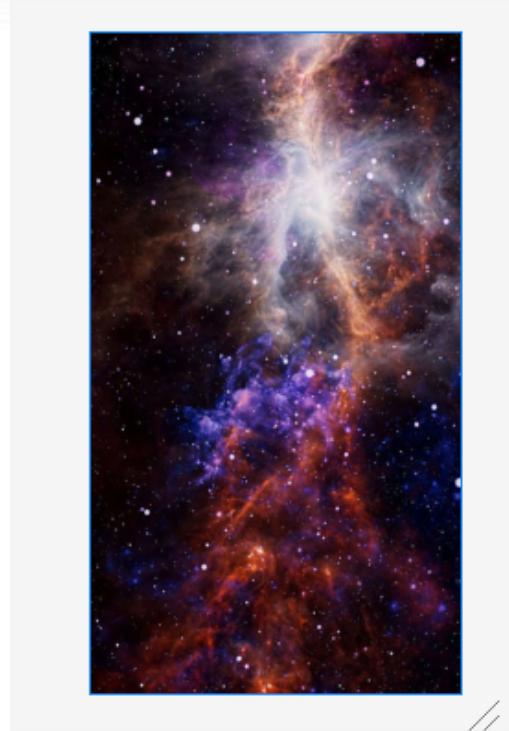
# App com o “ImageView”

- Selecione a imagem de **fundo\_tela** e clique em **OK**:



# App com o “ImageView”

- Sua tela deverá ficar assim:



- Observação: para adicionarmos uma imagem de fundo na tela da aplicação **NÃO É NECESSÁRIO**, neste caso, do componente *ImageView*.

# App com o “ImageView”

- Agora vamos arrastar para a tela um componente do tipo **ImageView** (seção “Common” ou “Widgets”), colocando-a no topo da tela:



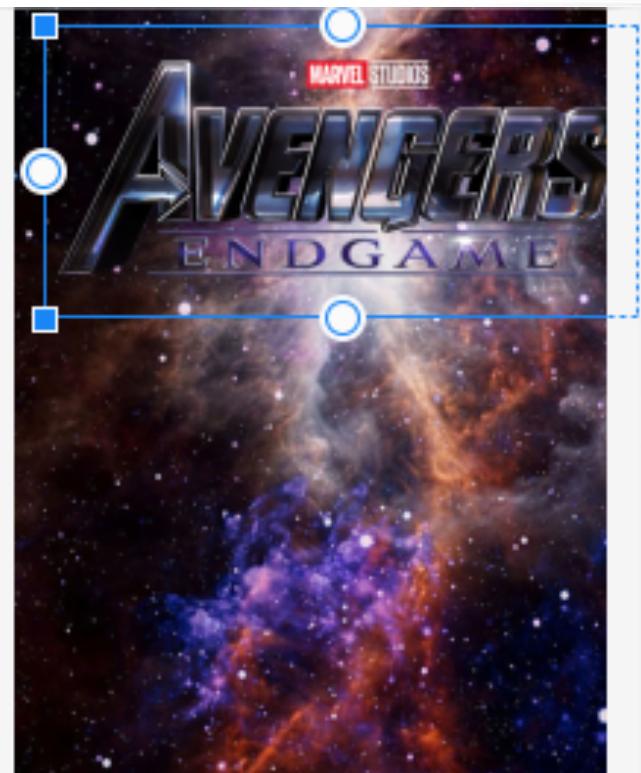
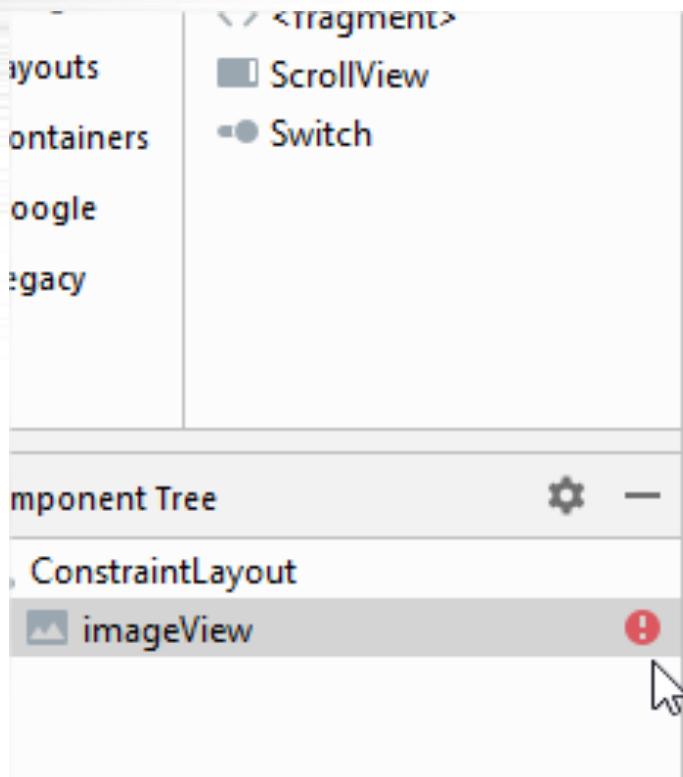
# App com o “ImageView”

- Em seguida selecione a imagem “**“titulo\_avengers\_end\_game”** e clique em **OK**:

|                |                                |
|----------------|--------------------------------|
| Name:          | titulo_avengers_end_game       |
| Reference:     | @drawable/titulo_avengers_...  |
| Type:          | Drawable File                  |
| Configuration: | default                        |
| Value:         | C:\ProjetosMobile\Visualiza... |

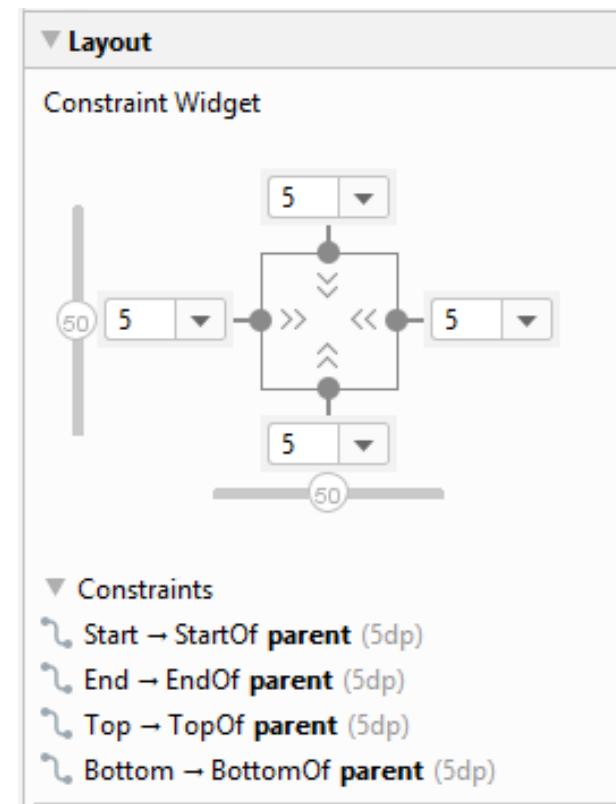
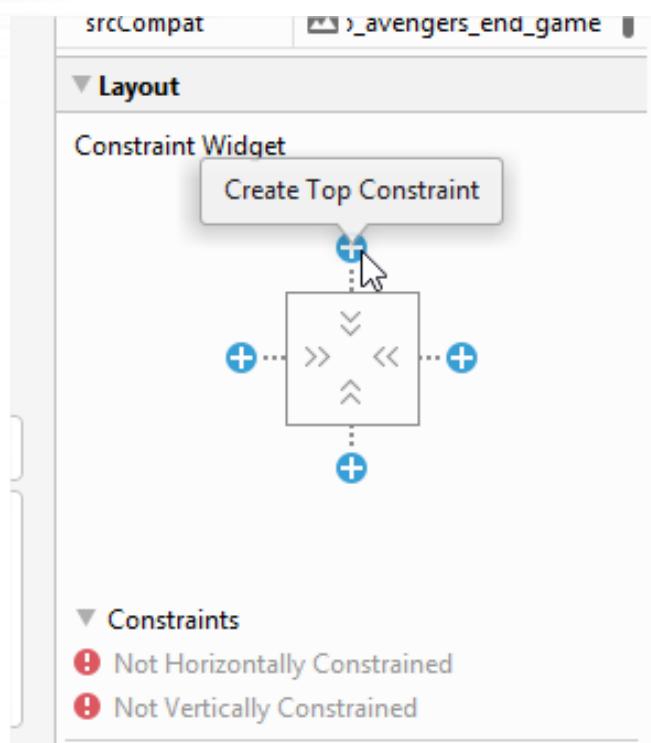
# App com o “ImageView”

- A sua tela ficará assim, vamos ajustar?



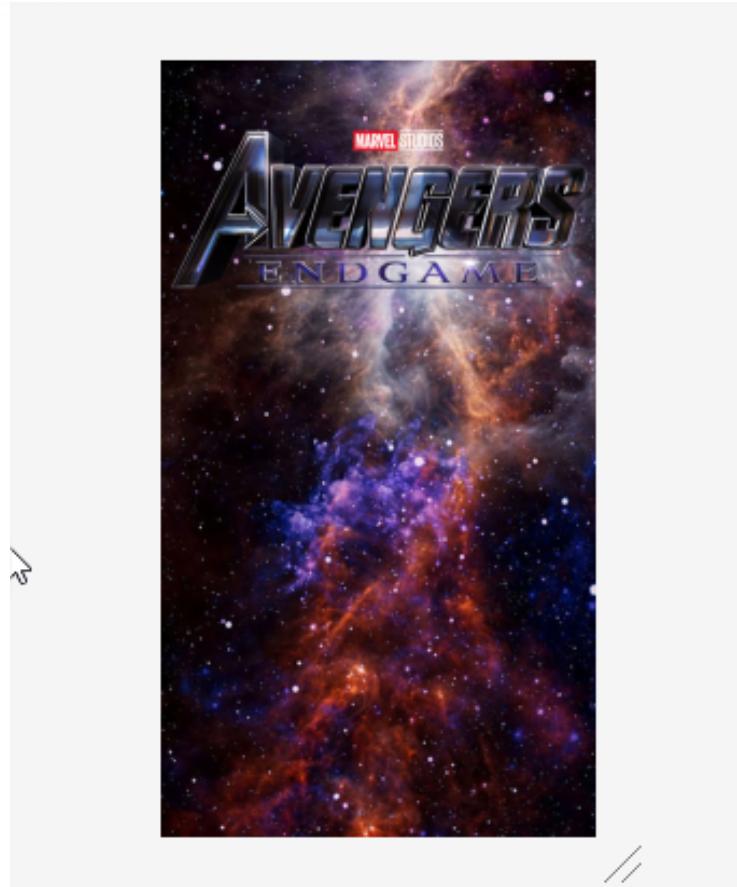
# App com o “ImageView”

- A imagem foi inserida, porém, serão necessários alguns ajustes em seu tamanho e alinhamento. Siga os passos abaixo (em **Layout** fazer as alterações):



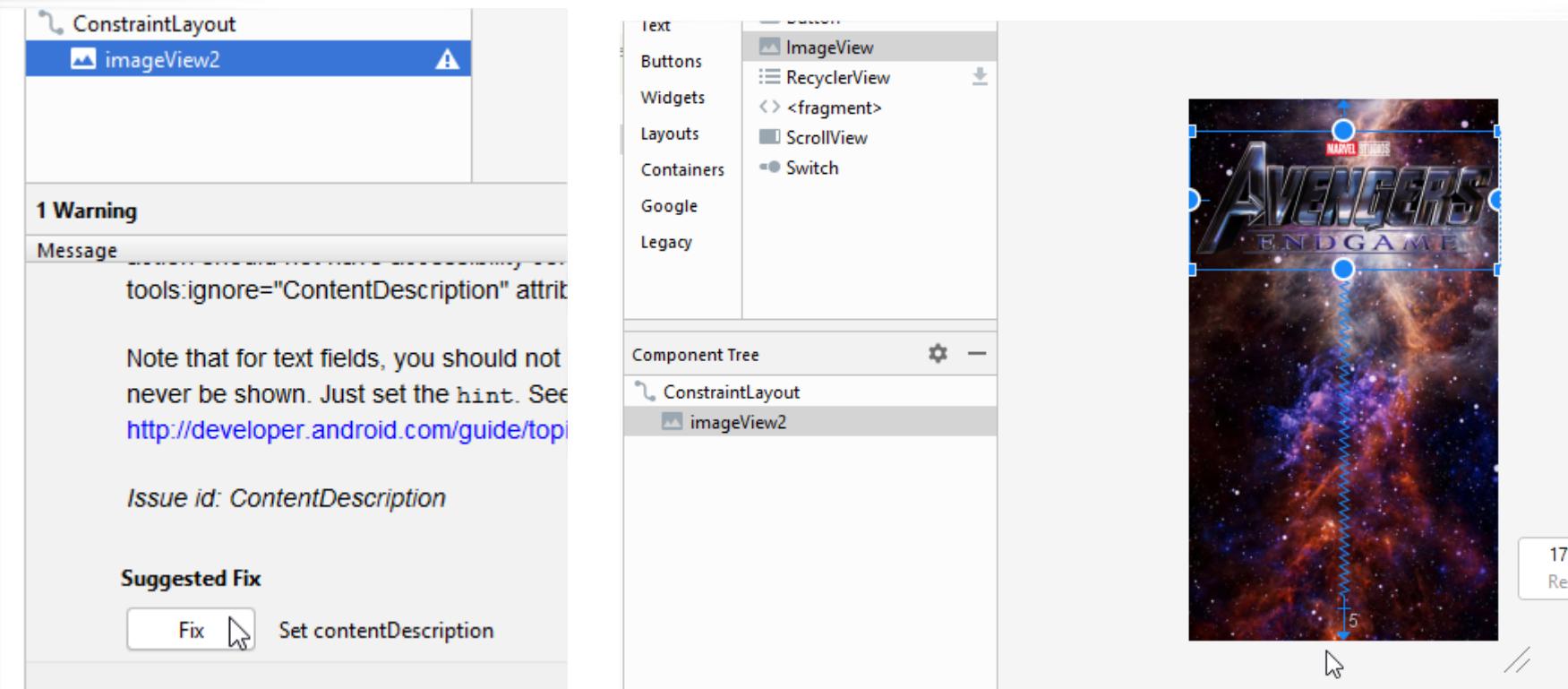
# App com o “ImageView”

- Se a sua imagem ficar ao centro, basta arrastá-la para cima, conforme abaixo:



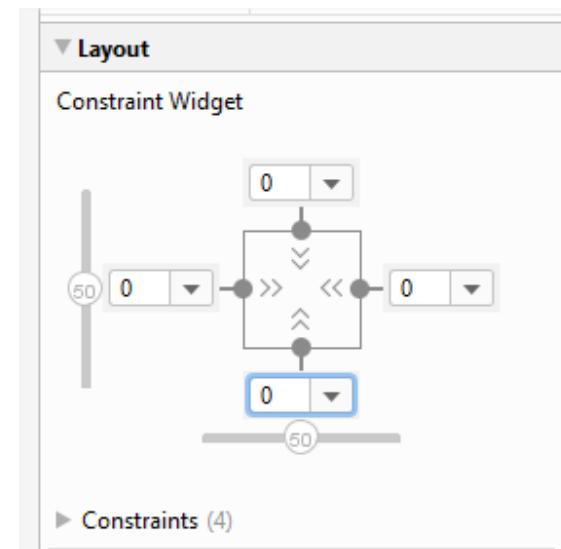
# App com o “ImageView”

- A advertência que aparece ao lado a imagem, basta aceitar a sugestão do Android Studio (**Fix**):



# App com o “ImageView”

- Agora vamos adicionar mais um componente **ImageView** logo abaixo do último componente que inserimos. Na caixa de diálogo que surge, selecione para este componente a imagem “**doutor\_estrano**” e clique em “**OK**”. Feito isso ajuste as configurações dos atributos e influências (conexões), de acordo como mostra a figura abaixo:



# App com o “ImageView”

- Versão final do App com três imagens:



# Conhecendo o “ImageSwitcher”



---

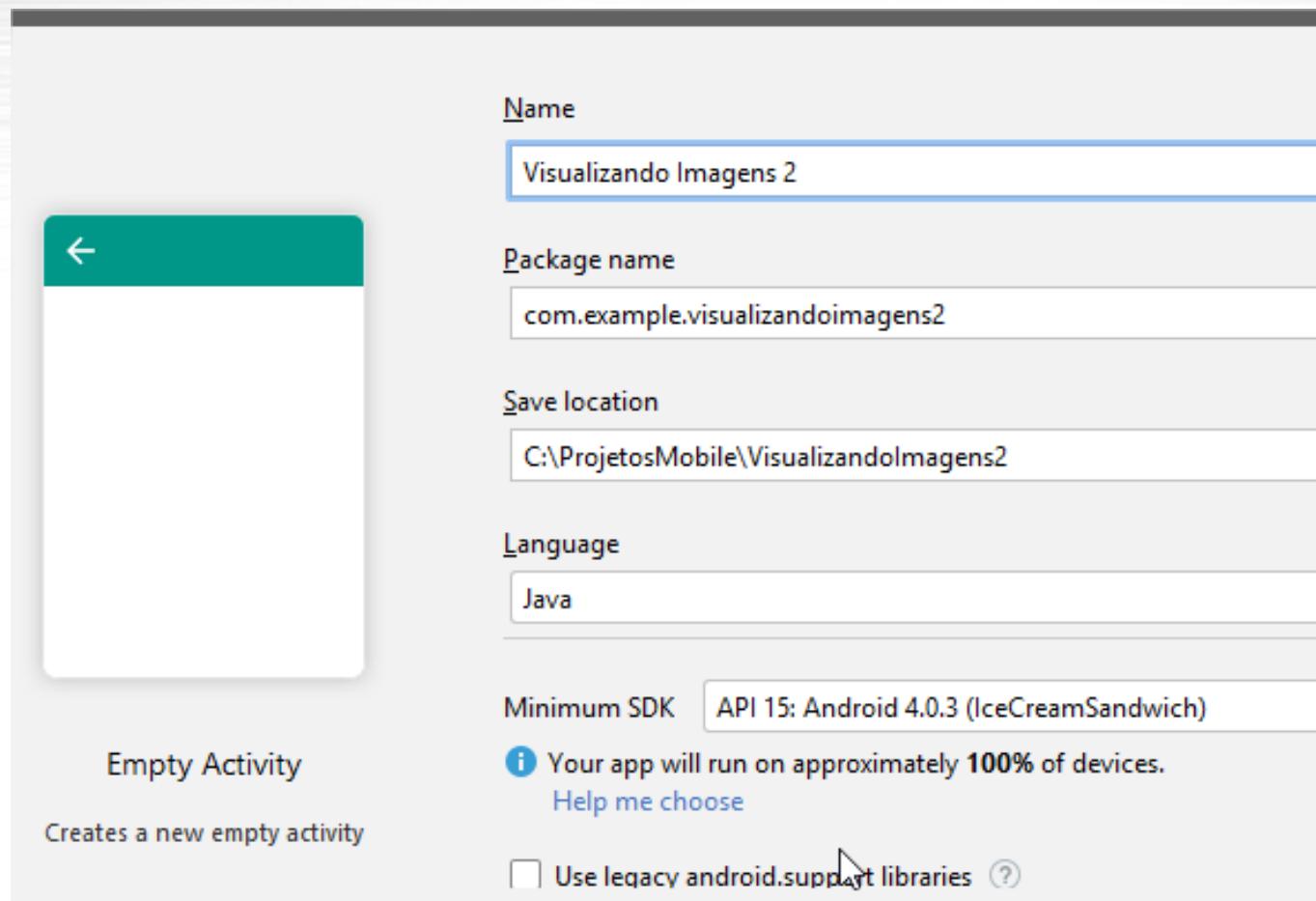


# App com o “ImageSwitcher”

- Agora iremos desenvolver um outra aplicação que visualiza imagens, só que um pouco diferente da que desenvolvemos no exercício anterior.
  - Essa aplicação que vamos criar vai utilizar um componente chamado **ImageSwitcher**, que também serve para visualizar imagens.
  - A diferença do **ImageSwticher** para o **ImageView** é a possibilidade de troca de imagens com algum efeito de transição .
  - Vamos demonstrar aqui o uso do **ImageSwitcher**, mas para isto digitem ou copiem do Blog o .XML da aplicação, pois o **ImageSwitcher** não está na Palheta do Android Studio.
-

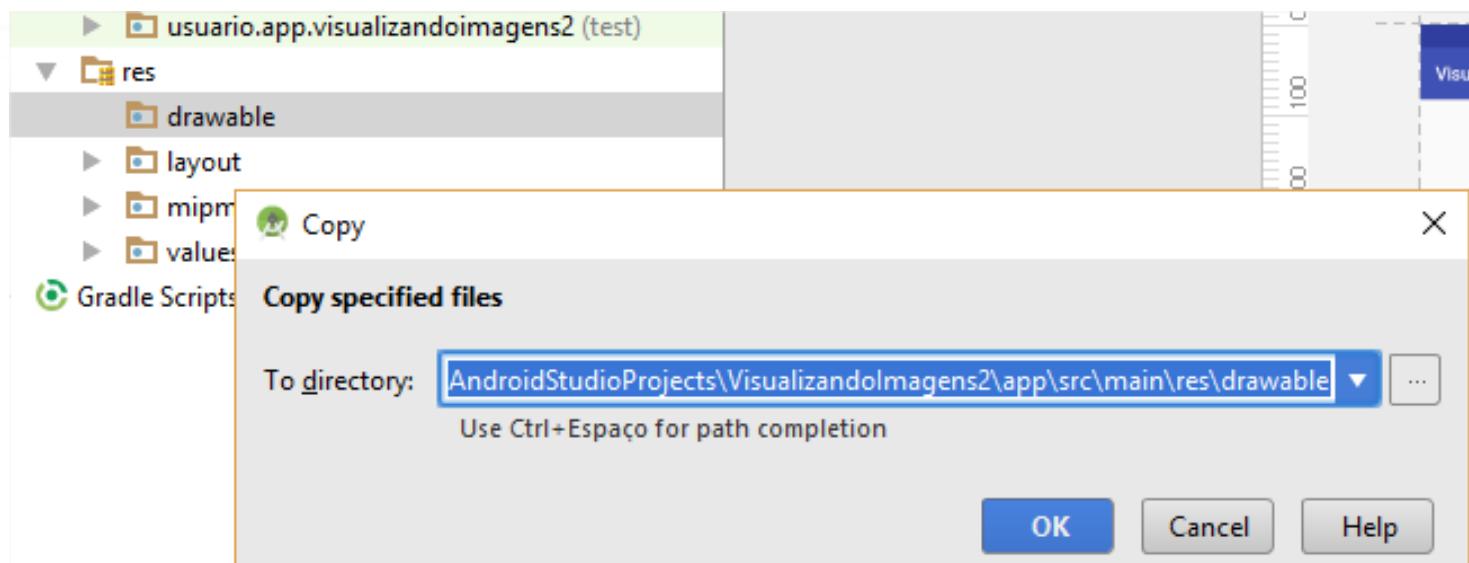
# App com o “ImageSwitcher”

- Crie um novo projeto de acordo os dados abaixo:



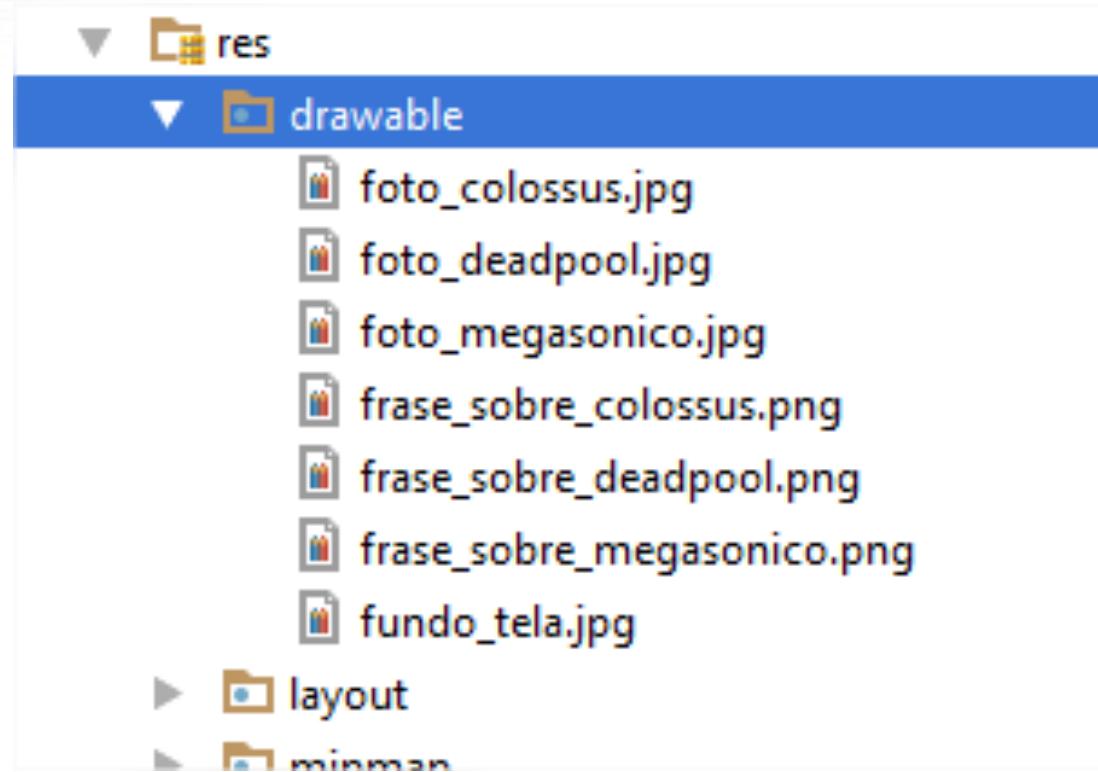
# App com o “ImageSwitcher”

- Depois de criado o projeto vamos em seguida “copiar” e “colar” para dentro do diretório “drawable” as imagens deste projeto (disponível no Blog).



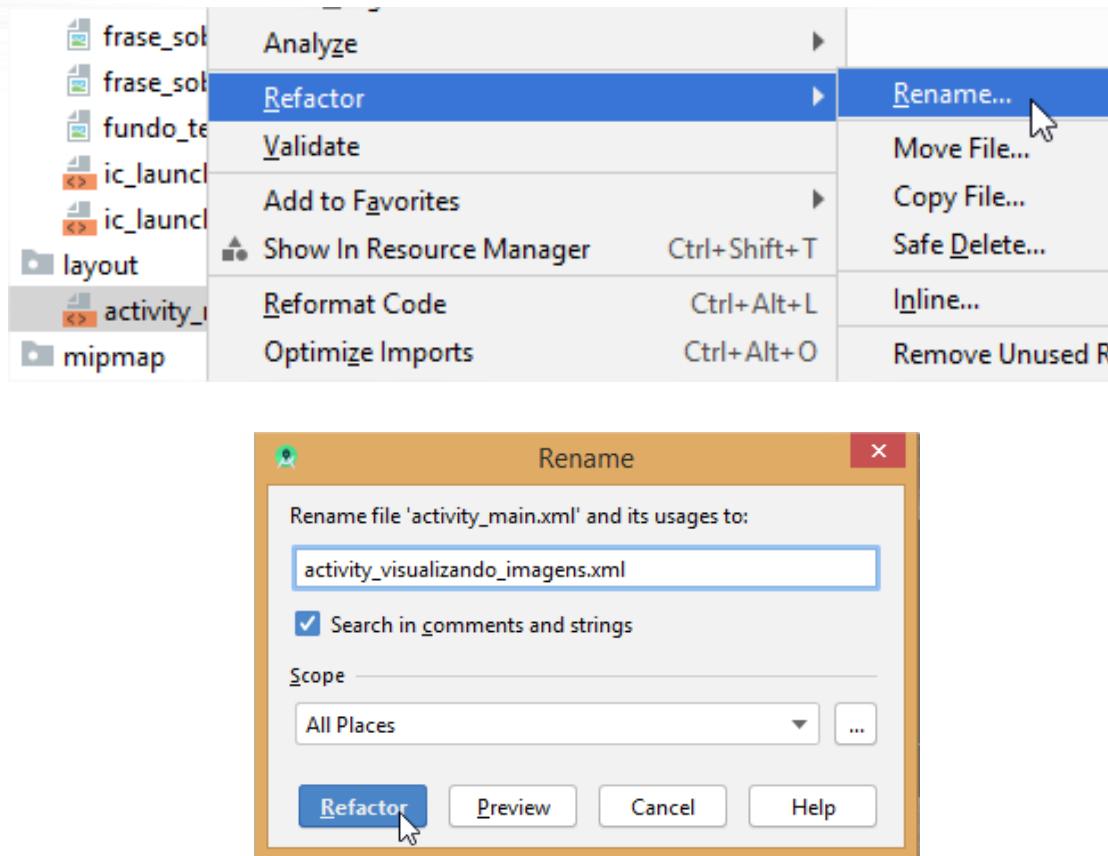
# App com o “ImageSwitcher”

- Após o OK as imagens estarão disponíveis conforme abaixo:



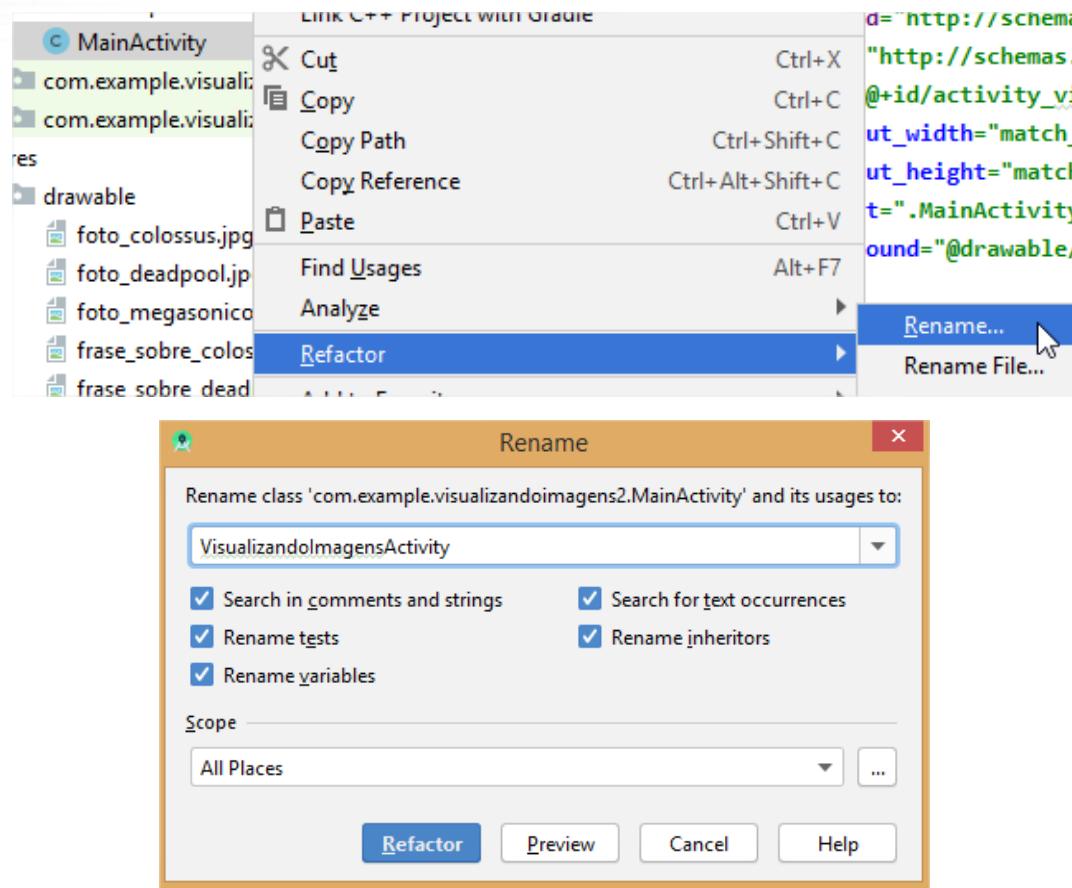
# App com o “ImageSwitcher”

- Vamos **renomear** o nosso .xml?
- Alterar para: ***activity\_visualizando\_imagens.xml***



# App com o “ImageSwitcher”

- Vamos **renomear** o nosso .java?
- Alterar para: ***VisualizandoImagensActivity.java***



# App com o “ImageSwitcher”

- Segue o código XML da tela da nossa aplicação:

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:id="@+id/activity_visualizando_imagens"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      tools:context=".VisualizandoImagensActivity"
9      tools:background="@drawable/fundo_tela">
10
11     <ImageSwitcher
12         android:layout_width="match_parent"
13         android:layout_marginTop="56dp"
14         android:layout_height="175dp"
15         android:layout_alignParentTop="true"
16         android:layout_alignParentLeft="true"
17         android:layout_alignParentStart="true"
18         android:id="@+id/imgFoto" />
```

# App com o “ImageSwitcher”

```
19
20    <ImageSwitcher
21        android:layout_width="match_parent"
22        android:id="@+id/imgSobre"
23        android:layout_height="120dp"
24        android:layout_below="@+id/imgFoto"
25        android:layout_alignParentLeft="true"
26        android:layout_alignParentStart="true" />
27
28    <LinearLayout
29        android:orientation="horizontal"
30        android:layout_width="match_parent"
31        android:layout_height="match_parent"
32        android:layout_alignParentBottom="true"
33        android:layout_alignParentLeft="true"
34        android:layout_alignParentStart="true"
35        android:layout_below="@+id/imgSobre"
36        android:id="@+id/layoutBotoes"
37        android:gravity="center">
38
```

# App com o “ImageSwitcher”

```
39 <Button  
40     android:id="@+id/btanterior"  
41     style="@android:style/Widget.Button"  
42     android:layout_width="wrap_content"  
43     android:layout_height="wrap_content"  
44     android:layout_weight="1"  
45     android:text="@string/anterior" />  
46  
47 <Button  
48     android:id="@+id/btproximo"  
49     style="@android:style/Widget.Button"  
50     android:layout_width="wrap_content"  
51     android:layout_height="wrap_content"  
52     android:layout_weight="1"  
53     android:text="@string/proximo" />  
54 </LinearLayout>  
55 </RelativeLayout>
```

# App com o “ImageSwitcher”

Palette

All Widgets Text Layouts Containers

OK Button  
ToggleButton  
CheckBox  
RadioButton  
CheckedTextView

3.2" HVGA slider (ADP1) 26 AppTheme

53% 0 100 200 300

BUTTON

Button

Component Tree

RelativeLayout

- imgFoto (ImageSwitcher)
- imgSobre (ImageSwitcher)

layoutBotoes (LinearLayout) (hc)

- btanterior (Button) - "Anterior"
- btproximo (Button) - "Próximo"

The screenshot shows the Android Studio interface with the Layout Editor open. The palette on the left has 'Widgets' selected, showing various UI components like Button, ToggleButton, CheckBox, RadioButton, and CheckedTextView. The main area displays a mobile application's layout. At the top is a title bar with the text 'Visualizando Imagens 2'. Below it is a movie poster for 'DEADPOOL'. At the bottom is a navigation bar with two buttons: 'ANTERIOR' on the left and 'PRÓXIMO' on the right. The 'Component Tree' panel on the left lists the components: a 'RelativeLayout' containing 'imgFoto' (ImageSwitcher), 'imgSobre' (ImageSwitcher), and a 'layoutBotoes' (LinearLayout) which contains 'btanterior' (Button) and 'btproximo' (Button). The 'btanterior' button is currently selected.

# App com o “ImageSwitcher”

- Vejamos o resultado:



Tela da aplicação em construção

# VisualizandolmagensActivity.java

Agora vamos modificar o arquivo “VisualizandolmagensActivity.java”. O código “completo” desse arquivo será como o código que é exibido abaixo e nos próximos slides. Na seção de importação de pacotes, vamos declarar as seguintes instruções destacadas a seguir:



```
package usuario.app.visualizandoimagens2;

import android.app.Activity;
import android.os.Bundle;
import android.widget.*;
import android.view.*;

public class VisualizandoImagensActivity extends
```

# VisualizandoImagensActivity.java

Agora declare os seguintes **objetos**, conforme destacada a próxima imagem:



```
public class VisualizandoImagensActivity extends Activity {  
    ImageSwitcher imgFoto, imgSobre;
```

# VisualizandolimagensActivity.java

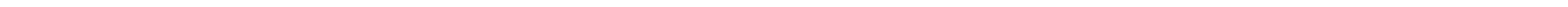
Dentro do método **onCreate** vamos adicionar as seguintes instruções destacadas em azul no código.

```
protected void onCreate(Bundle savedInstanceState) {  
  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_visualizando_imagens);  
  
    imgFoto = (ImageSwitcher) findViewById(R.id.imgFoto);  
    imgSobre = (ImageSwitcher) findViewById(R.id.imgSobre);  
  
    //Carrega a foto do dead pool  
    imgFoto.setImageResource(R.drawable.foto_deadpool);  
    //Carrega a info sobre o dead pool  
    imgSobre.setImageResource(R.drawable.frase_sobre_deadpool);  
}
```

# VisualizandolimagensActivity.java

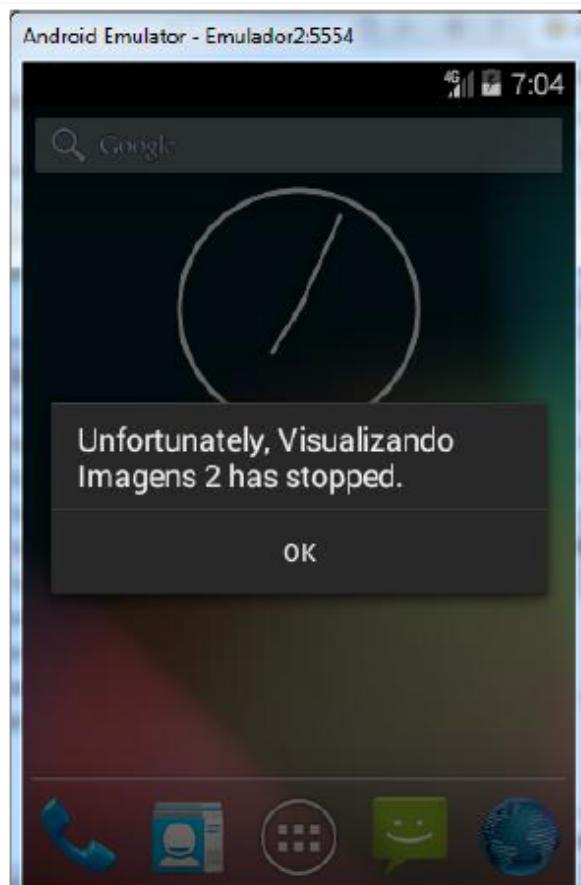
Entendendo as duas últimas linhas de código:

- Ambas carregam imagens já armazenadas em nosso projeto (uma referente a foto a ser exibida e outra sobre um informativo à respeito do personagem do filme), através do método **setImageResource**.



# VisualizandolmagensActivity.java

Vamos agora executar o nosso projeto, fazendo isso iremos nos deparar com o **seguinte ERRO:**



Ocorreu um erro durante a execução

# VisualizandolimagensActivity.java

Esse erro acontece pelo seguinte motivo:

- No **ImageSwitcher** precisamos, antes de carregar uma imagem (usando o **setImageResource**), definir uma **ViewFactory** (para cada objeto) usando o método **setFactory**.



# VisualizandolimagensActivity.java

Vamos digitar o seguinte trecho de código destacado em azul abaixo (após **imgFoto**):

```
imgFoto = (ImageSwitcher) findViewById(R.id.imgFoto);

imgFoto.setFactory(new ViewSwitcher.ViewFactory() {
    @Override
    public View makeView() {
        ImageView myView = new ImageView(getApplicationContext());
        myView.setScaleType(ImageView.ScaleType.FIT_XY);
        myView.setLayoutParams(new
            ImageSwitcher.LayoutParams(ActionBar.
                LayoutParams.WRAP_CONTENT,
            ActionBar.LayoutParams.WRAP_CONTENT));
        return myView;
    }
});
```

# VisualizandolimagensActivity.java

Vamos digitar o seguinte trecho de código destacado em azul abaixo (após **imgSobre**):

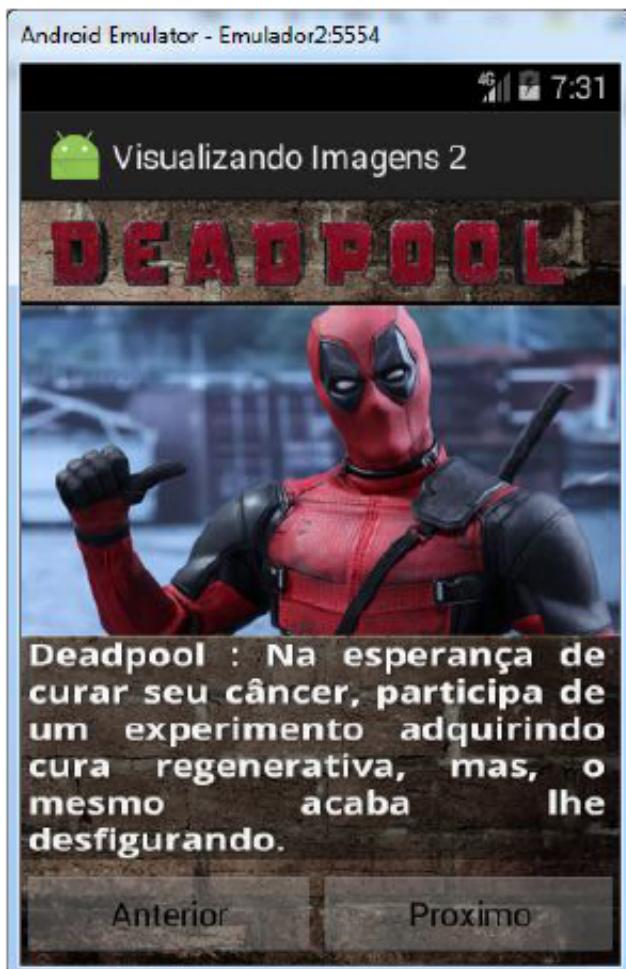
```
imgSobre = (ImageSwitcher) findViewById(R.id.imgSobre);

imgSobre.setFactory(new ViewSwitcher.ViewFactory() {
    @Override
    public View makeView() {
        ImageView myView = new ImageView(getApplicationContext());
        myView.setScaleType(ImageView.ScaleType.FIT_XY);
        myView.setLayoutParams(new
            ImageSwitcher.LayoutParams(ActionBar.
                LayoutParams.WRAP_CONTENT,
            ActionBar.LayoutParams.WRAP_CONTENT));
        return myView;
    }
});

//Carrega a foto do dead pool
imgFoto.setImageResource(R.drawable.foto_deadpool);
//Carrega a info sobre o dead pool
imgSobre.setImageResource(R.drawable.frase_sobre_deadpool);
```

# VisualizandolimagensActivity.java

Execute novamente a aplicação e confira o resultado:



Aplicação em execução

# VisualizandolimagensActivity.java

Dentro da nossa classe principal, adicione o seguinte trecho destacado a seguir na declaração de **atributos**: *btanterior e btproximo*



The screenshot shows a Java code editor with two tabs at the top: "activity\_visualizando\_imagens.xml" and "VisualizandoImagensActivity.java". The code in the editor is:

```
public class VisualizandoImagensActivity extends Activity {  
  
    ImageSwitcher imgFoto, imgSobre;  
  
    Button btanterior, btproximo;
```

A red rectangular box highlights the line "Button btanterior, btproximo;".

# VisualizandolmagensActivity.java

Agora vamos digitar o seguinte trecho de código destacado a seguir no método **onCreate**:

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_visualizando_imagens);  
  
    btanterior = (Button) findViewById(R.id.btanterior);  
    btproximo = (Button) findViewById(R.id.btproximo);  
  
    imgFoto = (ImageSwitcher) findViewById(R.id.imgFoto);  
  
    imgFoto.setFactory(new ViewSwitcher.ViewFactory() {
```

# VisualizandolimagensActivity.java

Digite agora o seguinte trecho de código destacado em azul a seguir (**btanterior**):

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    :
    //Carrega a foto do dead pool
    imgFoto.setImageResource(R.drawable.foto_deadpool);
    //Carrega a info sobre o dead pool
    imgSobre.setImageResource(R.drawable.frase_sobre_deadpool);

    btanterior.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {

            imgFoto.setImageResource(R.drawable.foto_deadpool);
            imgSobre.setImageResource(R.drawable.frase_sobre_deadpool);
        }
    });
}
```

# VisualizandolimagensActivity.java

Digite agora o seguinte trecho de código destacado em azul a seguir (**btproximo**):

```
btproximo.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        imgFoto.setImageResource(R.drawable.foto_colossus);
        imgSobre.setImageResource(R.drawable.frase_sobre_colossus);
    }
});
```

# VisualizandoImagensActivity.java

Execute novamente a aplicação e veja o resultado:



Aplicação em execução

# VisualizandolmagensActivity.java

Quando clicamos no botão “Anterior” e “Próximo” podemos ver informações à respeito dos dois personagens.

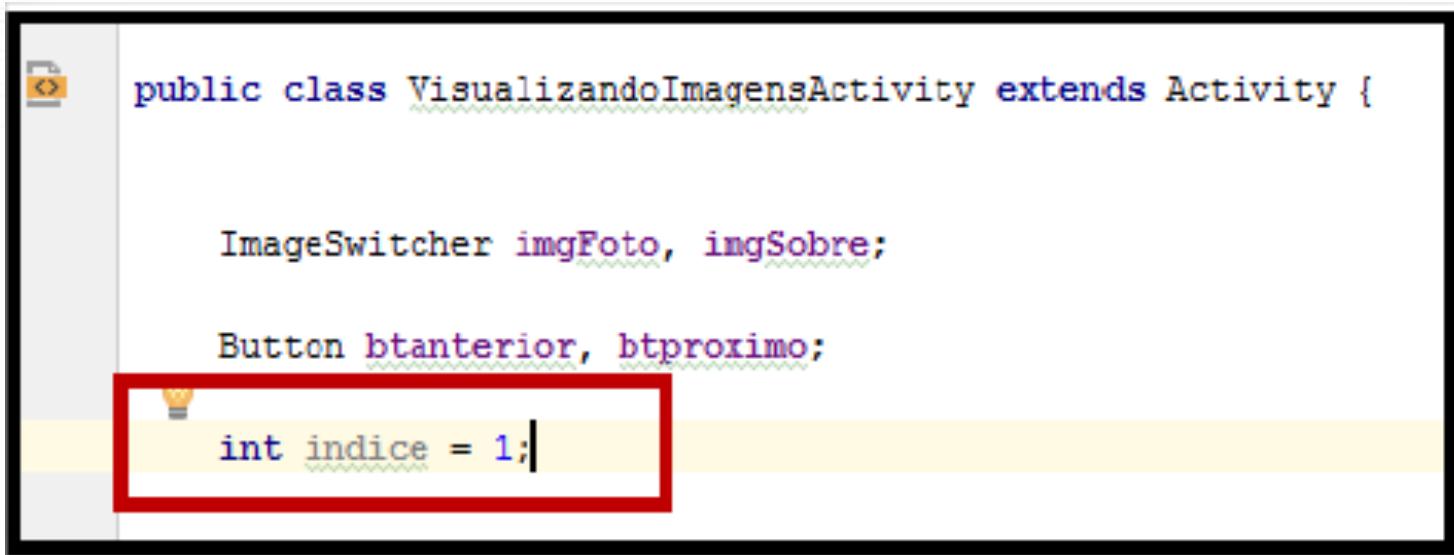
Mas, nesta aplicação iremos ver três personagens à respeito do filme “Dead Pool” (falta a “**Míssil Adolescente Megasônico**”).

Vamos ajustar nossa aplicação para que possamos visualizar sobre os três personagens.

---

# VisualizandoImagensActivity.java

Declare o seguinte atributo em nossa classe:



```
public class VisualizandoImagensActivity extends Activity {  
  
    ImageSwitcher imgFoto, imgSobre;  
  
    Button btanterior, btproximo;  
  
    int indice = 1;
```

# VisualizandolmagensActivity.java

Agora vamos criar o seguinte método dentro da nossa classe, conforme mostra o código a seguir: (chamado **mostrarInfoPersonagem**)

```
public void mostrarInfoPersonagem()
{
    switch (indice)
    {
        case 1:
        {
            imgFoto.setImageResource(R.drawable.foto_deadpool);
            imgSobre.setImageResource(R.drawable.frase_sobre_deadpool);

            }break;
        case 2:
        {
            imgFoto.setImageResource(R.drawable.foto_colossus);
            imgSobre.setImageResource(R.drawable.frase_sobre_colossus);

            }break;
        case 3:
        {
            imgFoto.setImageResource(R.drawable.foto_megasonico);
            imgSobre.setImageResource(R.drawable.frase_sobre_megasonico);

            }break;
    }
}
```

# VisualizandolimagensActivity.java

Vamos substituir o código dos botões , de acordo com o trecho de comandos destacado **em vermelho** a seguir (**btanterior**):

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    :  
    //Carrega a foto do dead pool  
    imgFoto.setImageResource(R.drawable.foto_deadpool);  
    //Carrega a info sobre o dead pool  
    imgSobre.setImageResource(R.drawable.frase_sobre_deadpool);  
  
    btanterior.setOnClickListener(new View.OnClickListener() {  
        @Override  
        public void onClick(View view) {  
            if(indice > 1)  
            {  
                indice--;  
                mostrarInfoPersonagem();  
            }  
        }  
    });
```

# VisualizandolimagensActivity.java

Vamos substituir o código dos botões , de acordo com o trecho de comandos destacado **em vermelho** a seguir (**btproximo**):

```
btproximo.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if(indice < 3)
        {
            indice++;
            mostrarInfoPersonagem();
        }
    }
});
```

}

# VisualizandoImagensActivity.java

Execute a aplicação novamente e veja o resultado:



Aplicação em execução

# VisualizandolimagensActivity.java

Usando efeitos de transição:

- Quando as informações são trocadas durante a sua exibição na aplicação, não ocorre nenhum efeito de transição. Vamos inserir um efeito de transição durante a troca das informações no **ImageSwitcher**.
- Dentro do método **onCreate** acrescentem as linhas de códigos destacadas em azul, conforme podemos ver em seguida:

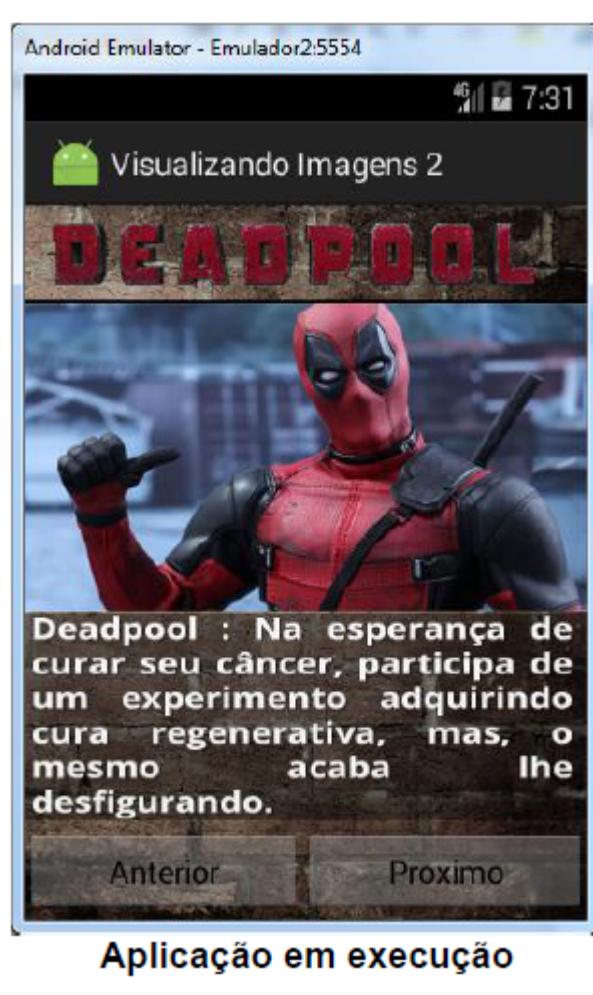


# VisualizandolmagensActivity.java

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_visualizando_imagens);  
  
    //Carrega o objeto de animação de entrada da imagem  
    Animation in = AnimationUtils.loadAnimation  
        (this, android.R.anim.slide_in_left);  
  
    //Carrega o objeto de animação de saída de imagem  
    Animation out = AnimationUtils.loadAnimation  
        (this, android.R.anim.slide_out_right);  
    :  
    //Carrega a foto do dead pool  
    imgFoto.setImageResource(R.drawable.foto_deadpool);  
    imgFoto.setInAnimation(in);  
    imgFoto.setOutAnimation(out);  
  
    //Carrega a info sobre o dead pool  
    imgSobre.setImageResource(R.drawable.frase_sobre_deadpool);  
    imgSobre.setInAnimation(in);  
    imgSobre.setOutAnimation(out);  
  
}
```

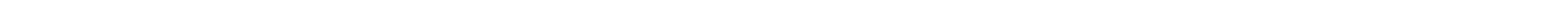
# VisualizandolimagensActivity.java

Depois de digitado o código, execute novamente a aplicação e confira os resultados.



# VisualizandolimagensActivity.java

Se a gente observar agora, existe uma transição (animação) na hora de exibir as imagens (um deslocamento quando uma imagem sai e quando outra entra, seguindo de um efeito do tipo **fade in/out**).



# VisualizandolmagensActivity.java

Vamos entender o código. As instruções a seguir:

```
Animation in = AnimationUtils.LoadAnimation  
(this, android.R.anim.slide_in_left);
```

```
Animation out = AnimationUtils.LoadAnimation  
(this, android.R.anim.slide_out_right);
```

Carrega os objetos responsáveis por realizar as animações de transição entre as imagens. A classe responsável por isso é a classe **Animation**.

---

# VisualizandolmagensActivity.java

Nas instruções a seguir:

```
imgFoto.setInAnimation(in);  
imgFoto.setOutAnimation(out);
```

Carregamos as animações de entrada (**setInAnimation**) e saída (**setOutAnimation**) armazenados no objeto “in” e “out” para a imagem. Cada imagem que irá possuir uma animação deverá efetuar a chamada desses dois métodos.

# Exercício

- Acrescentar mais uma imagem ao APP Visualizando Imagens 2, ou seja, o APP vai mostrar 4 imagens e suas frases de descrições dos personagens, conforme abaixo:





**Por hoje é só !!!!**

Até a próxima aula...

