Usando a câmera

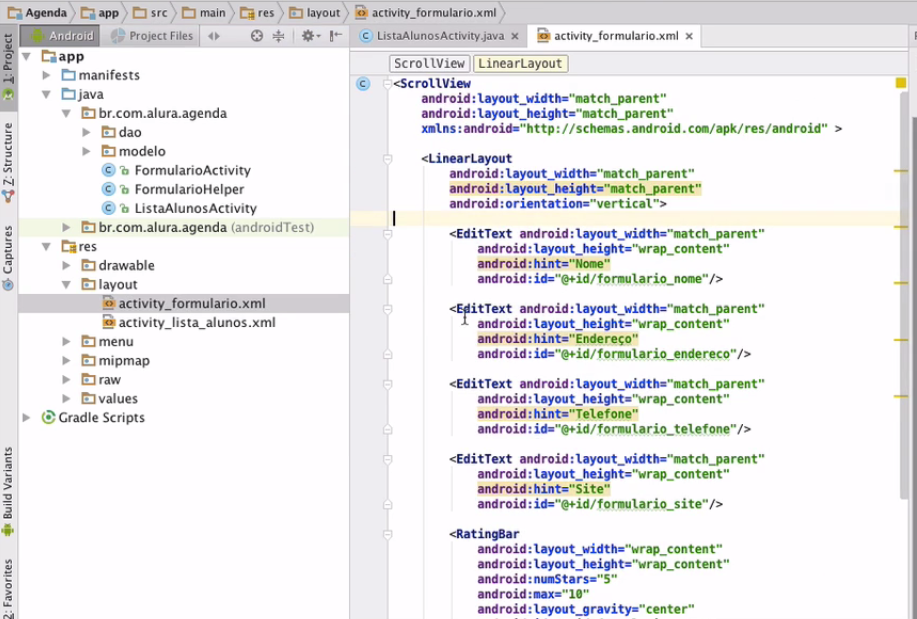
Na aula passada, nós adicionamos novos itens no menu de contexto. Agora nós podemos incrementar ainda mais a nossa agenda.

Pensando em exemplos do Whatsapp ou Gmail, quando temos uma lista de contato, encontramos uma foto ao lado do nome. No nosso aplicativo, iremos substituir o id por uma imagem. Assim, facilitaremos a identificação dos contatos, além de deixar o app mais interessante visualmente.

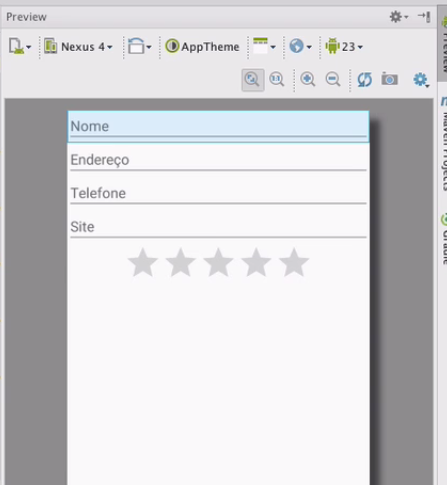
Para que isto aconteça, seria necessário que ao cadastrarmos um novo aluno, tivéssemos um campo para foto e pudéssemos acessar a câmera.

Como vamos modificar o layout da tela, teremos que modificar o xml, que criamos para cada Activity.

Vamos acessar a pasta de *layout*, e abrir o arquivo activity\_formulario.xml.



À direita, encontraremos o *Preview*.



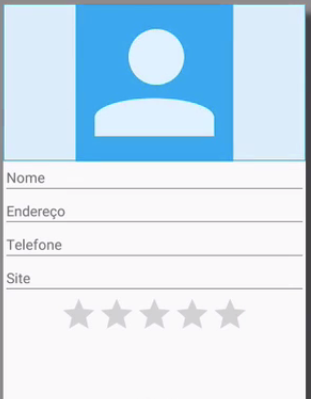
Queremos adicionar a foto do aluno no formulário. Iremos adicionar ImageView abaixo do LinearLayout e antes do primeiro EditText.

<ImageView  
 android:layout\_width="match\_parent"  
 android:layout\_height="200dp" android:src="@drawable/person"/>

Especificamos a largura como match\_parent (para ocupar a tela toda) e na altura, a quantidade de dp.

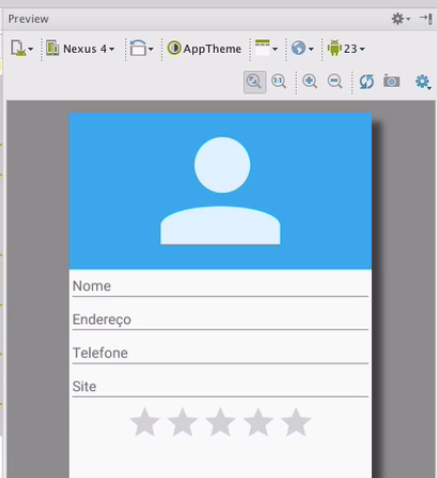
Aproveitamos o arquivo que acompanhava o material do curso, disponibilizado para baixar.

Nosso app ficou assim:



Vamos mudar a cor do fundo que aparece nas laterais da imagem, usando o atributo background e a cor #00A8EC.

<ImageView  
 android:layout\_width="match\_parent"  
 android:layout\_height="200dp" android:src="@drawable/person"/>  
 android:backgroud="#00A8EC"



Agora, vamos possibilitar o acesso da câmera, utilizando a ideia de floating action button. Iremos inserir um botão de foto no canto da imagem.

No código, usaremos Button e depois, especificaremos o tamanho e aparência. No botão, vamos usar uma imagem vetorial de fundo, para não haver alterações nos diferentes dispositivos. A imagem está na pasta drawble e o arquivo será fundo.

<Button  
 android:layout\_width="56dp"  
 android:layout\_height"56dp"  
 android:background="@drawable/fundo"/>

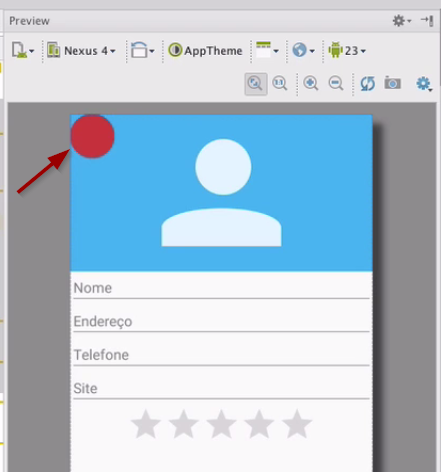
Queremos que o botão fique sobre a imagem, iremos incluir RelativeLayout dentro do LinearLayout.

<Button  
 android:layout\_width="56dp"  
 android:layout\_height"56dp"  
 android:background="@drawable/fundo"/>  
  
 <RelativeLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 </RelativeLayout>

Depois, iremos mover o ImageView e Button para dentro do RelativeLayout.

<RelativeLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 <ImageView  
 android:layout\_width="match\_parent"  
 android:layout\_height="200dp"  
 android:src="@drawable/person"/>  
 android:backgroud="#00A8EC"  
  
 <Button  
 android:layout\_width="56dp"  
 android:layout\_height"56dp"  
 android:background="@drawable/fundo"/>  
  
  
</RelativeLayout>

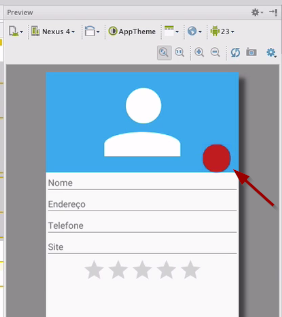
Agora, o botão já está sobreposto à imagem.



Mas ainda iremos alterar a posição. Adicionaremos alignParentBotton, dentro do button e dizer que é true. Com isto, o botão irá para baixo. Iremos movê-lo para direita, se inserirmos alignParentRight.

<Button  
 android:layout\_width="56dp"  
 android:layout\_height"56dp"  
 android:background="@drawable/fundo"  
 android:layout\_algnParentBottom="true"  
 android:layout\_alignParentRight="true"/>

Iremos adicionar uma borda de 16dp para afastar o botão do canto direito.



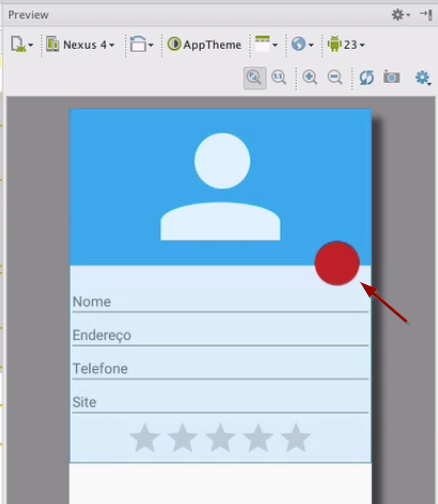
Se quiséssemos que o botão flutuasse uma parte fora do campo da imagem, poderíamos tentar adicionar novos valores de margem. Mas vale lembrar, que valores negativos como -32dp, não seriam aceitos.

O Android Studio não irá reposicionar o botão, porque a posição do button está limitada ao tamanho do RelativeLayout. Então, iremos aumentar o tamanho dele e manter fixo o tamanho da imagem.

Sabendo que a imagem tem 200 dp de altura, vamos aumentar a altura do RelativeLayout para 225 dp.

<RelativeLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="225dp">

Agora, o botão ficará flutuando parcialmente para fora da imagem.



Esta nova posição, fica mais visível para o usuário.

Agora, falta inserirmos a imagem de uma câmera dentro do botão. Dentro da pasta drawable, temos o arquivo ic\_camera\_png. Vamos fazer uma cópia e nomeá-la como fundo\_camera.xml. Iremos adicionar uma camada com a imagem da câmera sobre o fundo vermelho.

O código do arquivo ficará assim:

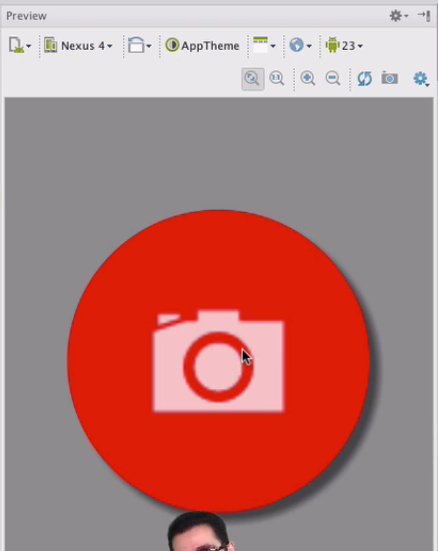
<?xml version="1.0" encoding="utf-8"?>  
 <layer-list xmlns:android="http://schemas.android.com/apk/res/android">  
 <item>  
 <shape  
 android:shape:"oval">  
 <solid android:color="#ff0000"/>  
 </shape>  
 </item>  
 <item android:drawble="@drawable/ic\_camera"/>  
</layer-list>

Observe que tívemos que criar um item para camada. O Android irá combinar as duas camadas, e irá transformá-las em uma única imagem.

Iremos adicionar também a tag padding para que a imagem da câmera não ocupe todo o fundo vermelho.

<?xml version="1.0" encoding="utf-8"?>  
 <layer-list xmlns:android="http://schemas.android.com/apk/res/android">  
 <item>  
 <shape  
 android:shape:"oval">  
 <solid android:color="#ff0000"/>  
 <padding android:top="10dp"  
 android:bottom="10dp"  
 android:left="10dp"  
 android:right="10dp"  
 </shape>  
 </item>  
 <item android:drawble="@drawable/ic\_camera"/>  
</layer-list>

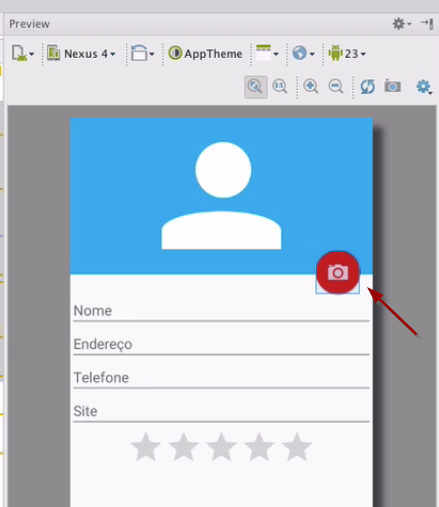
Quanto maior for o valor do padding, menos espaço sobrará preenchê-lo com texto ou imagem. Tivemos que específicar todas as direções.



Para finalizar, retornaremos ao arquivo activity\_formulario.xml.

No código, iremos substituir adicionar fundo\_camera dentro do Button, alterando o background.

<Button  
 android:layout\_width="56dp"  
 android:layout\_height"56dp"  
 android:background="@drawable/fundo\_camera"  
 android:layout\_alignParentBottom="true"  
 android:layout\_alignParentRight="true"  
 android:layout\_marginRight="16dp"/>



Em seguida, precisaremos referenciar a imagem e o botão. Nós usaremos ids para identificar os dois componentes.

<ImageView  
 android:id:"@+id/formulario\_foto"  
 \...

Vamos usar o nome do activity seguido pelo nome do componente.

Criaremos também o formulario\_botao\_foto.

<Button  
 android:id="@+id/formulario\_botao\_foto"

Nosso código está assim:

<ImageView  
 android:id:"@+id/formulario\_foto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_height="200dp" android:src="@drawable/person"/>  
 android:backgroud="#00A8EC"  
  
<Button  
 android:id="@+id/formulario\_botao\_foto"  
 android:layout\_width="56dp"  
 android:layout\_height"56dp"  
 android:background="@drawable/fundo\_camera"  
 android:layout\_alignParentBottom="true"  
 android:layout\_alignParentRight="true"  
 android:layout\_marginRight="16dp"/>

Em seguida, iremos criar o funcionamento do botão.