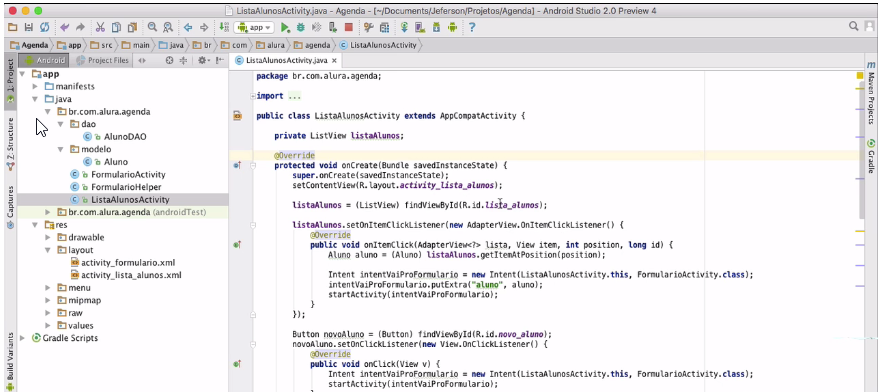
Melhorando o visual da lista de alunos

Na última aula, introduzimos a câmera e fizemos alterações no aplicativo. Agora, as imagens podem ser salvas.

Iremos voltar ao problema inicial, que era adicionar uma foto ao lado do menu com os contatos do nosso aplicativo.

Iremos procurar no código, o responsável por popular a nossa lista, de adicionar o comportamento de cadastrar um novo aluno.



A ListaAlunosActivity era responsável por popular a nossa lista. Com ela adicionamos comportamentos e novos alunos.

Para revisarmos o que havíamos feito, tínhamos colocado no onCreate, mas tínhamos o problema de que quando cadastrávamos um aluno e depois, voltávamos para o formulário, nós não atualizávamos a nossa lista. Então, para termos um ponto único que sempre funcionasse, inserimos o método para carregá-la dentro do onResume.

private void carregaLista() {  
 AlunoDAO dao = new AlunoDAO(this);  
 List<Aluno> alunos = dao.buscaAluno();  
 dao.close();  
  
 ArrayAdapter<Aluno> = new ArrayAdapter<ALuno>(this, android.R.layout.simple\_list\_item\_1, alunos);  
  
}   
  
@Override  
public void onCreateMenu(ContextMenu menu, View v, final ContextMenu.ContextMenuInfo menuInfo) {  
 AdapterView.AdapterContextMenuInfo info = (AdapterView.AdapterContextMenuInfo) menuInfo;  
 final Aluno aluno = (Aluno) listaAlunos.getItemAtPosition(info.position);  
//...

O responsável pelas *views* que vão entrar no nosso Listener é o Adapter. Ao criá-lo, especificávamos alguns parâmetros: contexto, layout e o *array* de alunos. Então, estávamos passando a lista de alunos para o Adapter.

ArrayAdapter<Aluno> = new ArrayAdapter<ALuno>(this, android.R.layout.simple\_list\_item\_1, alunos);

Depois que montamos o adapter, iremos mudar o layout(android.R.layout.simple\_list\_item\_1). Para isto, iremos entrar na pasta res, depois clicaremos com o botão direito em layout e selecionaremos Layout resource file. Vamos alterar o nome do arquivo .xml para list\_item e vou mudar para o modo texto.

O objetivo é que o nosso aplicativo tenha uma foto e o texto, com nome e telefone. Iremos alterar a orientação para horizontal e adicionaremos um ImageView e definir as dimensões. Usaremos a mesma imagem do formulário. Será interessante adicionarmos também um id.

<?xml version="1.0"  
encoding="uft-8"?>  
<LinearLayout xmlns:android='http://schemas.android.com/apk/res/android"  
 android:orientation="horizontal"  
 android:layout\_widht="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <ImageView   
 android:layout\_width="64dp"  
 android:layout\_height="64dp"  
 android=src="@drawable/person"  
 android:id="@+id/item\_foto"/>  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/item\_nome"  
 android:text="Nome"  
 android:textSize=20sp"  
 android:textStyle="bold"/>  
  
</LinearLayout>

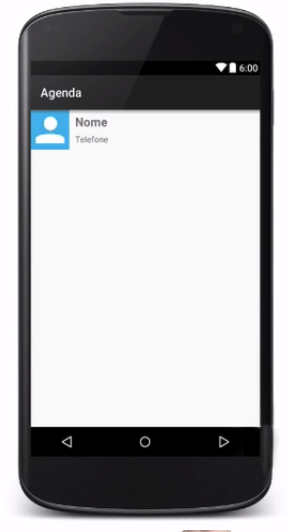
Adicionamos um espaçamento com paddingLeft para o texto Nome não ficar tão próximo à imagem e ter um espaçamento à esquerda. Após incluirmos o TextView, criaremos o trecho referente ao telefone. Mas se incluirmos tudo dentro do mesmo LinearLayout, o Nome e Telefone ficaram lado a lado.

Teremos que criar um novo LinearLayout com os dois e mudar a orientação.

<?xml version="1.0"  
encoding="uft-8"?>  
<LinearLayout xmlns:android='http://schemas.android.com/apk/res/android"  
 android:orientation="horizontal"  
 android:layout\_widht="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <ImageView   
 android:layout\_width="64dp"  
 android:layout\_height="64dp"  
 android=src="@drawable/person"  
 android:id="@+id/item\_foto"/>  
  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientatio="vertical">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/item\_nome"  
 android:text="Nome"  
 android:textSize=20sp"  
 android:textStyle="bold"  
 android:paddingLeft="10dp"/>  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/item\_telefone"  
 android:text="Telefone"  
 android:textSize=20sp"  
 android:textStyle="bold"  
 android:paddingLeft="10dp"  
 android:paddingTop="8dp"/>  
  
 </LinearLayout>   
  
</LinearLayout>

Mas se alterarmos o layout\_height para match\_parent do segundo LinearLayout, o layout ficará desorganizado. Para isto não acontecer, teremos que alterar o primeiro layout\_height para match\_parent, que agora, terá o mesmo tamanho do conteúdo do ImageView e também iremos centralizar o texto, adicionando gravity="center\_vertical".

<?xml version="1.0"  
encoding="uft-8"?>  
<LinearLayout xmlns:android='http://schemas.android.com/apk/res/android"  
 android:orientation="horizontal"  
 android:layout\_widht="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 <ImageView   
 android:layout\_width="64dp"  
 android:layout\_height="64dp"  
 android=src="@drawable/person"  
 android:id="@+id/item\_foto"/>  
  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientatio="vertical"  
 android:gravity="center\_vertical">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/item\_nome"  
 android:text="Nome"  
 android:textSize=20sp"  
 android:textStyle="bold"  
 android:paddingLeft="10dp"/>  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/item\_telefone"  
 android:text="Telefone"  
 android:textSize=20sp"  
 android:textStyle="bold"  
 android:paddingLeft="10dp"  
 android:paddingTop="8dp"/>  
  
 </LinearLayout>   
  
</LinearLayout>



Agora, que já criamos o arquivo XML, iremos em seguida colocar tudo dentro do Adapter.