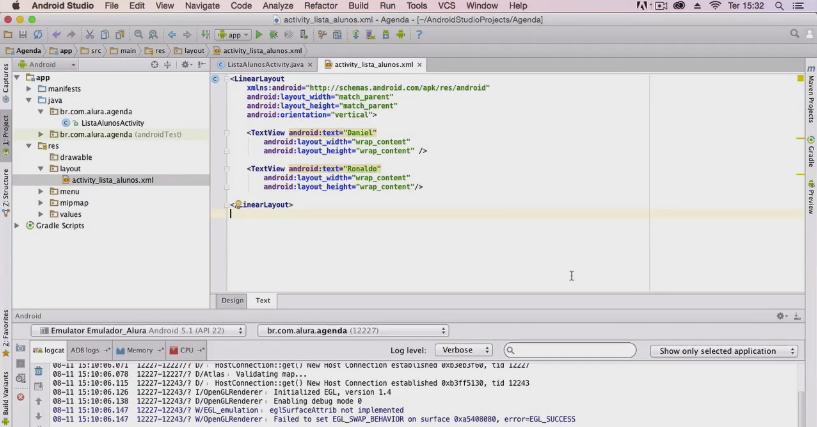
## **Tratando eventos de clique**

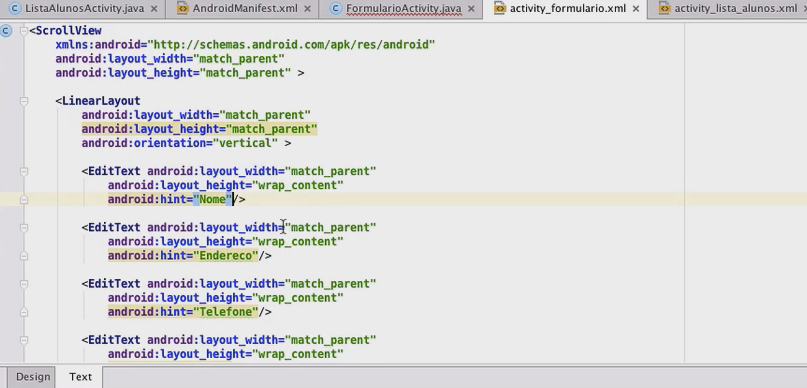
Nosso formulário já tem campos para preencher, estrelinhas para dar notas, um botão de salvar e a possibilidade de rolar a tela se necessário. O próximo passo é adicionar ao botão de "Salvar" uma reação ao ser pressionado. Para modificar o comportamento dele vamos na classe *java* na activity, isto é, voltamos na aba FormularioActivity.javae encontramos o seguinte:



Primeiro, é necessário criar o botão para depois construir o *feedback* (a reação). O setContentView da activity pega o 'layout' e gera as instâncias que estão na .xml. Para trazer as instâncias podemos usar o findViewById e após o setContentViewdamos um "Enter" e acrescentamos o Id:

@0verride  
 protected void onCreate(Bundle savedInstanceState) {   
 super.onCreate (savedInstanceState);  
 setContentView(R.layout.activity\_formularios);  
  
 findViewById()  
  
 }

Vamos aproveitar o momento e voltar na activity\_formulario.xml onde já adicionamos um Id em cada um dos campos, uma vez que precisaremos deles mais adiante. Retornamos a aba activity\_formulario.xml e a encontramos assim:



Vamos começar pelo primeiro EditText cujo campo é "Nome". Acrescentamos nele @+id/ e para organizá-lo melhor adicionamos onde ele está, no formulario\_nome:

android:id="@+id/formulário\_nome"

Ficamos com o EditText da seguinte maneira:

<EditText android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content”   
 android:hint="Nome"   
 android:id="@+id/formulario\_nome" />

Esse procedimento é repetido com todos os demais campos inclusive com a RatingBare com o Button. Ficaremos com:

<EditText android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content”   
 android:hint="Endereco"   
 android:id="@+id/formulario\_endereco"/>  
  
<EditText android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content”   
 android:hint="Telefone"   
 android:id="@+id/formulario\_telefone"/>  
  
<EditText android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content”   
 android:hint="Telefone"   
 android:id="@+id/formulario\_site"/>  
  
<RatingBar   
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"   
 android:layout\_gravity="center"   
 android:max="10“   
 android:numStars="5"   
 android:id="@+id/formulario\_nota"/>  
  
<Button   
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Salvar"/>  
 android:id="@+id/formulario\_salvar" />

Agora, todos os componentes estão identificados!

Vamos voltar para a aba FormulárioActivity e digitar, para buscar o botão, o findViewById(R.id.formulario\_salvar). Como fizemos o findView é preciso guardar o que queremos, assim, acrescentamos na frente de tudo o Button botaosalvar . Teremos:

Button botaosalvar = findViewById(R.id.formulario\_salvar);

E o código ficará da seguinte maneira:

@0verride  
 protected void onCreate(Bundle savedInstanceState) {   
 super.onCreate (savedInstanceState);  
 setContentView(R.layout.activity\_formulario);  
  
 Button botaoSalvar = (Button) findViewById(R.id.formulario\_salvar);  
  
 }

Se o 'Button' ficar assinalado em vermelho é porque falta importá-lo, nessa caso, basta dar um "Alt+Enter" em cima dele, mas provavelmente ele continuará marcado com a cor vermelha. Dessa forma, é preciso fazer o *cast*, assim, damos um "Alt+Enter" em cima do que ainda estiver em vermelho e selecionamos o "Cast to android.widget.Button". Feito isso o botão passa a ter uma referência! Se rodarmos uma simulação vemos que o *Android* traz uma animação automática, como percebemos pela imagem abaixo.:

Vamos tratar o clique do botão, isto é, inserir alguma reposta quando o botão for clicado, como uma espécie de *feedback*. Para isso vamos utilizar um Listener que grosseiramente falando é como se fosse um "ouvinte". Com isso o *Android* avisa a quem estiver "escutando" que algo foi salvo. Para cadastrar adicionamos botaoSalvar.setOnClickListener e depois que adicionamos isso ele abre chaves [] e podemos usar o atalho "Comand+P" ou "Ctrl+P". Vamos utilizar aqui a ideia de classe anônima, portanto, adicionamos o new\_onClickListener. Para fechar a instrução do botaoSalvar colocaremos o ; no final. Ficará assim:

botaoSalvar.setOnClickListener[new\_onClickListener]

Falta completar o método onClick com a ação que desejamos desencadear quando o botão for clicado. Vamos inserir uma mensagem que aparece e desaparece sempre que o botão é clicado, no *Android* isso é conhecido como *Toast*. Criaremos uma classe Toastque deve ser acompanhada do método makeText e este último pede alguns parâmetros, por exemplo, um contexto. O contexto serve para dizer ao *Android* quem somos, isto é, fornecer nossa identificação. No contexto a referência que queremos passar é da FormularioActivity e junto disso adicionamos o this:

Toast.makeText(FormularioActivity.this);

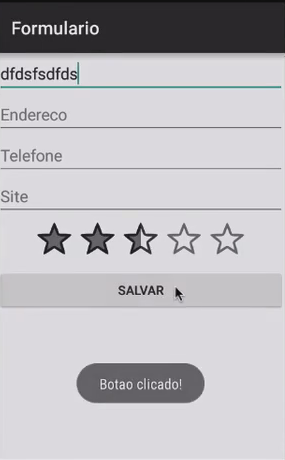
Além disso como queremos acrescentar uma mensagem escrevemos seu conteúdo dentro do parênteses, por exemplo: "Botao clicado!". É preciso definir por quanto tempo a mensagem será exibida e, por isso, usamos a constante Toast. Ao terminar de digitar aparecerão duas sugestões automáticas: a LENGTH\_SHORTe a LENGTH\_LONG. Escolhemos LENGTH\_SHORT . Teremos:

@Override  
 public void onClick(View v) {  
 Toast.makeText(FormularioActivity.this, "Botao clicado!", Toast.LENGTh\_SHORT);

Até o momento pedimos que a instância Toast fosse criada, agora, falta pedir que ela seja mostrada, então, acrescentamos "show" após o parênteses. Teremos:

@Override  
 protected void onCreate(Bundle savedInstanceState) {   
 super.onCreate (savedInstanceState);  
 setContentView(R.layout.activity\_formularios);  
  
 Button botaoSalvar = (Button) findViewById(R.id.formulario\_salvar);  
 botaoSalvar.setOnClickListener(new View.OnClickListener() {  
 @Override   
 public void onClick (View v) {  
 Toast.makeText(FormularioActivity.this, "Botao clicado!", Toast.LENGTH\_SHORT).show();  
  
  
 }  
 });  
  
 }

Vamos rodar o emulador para ver como ficou!



Agora temos um feedback no botão!