## **Mostrando vários alunos**

Voltando para o emulador podemos visualizar como está a aplicação até o momento. Já inserimos na tela uma barra inicial com o título da aplicação e também o nome do primeiro aluno. Antes de avançar e inserir o nome de mais alunos temos que relembrar alguns passos!

Lembrando que existem duas classes:

* A *Activity*, que é a tela no **Android** e nela está representado também o seu comportamento. A *Activity*aparece na tela na aba ListaAlunosActivity.java.
* O .xml, que contêm os componentes da tela, isto é, o seu conteúdo. A classe .xml está dentro da "pasta" *Layout*. E aparece na tela na aba activity\_lista\_alunos.xml

Para inserir mais nomes de alunos é preciso modificar o .xml inserindo novas *tags*. Já temos no .xml uma *tag*que adicionou o nome do primeiro aluno, o Daniel. Para acrescentar novos nomes usamos a mesma tag. Damos dois "Enter" após terminar a TextView do "Daniel". Nessa *tag* vamos inserir um conteúdo diferente, pois queremos deixá-la vazia. Inserimos um atributo text, igual como está escrito na primeira *tag*, portanto,<TextView android:text.... Perceba que a medida que escrevemos a palavra "text" ela vai sendo completada automaticamente. Só é preciso selecionar a opção android:text e pressionar *enter* para poupar a digitação.

Agora, vamos inserir o conteúdo do TextView. Queremos acrescentar o nome do próximo aluno e que ele seja "Ronaldo". Só precisamos completar com seu nome: <TextView android:text="Ronaldo".... Note que aTextView vai aparecer sublinhada de vermelho chamando atenção para um fato. Isso ocorre pois qualquerView ou qualquer componente que acrescentarmos na tela tem que ter obrigatoriamente dois atributos. Os atributos que são pedidos já estão no primeiro TextView e são layout\_width e o layout\_height:

<TextView android:text="Daniel" android:layout\_width="wrap\_content"  
android:layout\_height="wrap\_content" />

Esses dois atributos servem para definir o tamanho do componente na tela, isto é, quanto ele vai ocupar de espaço. Como não sabemos quanto esse componente ocupa em pixels utilizamos a constante do *Android*, awrap\_content, tanto para o *height* quanto para a *width*. Teremos:

<TextView android: text="Daniel'  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content" />

Isso é importante, pois não é preciso preocupar-se com essas medidas!

Obrigatoriamente, abaixo do nome Ronaldo, é preciso preencher digitando a mesma constante. Digitando apenas o "wid..." o **Android** mostra como completar: android:layout\_width. Pressione mais um enter para trocar de linha e faça o mesmo procedimento para a altura, android:layout\_height.

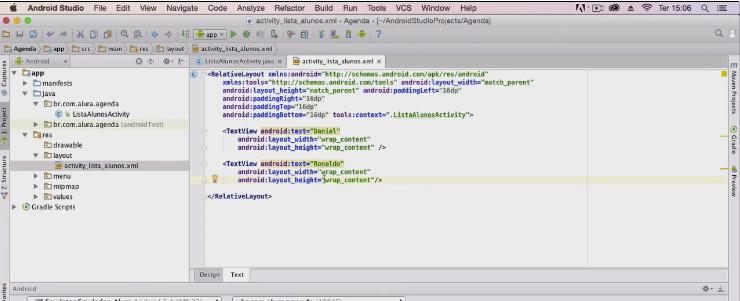
Teremos o seguinte:

<TextView android:text="Ronaldo"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"/>

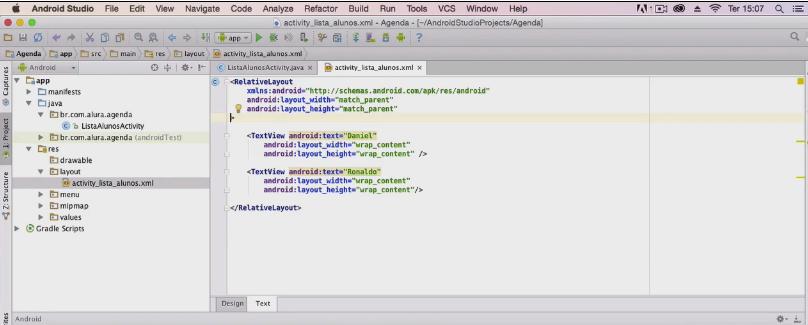
Agora que a *tag* está completa é preciso fechá-la, para isso coloque apenas a barra /> no final como demonstrado acima no final da linha.

**Retomando:** Para cada uma das Views além do conteúdo, especificamos a largura e a altura que vão seguidos de wrap\_content.

Falta especificarmos onde queremos que nossos nomes apareçam na tela. A responsabilidade disso fica a cargo da RelativeLayout. Repare que a RelativeLayout começa em cima, e termina com ListaAlunosActivityque está aberto pois, não tem o />. Dentro disso inserimos o TextView "Daniel" que está fechado e oTextView "Ronaldo", também fechado. Dentro do RelativeLayout temos dois componentes de View e no final temos o RelativeLayout que está fechado com a barra.

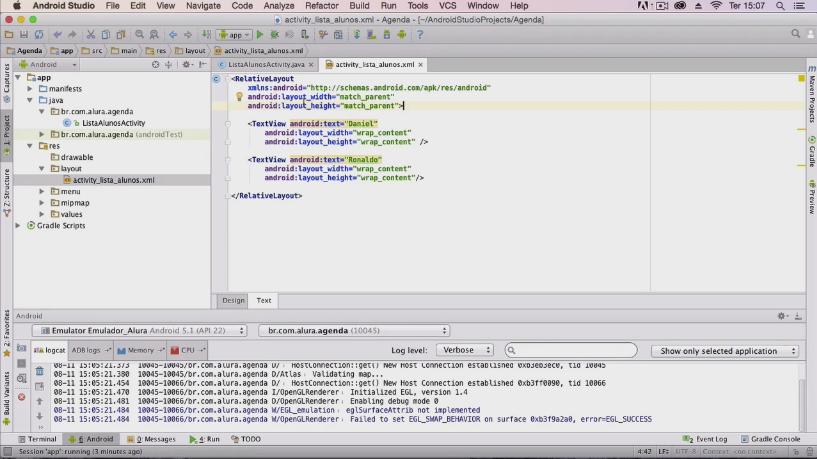


O primeiro RelativeLayout, aquele que está no início de tudo, é um Layoute vai dispor dentro de si os componentes , como se fossem seus "filhos". Falaremos sobre o RelativeLayout mais adiante, antes, vamos trocar o RelativeLayout por um outro tipo de layout. Primeiro, vamos simplificar um pouco as coisas. Percebemos que no RelativeLayout temos vários atributos que nem chegamos a utilizar. Vamos dar um enter ao final do RelativeLayout na primeira linha e depois, dos **Android** e tools nas próximas. Ficará da seguinte maneira:



Agora, vamos deixar apenas o necessário removendo aquilo que não é útil nesse momento, como a linhaxmlns:tools="http://.....

As linhas de largura e altura são necessárias ***obrigatoriamente*** para todos os componentes e não devem ser removidas. Todas as linhas seguintes até o ListaActivitypodem ser apagadas até o sinal > da *tag*RelativeLayout. O sinal de maior deve permanecer. Bom, ficamos apenas com três atributos dentro daRelativeLayout que é justamente apenas o necessário para fazer as coisas funcionarem.



Agora, vamos trocar esse RelativeLayout por um Layout diferente.

Como queremos dispor esses componentes?

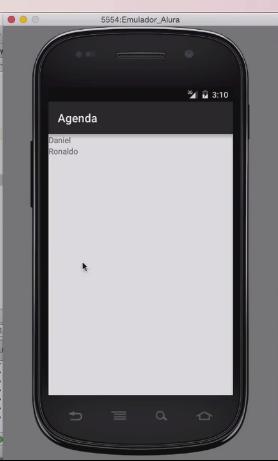
Queremos dispor os nomes de maneira linear, como uma lista. No **Android** temos uma *tag* chamadaLinearLayout que faz justamente a disposição que queremos, ou seja, de maneira linear. Um detalhe importante é que sempre que mudamos algo na parte de cima também é alterado lá embaixo. Essa alteração automática ajuda e muito, pois às vezes o arquivo é muito grande e acabamos esquecendo de trocar em outros lugares. Então, uma vez que trocou em cima vai alterar a tag embaixo automaticamente.

<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 //...  
</LinearLayout>

Após acrescentar o LinearLayout ainda temos que fazer algumas alterações, caso contrário, os nomes "Daniel" e "Ronaldo" aparecerão colados e ficarão "DanielRonaldo" e não na disposição linear que desejamos. Como o responsável por posicionar os componentes é o LinearLayout é nele que vamos introduzir as modificações para alterar a distribuição dos componentes.

Damos um "Enter" criando uma nova linha abaixo do android:layout\_height="match\_parent" e nela inserimos um atributo chamado orientation que vai auxiliar, como indica o próprio nome, na orientação que queremos dar aos nossos componentes. Ao começar a digitar a palavra *orientation* ela já vai aparecer nas opções e aí é só pressionar um enter. Depois de dar "Enter" ele vai perguntar qual a disposição que queremos dar, ou vertical ou horizontal. No nosso caso vamos selecionar a opção vertical e clicar nela. Vai ficar da seguinte maneira:android:orientation="vertical">.

Para visualizar se o que fizemos funciona, salvamos e damos um *play* . Aguardando um pouco abre-se uma janela perguntando onde queremos rodar. Como o emulador já está aberto lembre-se de selecioná-lo e pronto, aguarde ele subir.



Lembre-se que caso aconteça algo na parte visual, por exemplo, uma sobreposição de nomes, isso indica problemas no .xml.

Bom, o LinearLayout irá mostrar a lista completa dos nomes dos alunos que iremos inserir na tela, mas e se tivéssemos definido uma altura fixa para o LinearLayout, por exemplo, a metade da tela? O que ocorreria é que conforme acrescentássemos mais nomes de alunos, menor iria ficando o espaço até que ele se tornasse insuficiente.

Então, vamos inserir no LinearLayout uma constante diferente, a match\_parent. A match\_parent significa, mais ou menos, que você tem a largura igual a do seu pai. No caso da TextView se nós apagarmos owrap\_content e digitarmos o match\_parent no lugar, estaríamos dizendo que sua largura tem que ser igual a do seu pai, ou seja, igual a largura do LinearLayout que é a *tag* "pai". No caso do TextView isso não acarreta nenhum problema. Se alterarmos o LinearLayout inserindo um match\_parent na largura, teremos o seguinte android:layout\_width="match\_parent".

O LinearLayout não irá se guiar em nenhuma outra *tag* pois, ele próprio é a *tag* raiz. Ele vai se guiar a partir da largura da própria tela do celular!

Isso é justamente o que queremos, que a lista ocupe todo espaço da tela e faremos o mesmo para a altura da lista! E ficaremos com:

android:layout\_width= "match\_parent"   
android:layout\_height="match\_parent"

Assim, conseguimos mostrar quantos alunos quisermos!