02

Mostrando vários alunos

Transcrição

Vamos voltar no emulador e visualizar como está a aplicação no momento. Lembrando que já inserimos na tela uma barra inicial com o título da aplicação e com o nome do primeiro aluno. Antes de avançar e inserir o nome de mais pessoas temos que relembrar alguns passos!

Lembrando que existem duas classes:

- A *Activity*, que corresponde a tela do **Android** e representa o comportamento da aplicação. A *Activity* está na `ListaAlunosActivity.java`.

- O `.xml`, que contêm os componentes da tela, isto é, o seu conteúdo. A classe `.xml` está dentro da "pasta" *Layout* e na aba `activity_lista_alunos.xml`.

Para inserir o nome do primeiro aluno, usamos uma tag. Portanto, vamos usar o mesmo recurso para adicionar o próximo nome, damos dois "Enter" após terminar a `TextView` do "Daniel" e inserimos um atributo `text`, o `TextView`. Perceba que a medida que escrevemos a palavra "text" ela é completada automaticamente. Só é preciso selecionar a opção `android:text` e pressionar *Enter* para poupar a digitação.

Agora, vamos inserir o conteúdo do `TextView`. Como queremos acrescentar o nome do próximo aluno, que é "Ronaldo", completamos com seu nome: `TextView android:text="Ronaldo"`. Note que a `TextView` aparecerá sublinhada em vermelho. Isso ocorre, pois qualquer `View` ou qualquer componente que acrescentarmos na tela deve conter, obrigatoriamente, dois atributos. Os atributos que são pedidos já estão no primeiro `TextView` e são `layout_width` e o `layout_height`. Observe:

```
<TextView android:text="Daniel" android:layout_width="wrap_content"
android:layout_height="wrap_content" />
```

Estes dois atributos definem o tamanho que o componente ocupa de espaço na tela. Como não sabemos a quantidade de pixels desse componente, utilizaremos a constante do *Android*, a `wrap_content`, tanto para altura (*height*) quanto para largura(*width*). Teremos:

```
<TextView android:text="Daniel'
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

Ou seja, não é preciso preocupar-se com as medidas!

É obrigatório, abaixo do nome Ronaldo, preencher usando 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` .

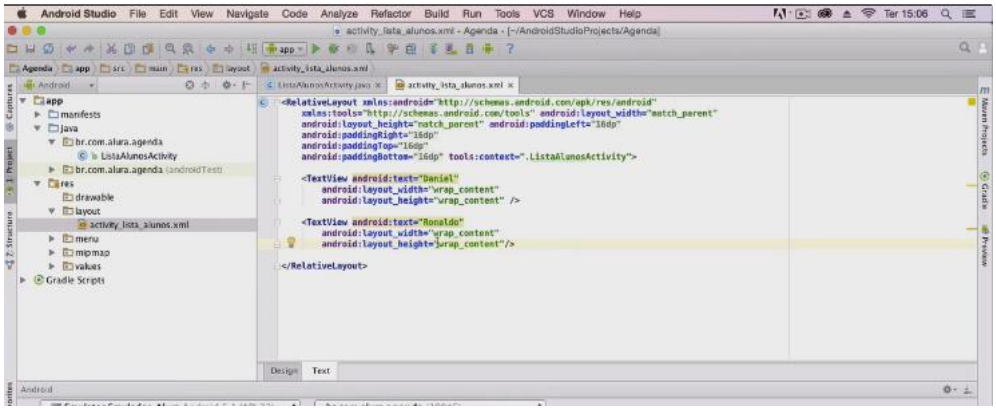
Teremos o seguinte:

```
<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 e o sinal de maior `/>` no final.

Retomando: Para cada uma das `Views` além do conteúdo, especificamos a largura e a altura que vão seguidos de `wrap_content` .

Falta especificarmos em qual local da tela queremos que os nomes apareçam. A responsabilidade disso fica a cargo da `RelativeLayout` . Repare que a `RelativeLayout` começa em cima e termina com `ListaAlunosActivity` que está aberto pois, não tem o `/>` . Dentro disso inserimos o `TextView` do "Daniel", que está fechado, e o `TextView` do "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, como o próprio nome diz, é um `Layout` . Ele dispõe dentro de si os componentes, como se fossem seus "filhos". Falaremos sobre o `RelativeLayout` mais adiante, antes, vamos trocar o `RelativeLayout` por outro tipo de `layout` . Primeiro, vamos simplificar! Note que no `RelativeLayout` existem vários atributos que nem chegamos a utilizar. Vamos dar um *Enter* ao final do `RelativeLayout` , na primeira linha, depois dos **Android** e `tools` . E deixaremos apenas o necessário removendo aquilo que não é útil, como a linha `xmlns:tools="http://....` .

As linhas de largura e altura são ***obrigatórias*** para todos os componentes e não devem ser removidas. Todas as linhas seguintes até o `ListaActivity` podem ser apagadas até o sinal `>` da *tag* `RelativeLayout` . O sinal de maior deve permanecer. Ficaremos apenas com três atributos dentro da `RelativeLayout` que é justamente apenas o necessário para fazer as coisas funcionarem:

```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView android:text="Daniel"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
```

```
    <TextView android:text="Ronaldo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

Agora, vamos trocar esse `RelativeLayout` por um `Layout` diferente.

Como queremos dispor esses componentes?

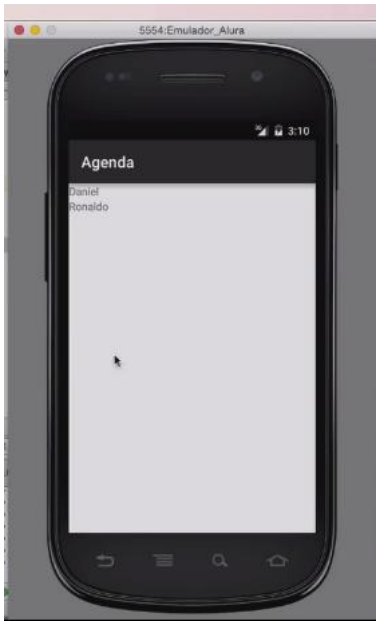
Queremos dispor os nomes de maneira linear, como uma lista. No **Android** existe uma *tag* chamada `LinearLayout` que faz justamente a disposição que queremos, ou seja, de maneira linear. Um detalhe importante é que sempre que mudamos algo na parte de cima também é alterado embaixo. Essa alteração automática ajuda e muito, evitando erros por esquecimento. Uma vez que a tag é alterada no topo do código ela também é modificada automaticamente embaixo.

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

Após acrescentar o `LinearLayout` é preciso fazer algumas alterações, caso contrário, os nomes "Daniel" e "Ronaldo" aparecerão como "DanielRonaldo" e não na disposição linear que desejamos. O `LinearLayout` é responsável por posicionar os componentes e é nele que introduzimos 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 indica a orientação dos componentes. Ao começar a digitar a palavra *orientation* ela já aparece nas opções, então, é só pressionar um Enter. Depois do "Enter" ele pergunta qual a disposição que queremos dar, vertical ou horizontal. No caso, vamos selecionar a opção vertical e clicar nela. Ficará da seguinte maneira: `android:orientation="vertical">` .

Para visualizar se o que fizemos funciona, salvamos e damos um *play*. Como o emulador já está aberto lembre-se de selecioná-lo e pronto, aguarde subir.



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

O `LinearLayout` mostra a lista completa dos nomes dos alunos que vamos inserir na tela, mas e se tivéssemos definido uma altura fixa para o `LinearLayout`? Por exemplo, a metade da tela? Conforme acrescentássemos mais nomes, menor iria ficar o espaço até se tornar insuficiente.

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 apagarmos o `wrap_content` e digitarmos o `match_parent` no lugar estaremos dizendo que sua largura deve ser igual a do seu pai, ou seja, igual a largura do `LinearLayout` que é a *tag* de referência. No caso do `TextView` isso não acarreta nenhum problema. Se alterarmos o `LinearLayout` inserindo um `match_parent` na largura, teremos: `android:layout_width="match_parent"`.

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

Queremos que a lista ocupe todo espaço da tela e faremos o mesmo para a altura da lista. Portanto, ficaremos com:

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >


    <TextView android:text="Daniel"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <TextView android:text="Ronaldo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</LinearLayout>
```

Assim, mostramos quantos alunos queremos!