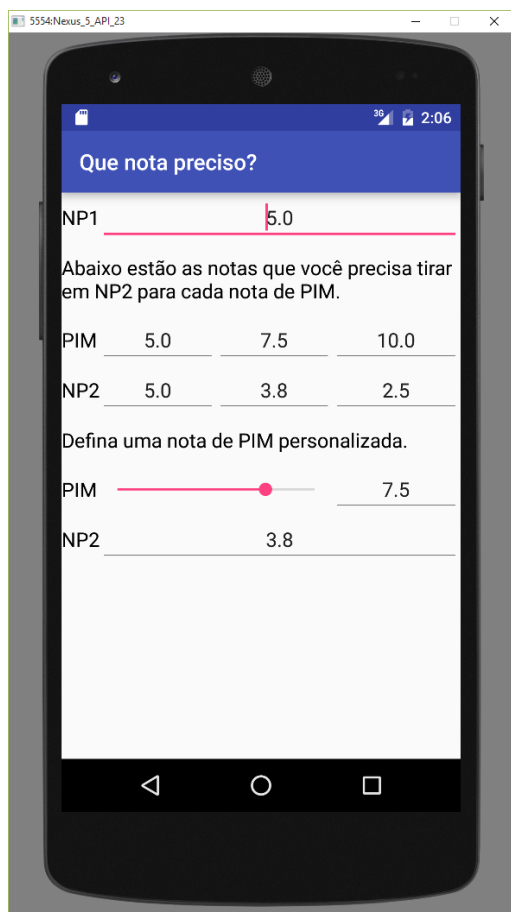


Roteiro de Laboratório 03

José Cassiano Grassi Gunji



Neste roteiro vamos desenvolver um aplicativo completo que calcula qual é a nota necessária na NP2 para que a média final seja maior ou igual a 5.0. O usuário fornece o valor da nota NP1 e o aplicativo calcula a nota NP2 para os valores fixos de PIM (5.0, 7.5 e 10.0). O usuário ainda tem a oportunidade de definir um valor personalizado para o PIM.

Para que o roteiro não fique muito longo, vamos dividir o aplicativo em duas etapas. Neste roteiro, vamos desenvolver a interface gráfica. No próximo roteiro, será feito o código que realiza o controle do aplicativo e suas regras de negócio.

Vamos iniciar o projeto no Android Studio criando um novo projeto chamado “Que nota preciso?” sob o domínio “unip.br”.

Na próxima tela, indique a compatibilidade do aplicativo com “Phone and Tablet” usando a API mínima 15 (IceCreamSandwich). Para este aplicativo você pode selecionar uma API ainda mais baixa, como a 10 (Gingerbread). Entretanto,

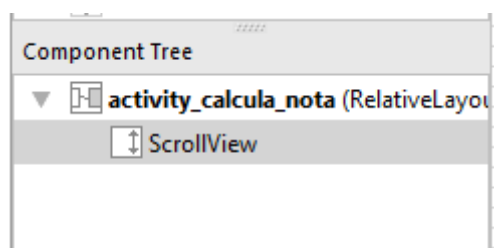
o número de dispositivos compatíveis não sobe muito (de 97,4% para 100%).

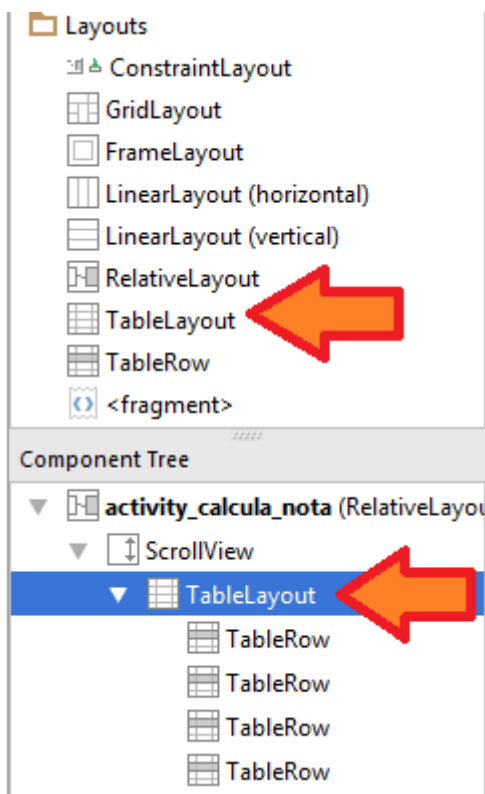
Por fim, permita que o assistente crie uma Activity para você. Utilize o padrão “Empty Activity”, assim como fizemos no exemplo “Olá Mundo!”.

Defina o nome da Activity como “CalculaNotaActivity” e o nome do layout como “activity_calcula_notas”. Clique em “Finish” para terminar o assistente.

O assistente cria uma Activity com o TextView “Hello World!”. Selecione-o no editor gráfico ou na árvore de componentes e exclua-o.

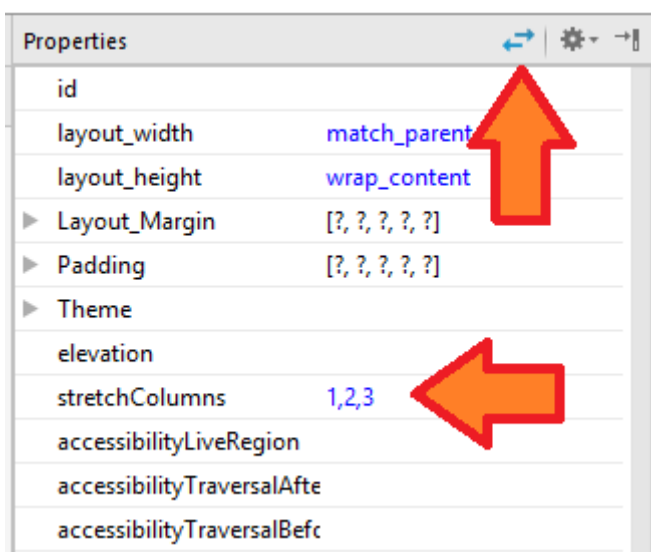
Nossa Activity poderá ser grande demais para ser apresentada por inteiro na tela de dispositivos menores ou se o dispositivo for girado para a orientação paisagem. Assim, vamos criar um projeto gráfico que utiliza uma barra de rolagem vertical. Arraste o Container ScrollView para o RelativeLayout da árvore de componentes como

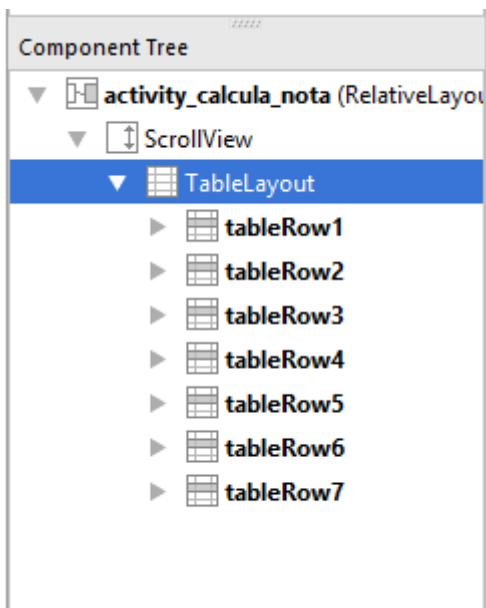




mostrado na figura.

Para criar a interface gráfica, vamos utilizar o Layout TableLayout, que organiza os componentes como uma tabela. Para tanto, arraste o TableLayout para dentro do ScrollView na árvore de componentes. O componente TableLayout irá redimensionar e reposicionar os componentes gerenciados por ele de maneira automática sempre que a tela mudar de tamanho (ao ser instalado em dispositivos diferentes) ou se a tela mudar de orientação. Para tanto, precisamos indicar ao TableLayout quais as colunas ele pode esticar para ocupar toda a sua área disponível. Primeiro, modifique a tela de propriedades para exibir todas as propriedades. A seguir, vamos modificar sua propriedade stretchColumns para o valor "1, 2, 3" para indicar que a segunda, terceira e quarta colunas podem ser esticadas (a numeração das colunas inicia em zero, como qualquer array ou coleção em Java).

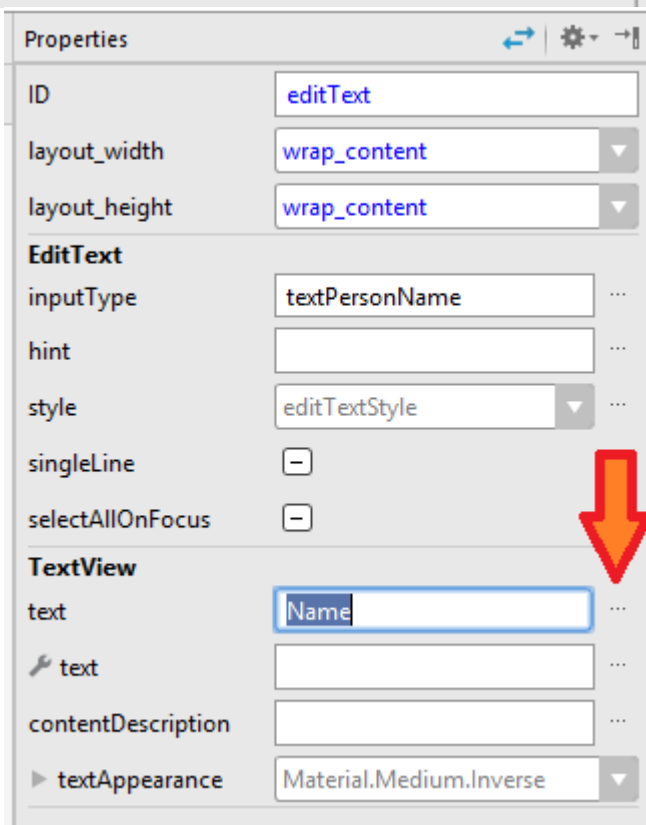
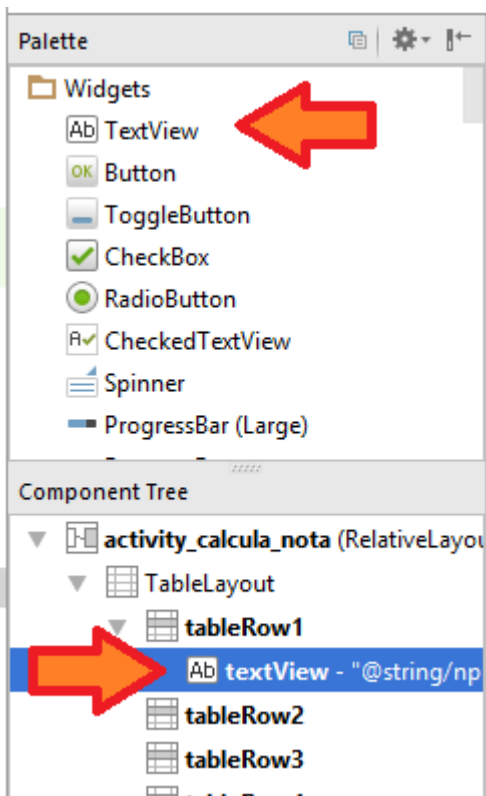
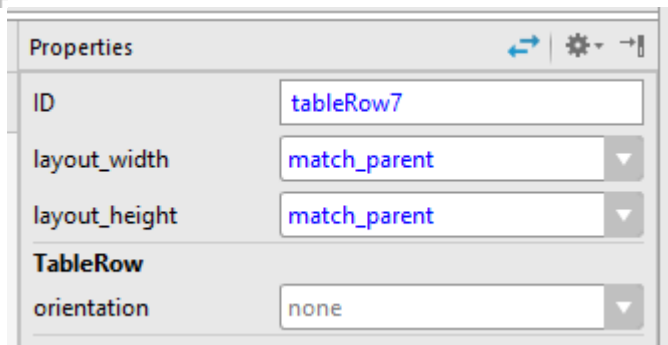




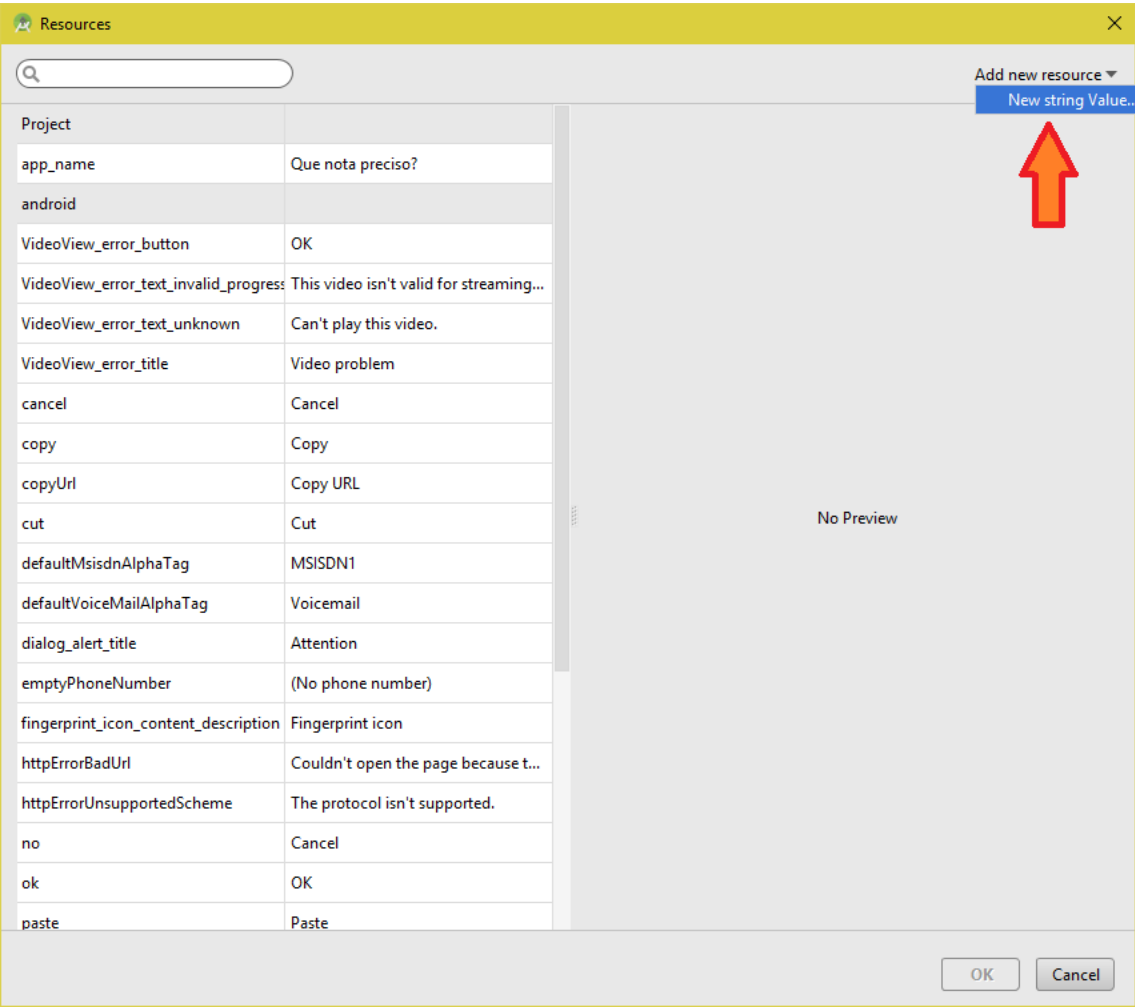
A interface gráfica que projetamos apresenta 7 linhas de componentes. Assim, vamos arrastar sete TableRow para dentro do TableLayout na árvore de componentes. Também vamos alterar a propriedade id de cada TableRow para atribuir um nome único a cada um. Atribua os nomes tableRow1, tableRow2, ..., tableRow7. Sua árvore deve apresentar a estrutura mostrada na figura.

É uma boa prática de programação atribuir um nome a todos os componentes da interface gráfica, não só àqueles que deverão ser referenciados na classe Java. Só é possível referenciar componentes que tenham uma id definida.

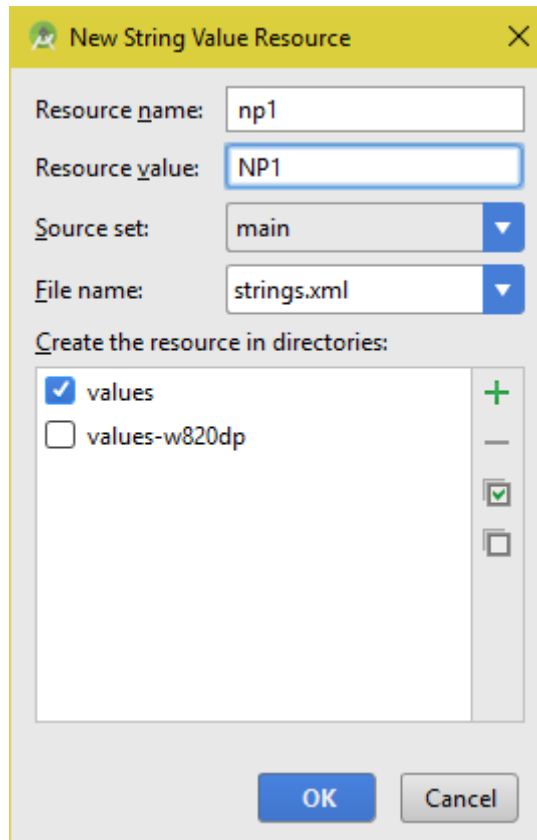
Na tableRow1, adicione um TextView. Em sua propriedade text, ao invés de definir o texto "NP1" diretamente, vamos criar uma entrada no arquivo string.xml com o valor que será apresentado no TextView. Para tanto, clique no botão com as reticências na propriedade text.



Na janela que se abre, selecione New Resource e New String Value.



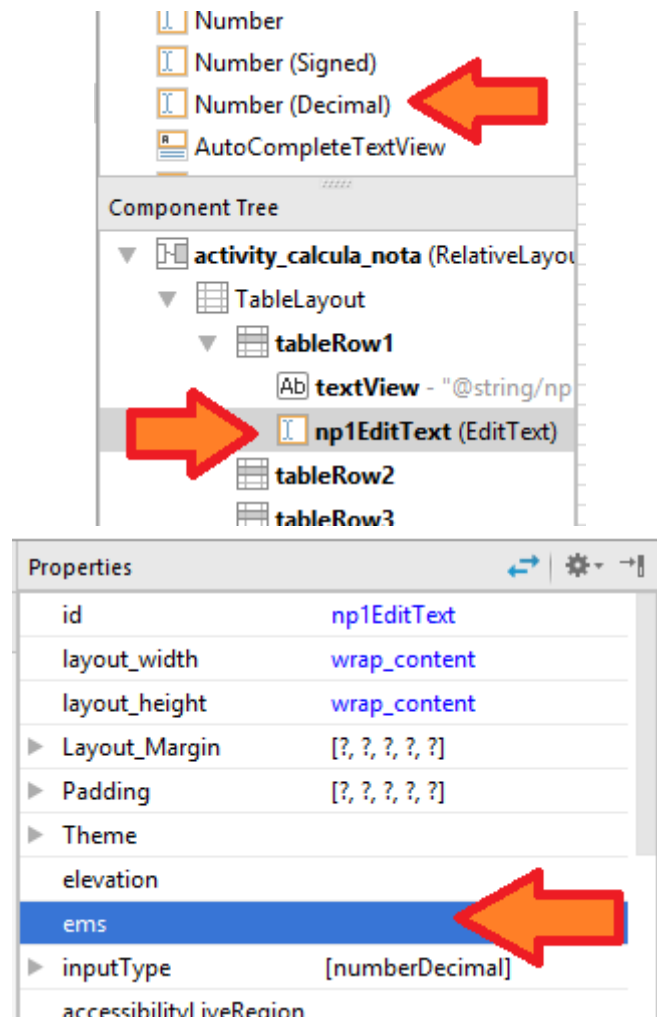
Na janela que se abre, indique Resource name como “np1” e Resource value como “NP1”.



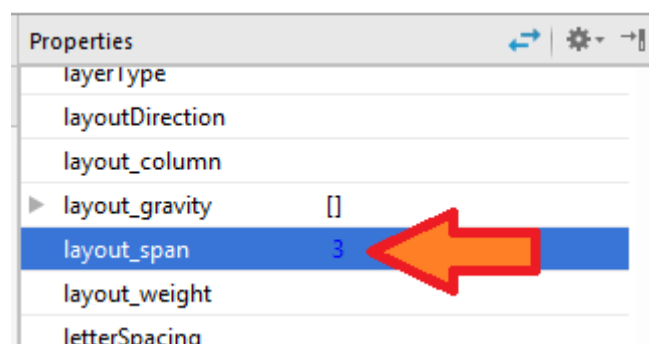
Isto cria a entrada np1 com o valor NP1 no arquivo strings.xml.

```
<resources>
  <string name="app_name">Que nota preciso?</string>
  <string name="np1">NP1</string>
</resources>
```

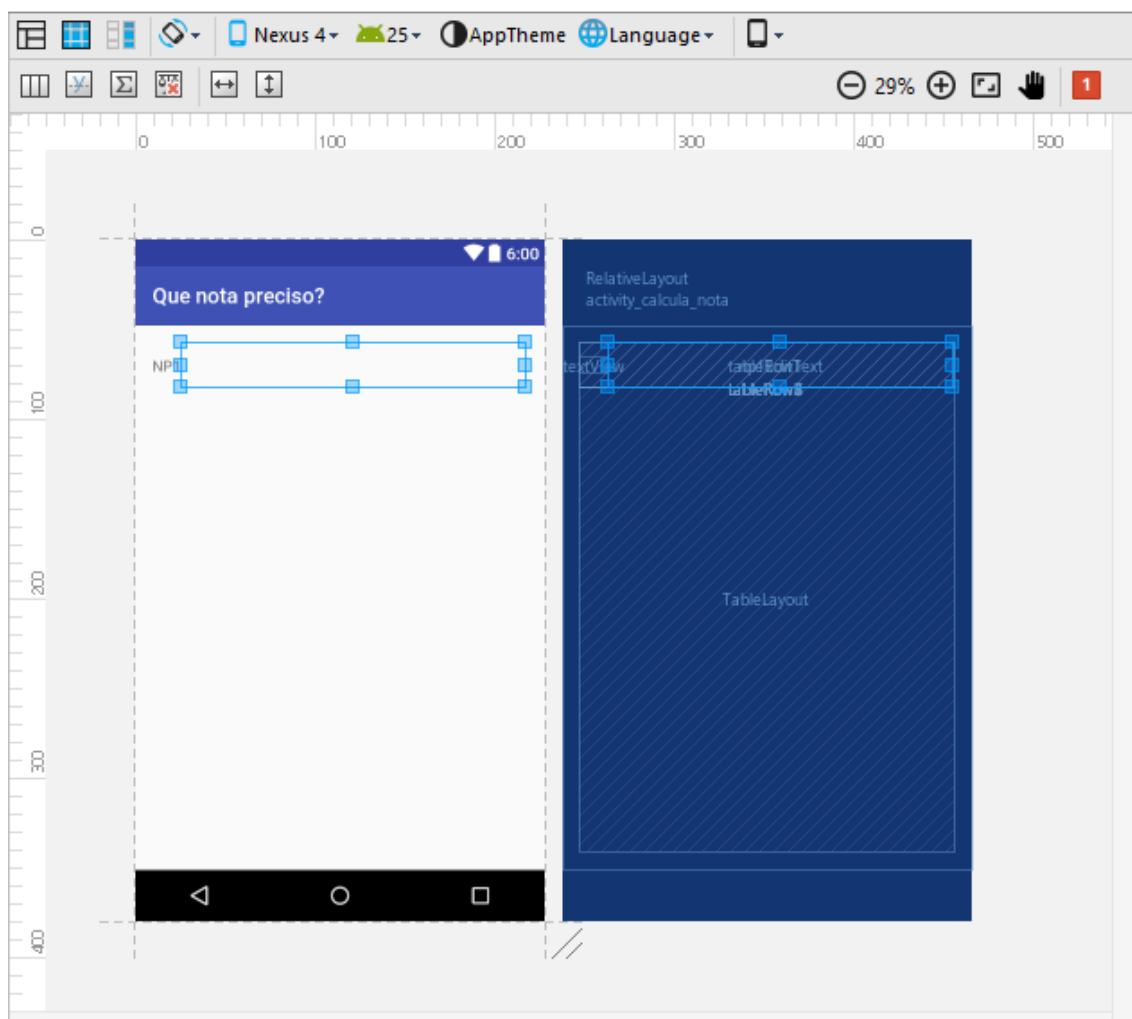
Adicione agora ao `tableRow1` um `TextField` `Number` (`Decimal`). Defina sua `id` como `np1EditText`. Defina também sua propriedade `ems` para nenhum valor. Um valor na propriedade `ems` indica um tamanho fixo para o `EditText`, o que acaba fazendo com que o `layout` acabe ultrapassando os limites da tela.



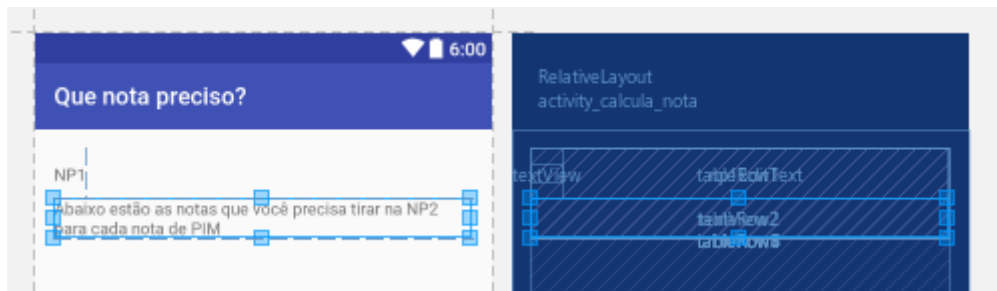
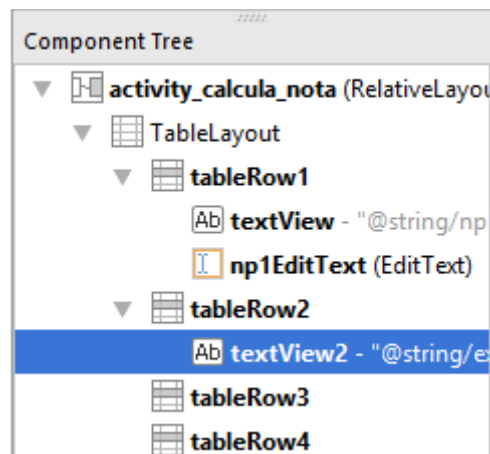
Além disso, queremos que o `np1EditText` ocupe o espaço total das três colunas restantes da tabela. Assim, vamos definir a sua propriedade `layout:span` para o valor 3.



Seu editor gráfico deve estar semelhante à imagem abaixo.



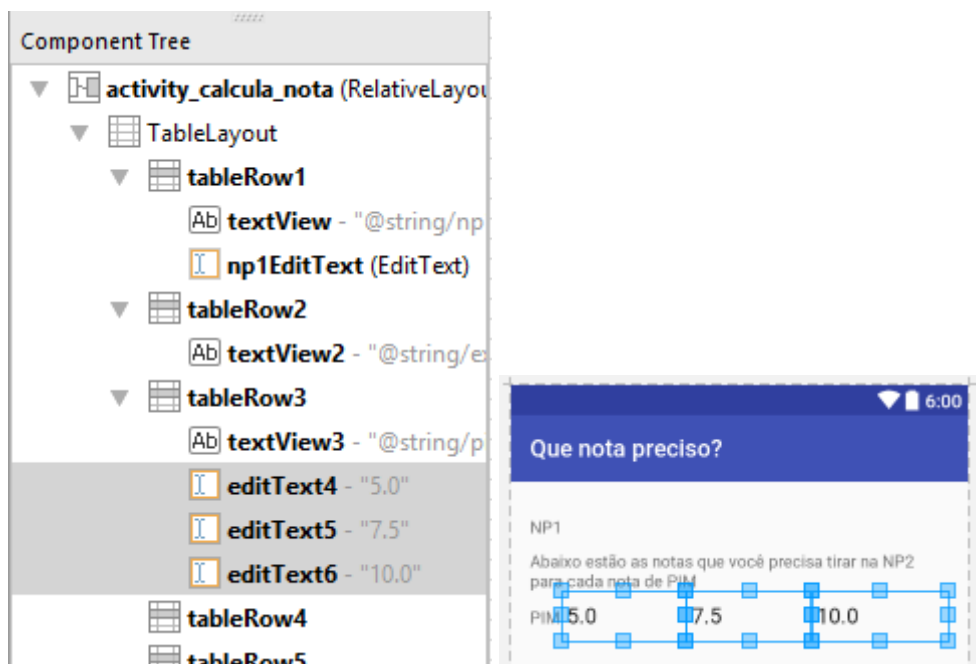
Na `tableRow2`, adicione um `TextView`. Em sua propriedade `text`, crie uma nova entrada no arquivo `strings.xml` chamada `"explicacao1"` com o valor "Abaixo estão as notas que você precisa tirar na NP2 para cada nota de PIM." Defina sua propriedade `layout:span` para 4, de modo que ele ocupe as quatro colunas da tabela.



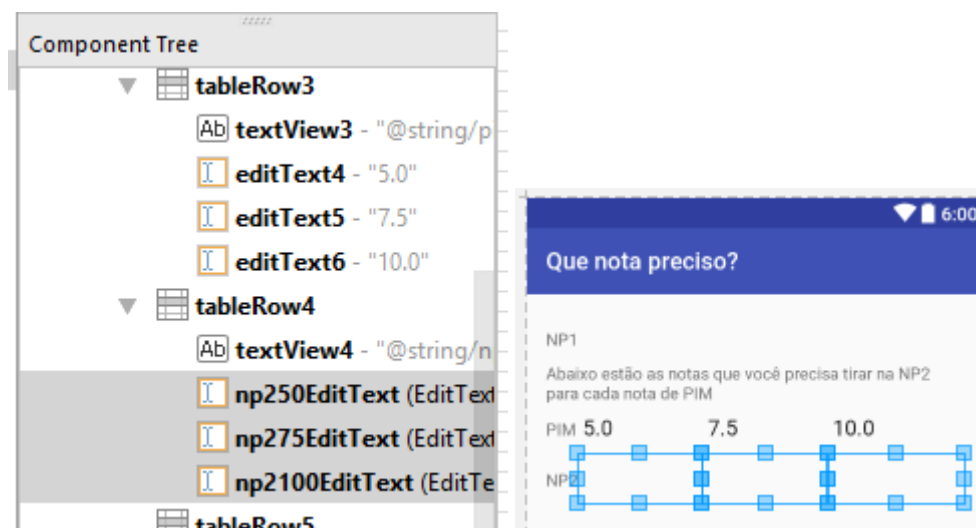
Na `tableRow3`, adicione um `TextView` com uma nova entrada no arquivo `strings.xml` chamada “pim” com o valor “PIM”.

Adicione três `Text Fields Number (Decimal)`, com os textos “5.0”, “7.5” e “10.0”. Estes textos podem ser definidos diretamente na propriedade `text` de cada `EditText`, pois nunca seriam traduzidos.

Selecione os três `EditText` clicando neles na árvore de componentes mantendo a tecla `CTRL` pressionada. Isso permite que você selecione mais de um componente simultaneamente. Com isso, a palheta de propriedades exibe apenas as propriedades em comum que os componentes possuem. Altere a propriedade `ems` das três para nada.

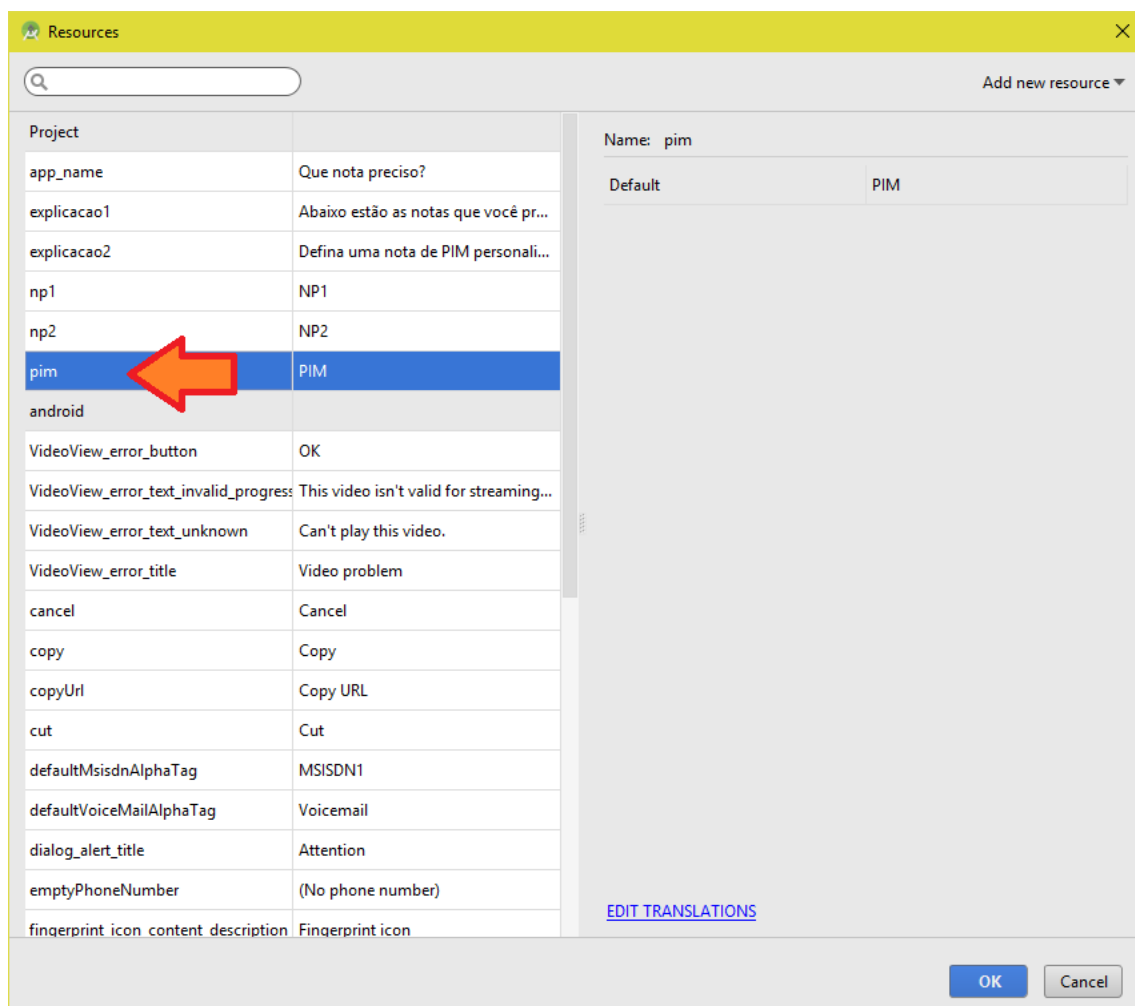
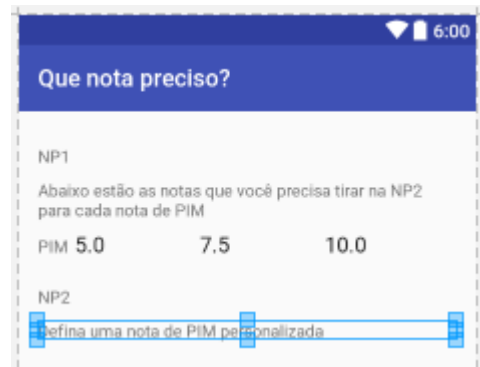


Na `tableRow4`, crie um `TextView` com uma nova entrada no arquivo `strings.xml` chamada “np2” com o valor “NP2”. Adicione três `Text Fields Number (Decimal)`. Eles receberão os valores calculados da nota NP2, por isso devem ser nomeados como `np250EditText`, `np275EditText` e `np2100EditText`. Não se esqueça de apagar o valor da propriedade `ems` deles.



Ao `tableRow5`, adicione um `TextView` com uma nova entrada no arquivo `strings.xml` de nome “`explicacao2`” e valor “Defina uma nota de PIM personalizada”. Defina seu `layout:span` para 4.

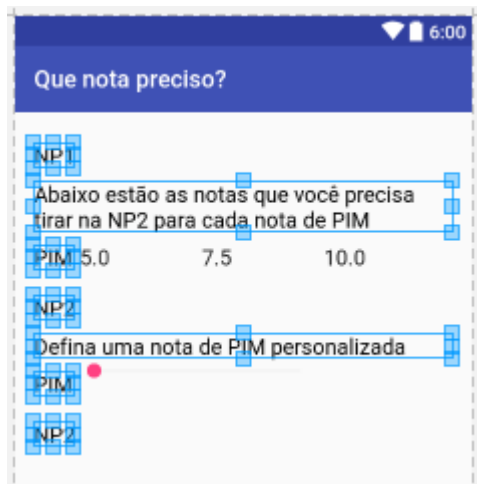
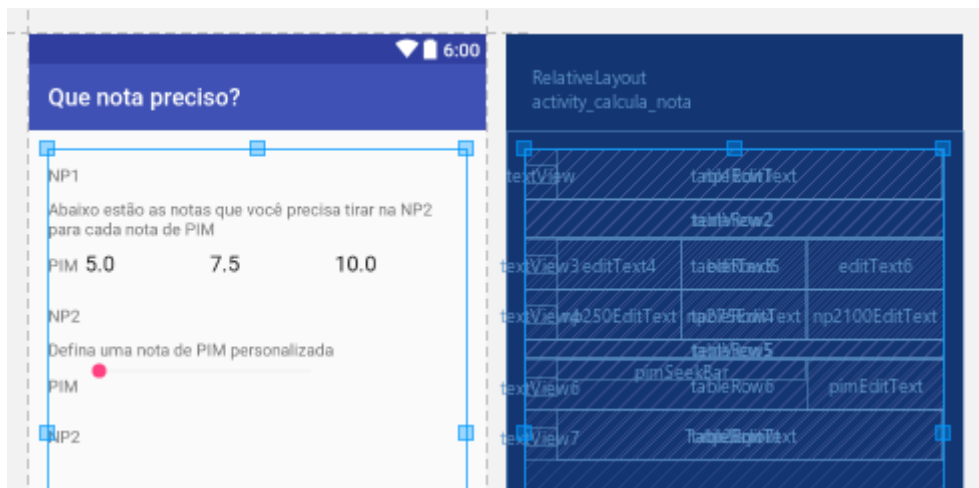
Adicione ao `tableRow6` um `TextView`. Desta vez não é preciso definir uma nova entrada no arquivo `strings.xml`. Ela já existe e vamos reutiliza-la. Clique no botão com as reticências na propriedade `text`. Escolha a entrada “`pim`” de seu arquivo `strings.xml` do seu projeto.



Adicione um SeekBar ao tableRow6. Defina sua id como pimSeekBar. Altere sua propriedade layout:span para 2.

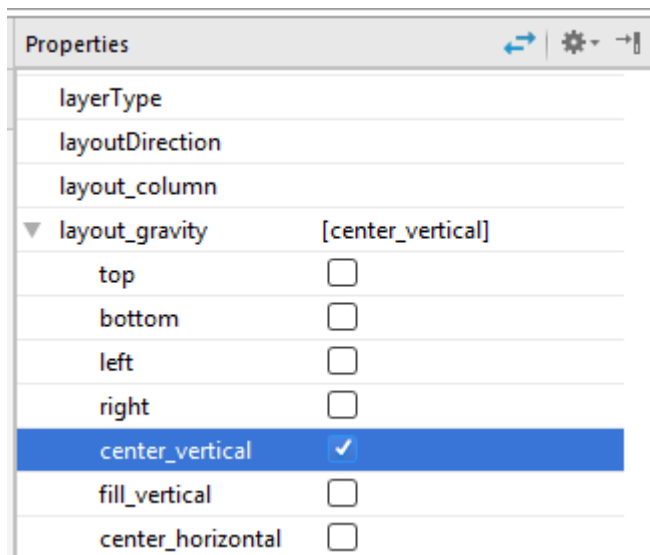
Adicione um Text Field Number (Decimal). Defina sua id para pimEditText e apague o valor de sua propriedade ems.

Na tableRow7, adicione um TextView com a entrada “np2” do arquivo strings.xml. Adicione um Text Field Number (Decimal). Defina seu layout:span para 3 e apague sua propriedade ems. Defina seu id para np2EditText.

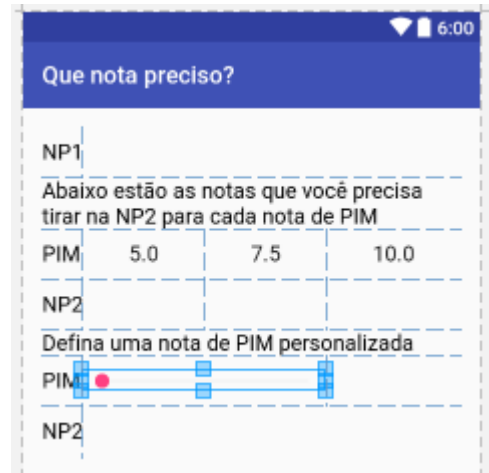


Vamos mudar a cor e o tamanho dos textViews. Selecione-os todos mantendo a tecla CTRL pressionada. Altere a propriedade textColor para #000000 (preto) e a propriedade textSize para 18sp. Como cada dispositivo Android possui um tamanho de tela diferente e uma densidade de pixels diferente, é muito difícil definir tamanhos por pixels absolutos. Assim, o Android define dois tipos de pixels que não dependem da resolução da tela. Eles são os pixels independentes da escala (sp), que são úteis para definir tamanhos de fontes e os pixels independentes da densidade (dp), que são úteis

para todo o resto. Assim, com estes tipos de pixels, sua interface gráfica terá a mesma aparência em qualquer dispositivo.



Selecione o pimSeekBar. Para fazer com que ele seja centralizado verticalmente, altere sua propriedade layout:gravity para center_vertical.



Neste ponto, o leiaute do aplicativo está pronto. Você pode executar o aplicativo para confirmar que seu leiaute funciona corretamente, mas o aplicativo em si ainda não faz nada. Na próxima atividade, vamos criar o código que faz o controle e as regras de negócio do app.