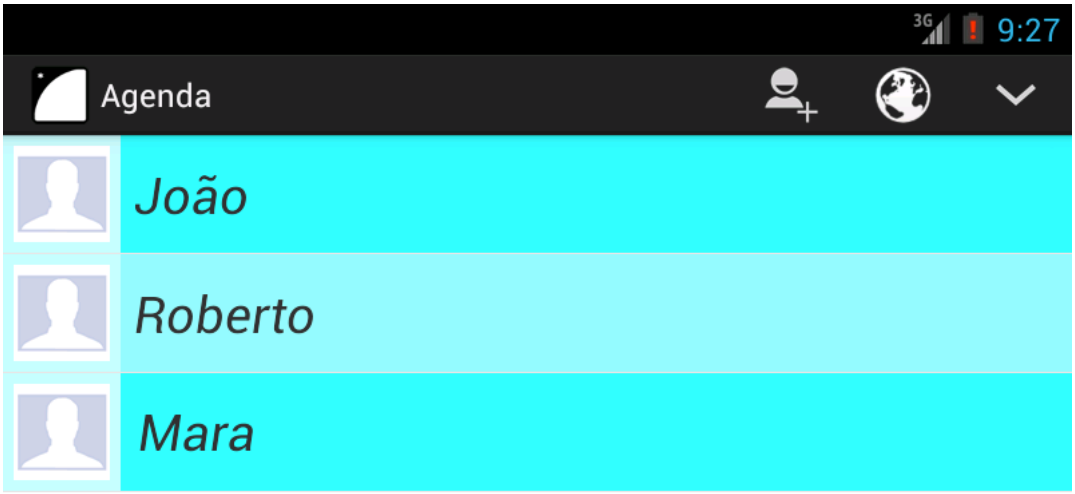
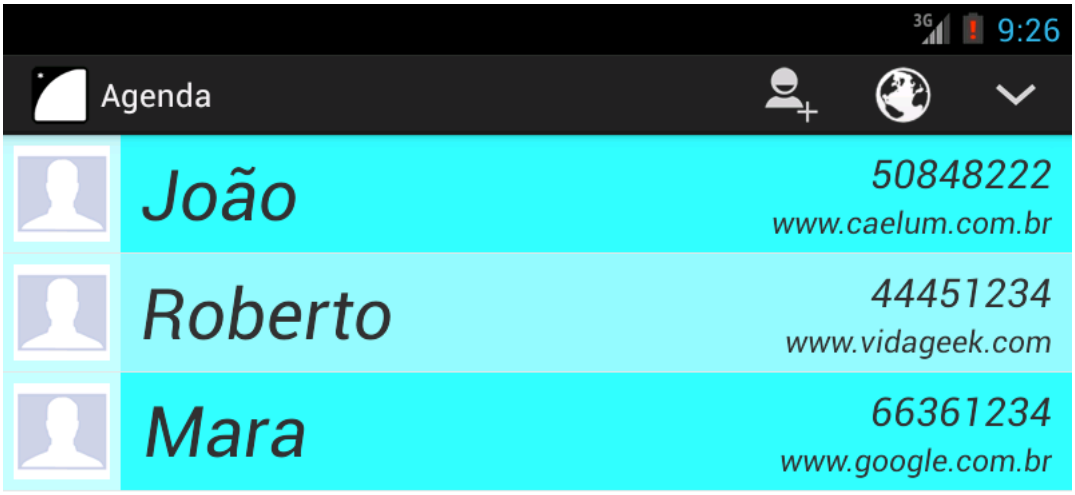
02

Melhorando a usabilidade com Application Resources

Se rodarmos nossa aplicação atualmente em um *smartphone* com tela maior ou um *tablet* e colocarmos o dispositivo na orientação horizontal teremos uma tela como:



Repare que o espaço está mal aproveitado, a linha do `ListView` traz apenas o nome do aluno e existe muito espaço disponível para mostrarmos outras informações úteis, como talvez o telefone e o site do aluno, como a imagem abaixo:



Para que esse objetivo seja atingido basta alterarmos nosso layout `item.xml`, que representa um item da `ListView` que apresenta os alunos.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/fundo"
    ... >

    <ImageView
        android:id="@+id/foto"
        ...
        android:layout_alignParentLeft="true"
```

```
            android:src="@drawable/ic_no_image" />

        <TextView
            android:id="@+id/nome"
            ...
            android:layout_centerVertical="true"
            android:layout_toRightOf="@id/foto" />

        <TextView
            android:id="@+id/telefone"
            ...
            android:layout_alignParentRight="true"/>

        <TextView
            android:id="@+id/site"
            ...
            android:layout_alignParentRight="true"
            android:layout_below="@+id/telefone"/>

    </RelativeLayout>
```

Para preencher as novas informações basta alterarmos o `inflater` para buscar as novas `Views` e colocar os dados do aluno da linha:

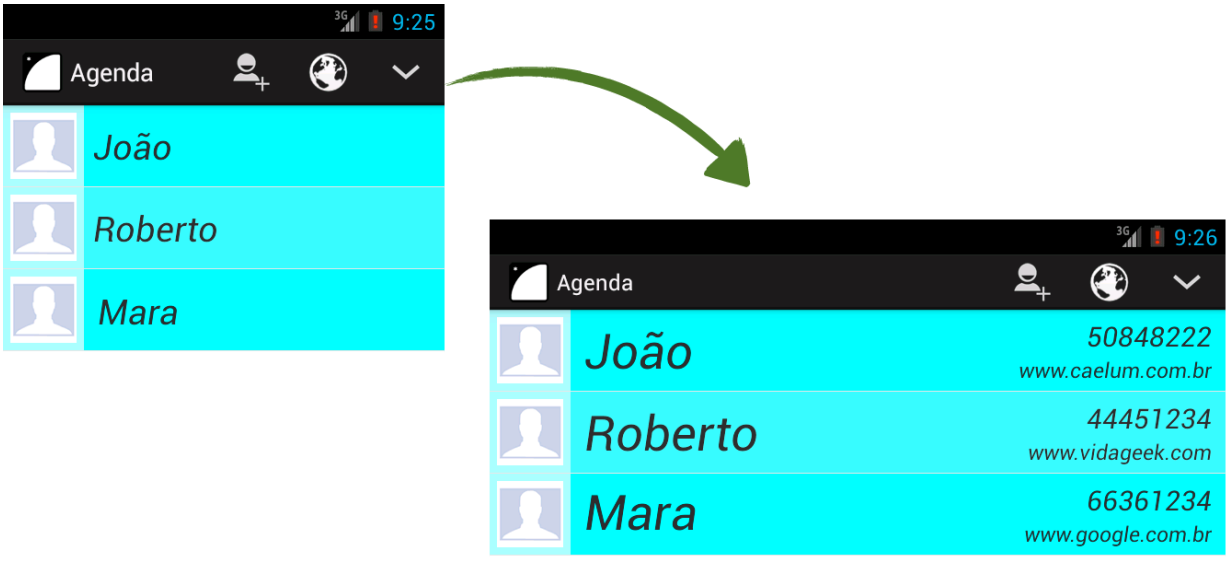
```
public View getView(int posicao, View convertView, ViewGroup parent) {
    // ...

    TextView telefone = (TextView) view.findViewById(R.id.telefone);
    telefone.setText(aluno.getTelefone());

    TextView site = (TextView) view.findViewById(R.id.site);
    site.setText(aluno.getSite());

    return view;
}
```

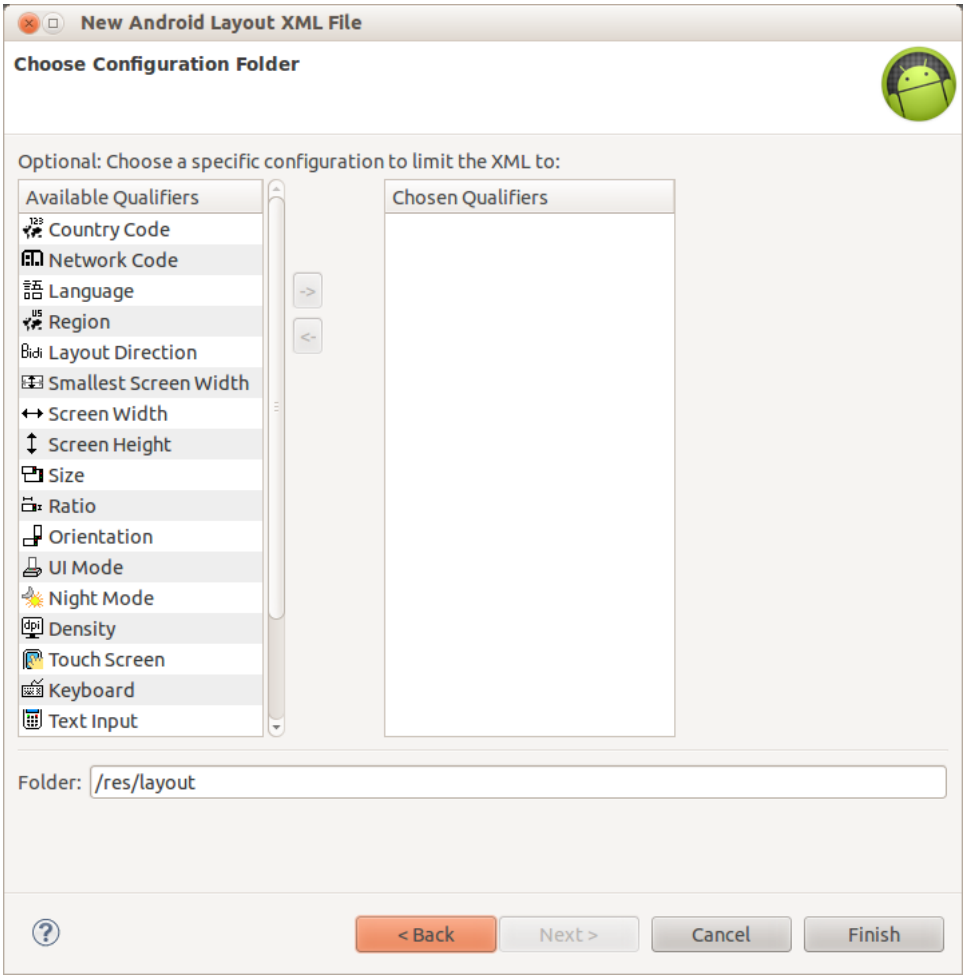
O grande desafio é: como fazer isso apenas se o aparelho estiver na horizontal?



Application Resources

A plataforma Android desenvolveu uma maneira prática de dar suporte a diversos tipos de dispositivos com diferentes versões do Android, diversas densidades e tamanhos de tela.

Era necessário criar uma forma para identificar *versão do android, tamanho de tela, densidade de tela, idiomas* e outros. A forma encontrada foram os *qualifiers*: identificadores que representam características do aparelho.



Os *qualifiers* para a densidade de tela são: *xhdpi* para extrema, *hdpi* para alta, *mdpi* para média e *ldpi* para baixa.

Agora queremos criar um ícone da nossa aplicação para cada qualidade de tela. Mas onde devemos colocar cada ícone para que o Android *renderize* ele de acordo com a densidade do dispositivo?

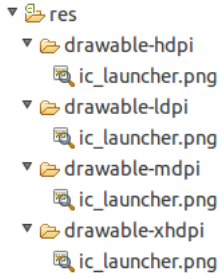
Aqui há um grande diferencial competitivo do Android: em vez de definirmos via programação o momento de usá-las, podemos jogar essa responsabilidade para o ***Application Resources***.

Vamos definir o nome de todos as imagens como *ic_launcher.png*. Sabemos que a pasta destinada a imagens é a pasta *res/drawable*. Se tentarmos colocar as imagens lá dentro não será possível, por que todas possuem o mesmo nome. Então vamos definir pastas específicas para cada imagem de acordo com sua **densidade de tela**.

Vamos criar 4 pastas *drawable* informando as densidade de telas para cada uma delas. Para vincular um *qualifier* a uma pasta usamos um **hífen**. Então teremos como resultado as pastas: *res/drawable-xhdpi*, *res/drawable-hdpi*, *res/drawable-mdpi* e *res/drawable-ldpi*.

Agora vamos copiar cada *ic_launcher.png* para sua pasta específica de acordo com sua densidade.

O projeto ficará como na imagem abaixo:



Repare que agora a imagem será carregada pelo ***Application Resources***. Quando nossa aplicação for executada no device, o *Application Resources* entrará em ação e irá identificar qual a densidade de *pixels* da tela do aparelho. Com essa informação ele vai procurar as pastas que contenham essa característica, ou seja, ele vai procurar as pastas com esse *qualifier* e carregar o conteúdo delas.

Como os nossos arquivos tem o mesmo nome não precisamos de um `if` no nosso código.

Note que não precisamos de um grande esforço para resolvermos esse desafio. A plataforma Android já possui uma solução bem interessante para problemas deste tipo.