GREat
Grupo de Redes de Computadores, Engenharia de Software e Sistemas

UFC VIRTUAL

Universidade Federal do Ceará

Hello World e Alternative resources em Android

Prof. Windson Viana

Prof. Fernando Trinta

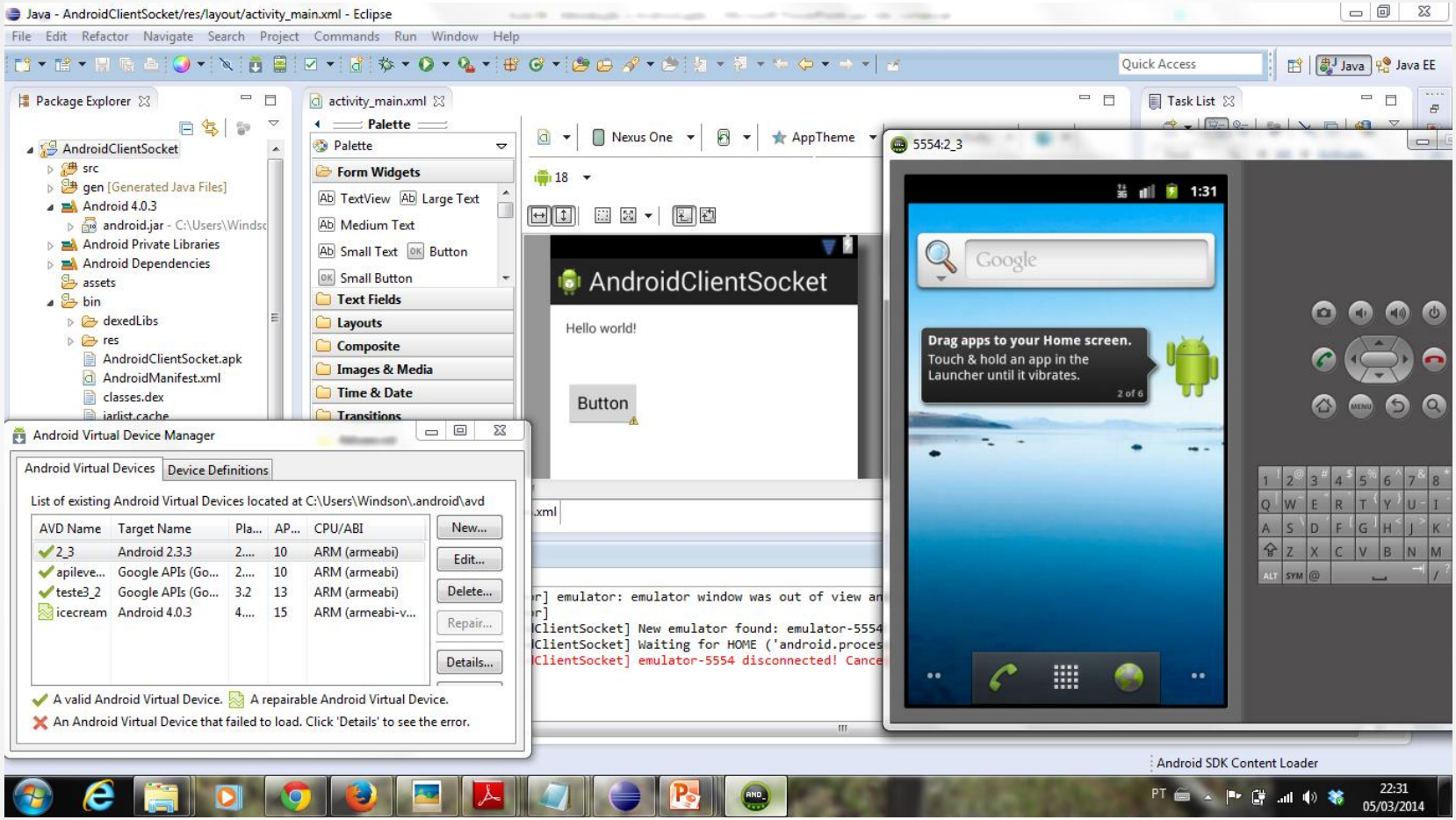
Disciplina de Introdução à Computação Móvel e Ubíqua – 2014.1
Curso de Sistemas e Mídias Digitais

Disciplina Redes Móveis Sem Fio – 2014.1
Mestrado e Doutorado em Ciência da Computação

Ambiente de Desenvolvimento

Android Studio v 0.4.2 + JDK

Ambiente de Desenvolvimento



Eclipse + ADT Plugin + JDK

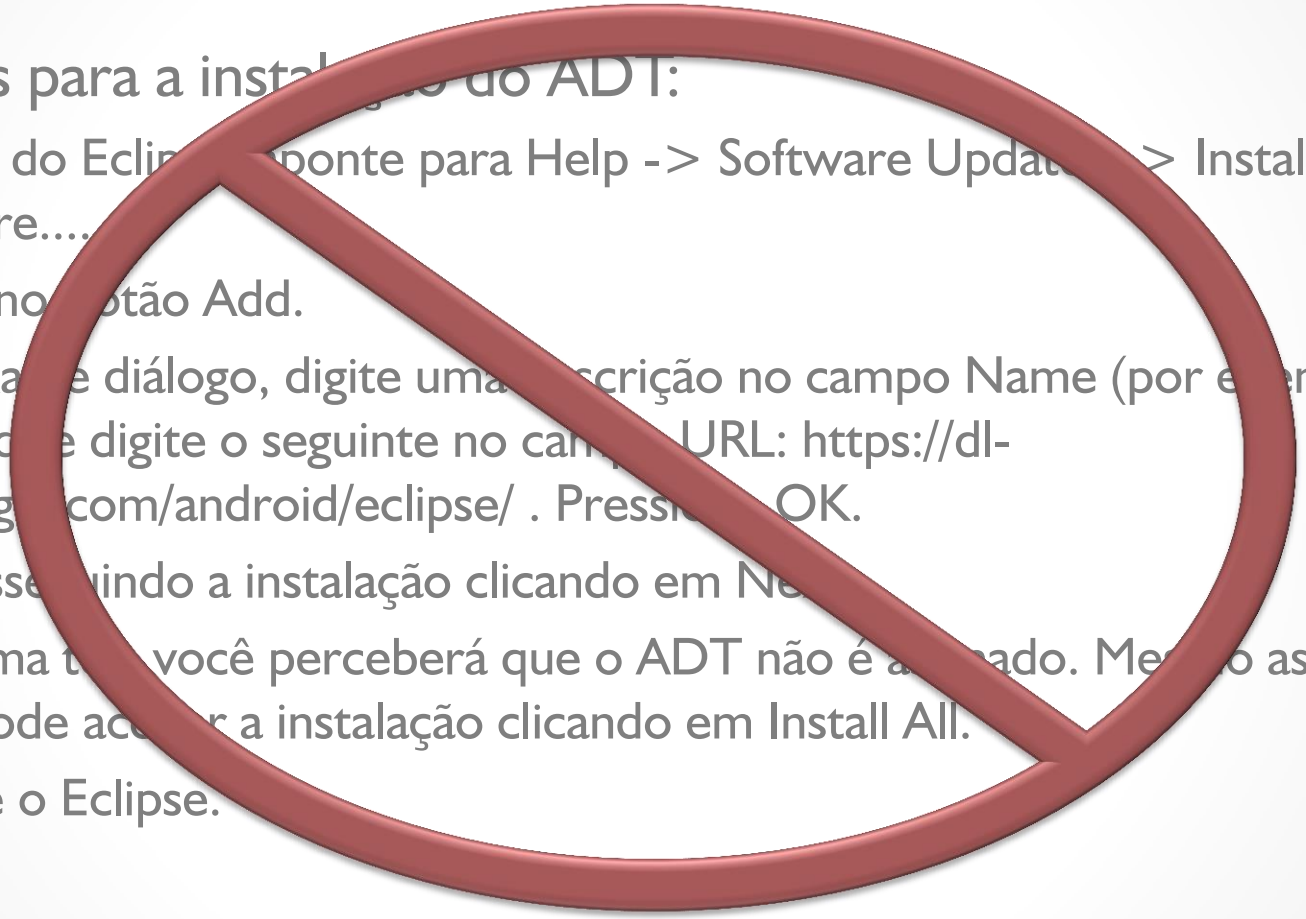
Ambiente de Desenvolvimento

- Instalar o SDK Android:
- http://developer.android.com/sdk/index.html
- Fazer o download do Eclipse
- http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/galileo/SR2/eclipse-java-galileo-SR2-win32.zip
- Instalar no Eclipse o ADT (Android Development Tools), plugin para Desenvolvimento de Aplicações com Android

Ambiente de Desenvolvimento

- Passos para a instalação do ADT:

1. Dentro do Eclipse, aponte para Help -> Software Updates -> Install new Software....
2. Clique no botão Add.
3. Na caixa de diálogo, digite uma descrição no campo Name (por exemplo, Android) e digite o seguinte no campo URL: https://dl-ssl.google.com/android/eclipse/ . Pressione OK.
4. Vá prosseguindo a instalação clicando em Next.
5. Na última tela você perceberá que o ADT não é assinado. Mesmo assim, você pode aceitar a instalação clicando em Install All.
6. Reinicie o Eclipse.

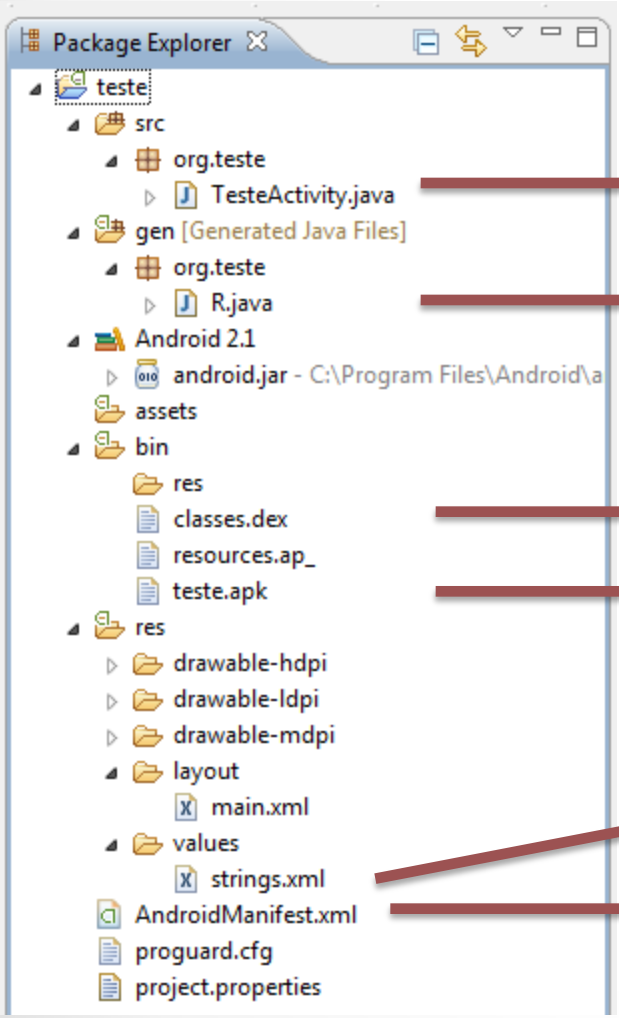


Vamos conferir o que temos instalado!

Criand o projeto

- Abra o
  - File> N roject
  - Escolh 4.0 como alvo
  - Coloq
- Definin
  - Clique nager
  - Defino ersão 4.0
- Execute

Analisando o que foi gerado



Sua classe

Classe gerada para inserção dos recursos

Código Binário

Pacote de Instalação

Valor das strings

Definição das Permissões

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.teste"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="7" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <activity
            android:label="@string/app_name"
            android:name=".TesteActivity" >
            <intent-filter >
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

String.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="hello">Hello World!!</string>
    <string name="app_name">Teste</string>

</resources>]
```

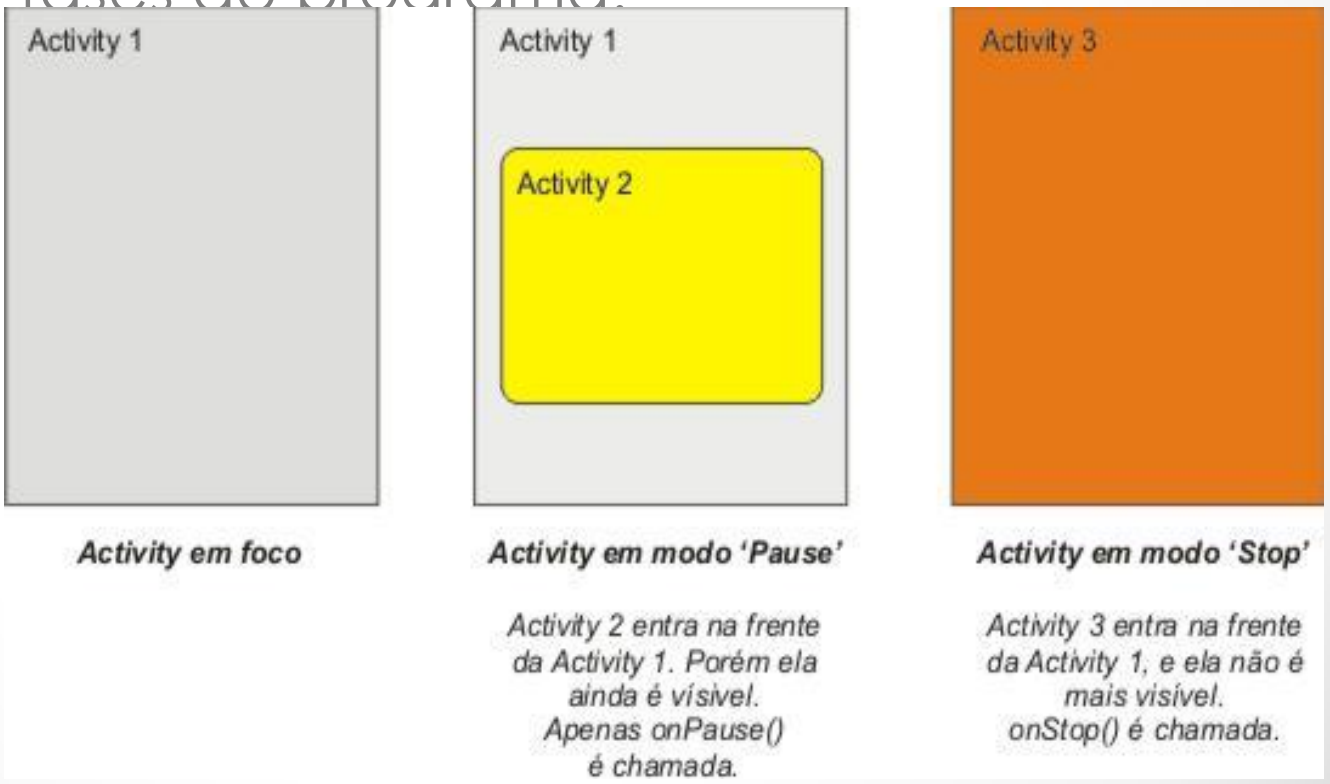
R.java

```
/* AUTO-GENERATED FILE.  DO NOT MODIFY.
*/

package org.teste;
public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int
ic_launcher=0x7f020000;
    }
    public static final class layout {
        public static final int main=0x7f030000;
    }
    public static final class string {
        public static final int app_name=0x7f040001;
        public static final int hello=0x7f040000;
    }}
```

Activity

- Uma Activity (ou Atividade) representa uma interface gráfica com o usuário. Aplicações podem definir uma ou mais atividades para manipular diferentes fases do programa.

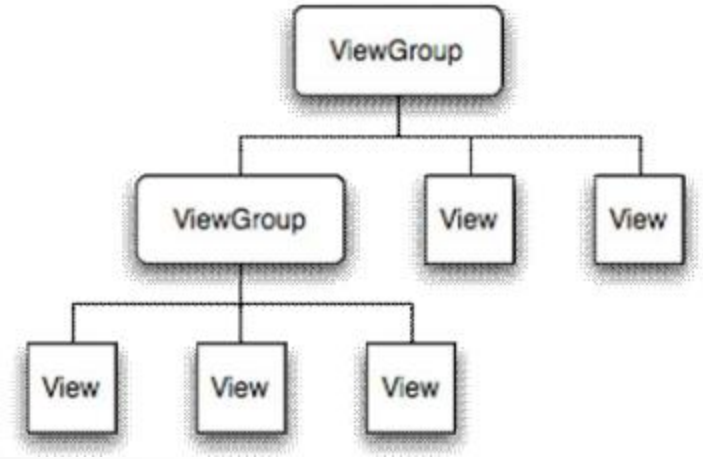


Activity

```
package org.teste;
import android.app.Activity;
import android.os.Bundle;
public class TesteActivity extends Activity {
/** Called when the activity is first created. */
@Override
 public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
}
}
```

Interface gráfica e layouts

- Principais Componentes Gráficos:
  - Views e Viewgroups.



Componente View

- Área retangular limitada da tela do dispositivo.
- Podemos obter as seguintes informações
  - medidas de largura e altura, desenho, mudança de foco, capacidade de rolagem (scrolling) e captura de comandos (teclas dos dispositivos) percebida por aquela área específica.
- Componente possui vários Widgets

Componentes

- Widgets disponíveis: Text, EditText, InputMethod*, MovementMethod, Button, RadioButton, Checkbox, ScrollView.
- ViewGroup:
  - Container de Views.
  - Criar estruturas mais complexas, ricas e robustas.
  - Classe base para outros Layouts:LinearLayout, RelativeLayout, AbsoluteLayout, etc

Interface gráfica e layouts

- Na estrutura de arquivos do seu projeto, crie em res>layout> main.xml

Trabalhando com Spinner

- Como adicionar opções no spinner:

- Descubram vocês mesmos!
- Link: http://developer.android.com/resources/tutorials/views/hello-spinner.html

Eventos

- Similar as aplicações Java, Android oferece um série de Listeners para tratar eventos baseados em notificações
  - OnClickListener, OnLongClickListener, OnTouchListener
- Passos para uso
  - Declare uma classe que implementa uma dessas interface
  - Indique no componente a ser monitorado quem deve ser seu listener

Eventos - Exemplo

- public class TesteActivity extends Activity implements **OnClickListener** {
- Button btcalcular;
- …
- btcalcular.setOnClickListener(**this**);
- …
- @Override
- public void onClick(View v) { //Código de //gerenciamento do evento
- If(v==btcalcular){ }
- }

Acessando os Componentes da UI

- Criação de varíaveis “ponteiros”
- Uso do método findViewById
- Exemplo:

```
btcalcular=(Button)findViewById(R.id.button1);
oper1=(EditText)findViewById(R.id.editText1);
result=(TextView)findViewById(R.id.textView4);
```

- **Complete a calculadora para que ela funcione!**

Dicas do Dia

Ctrl+F12 muda a orientação do emulador!

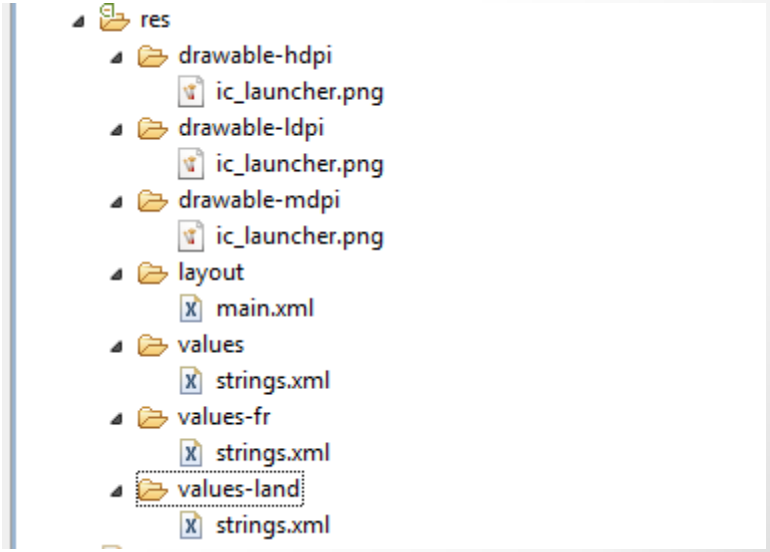
Inicie manualmente o emulador (Run Configuration) para mudar o tamanho da tela

Adaptando a Interface apenas com XML

- O Android oferece uma lista qualificadores que podem ser utilisados para criar versões de recursos
  - Internacionalização
  - Mudança de orientação da Tela
  - Disponibilidade de Teclado ou Trackball
  - Resolução da tela
  - …

Adaptando a Interface apenas com XML

- Para utilizar esses "qualifiers" basta criar pastas distintas dentro de resources e colocar arquivos xml com o mesmo nome em cada pasta
  - values > string.xml
  - values-fr > string.xml
  - values-land > string.xml

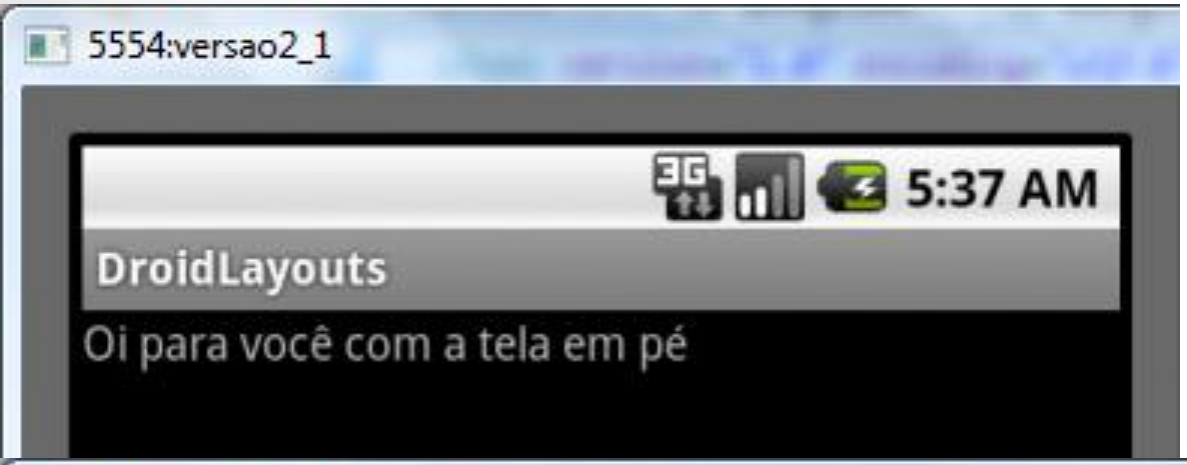


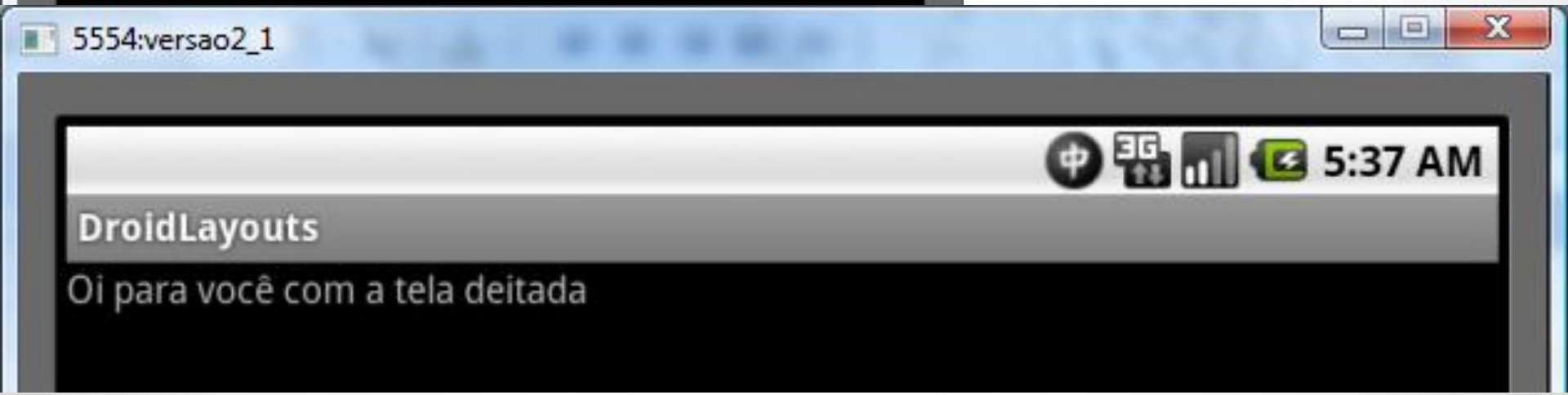
Adaptando a Interface apenas com XML

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Oi para você com a tela em pé</string>
    <string name="app_name">DroidLayouts</string>
</resources>
```

```
<?xml version="1.0" encoding="utf-8"?>
    <resources>
        <string name="hello">Oi para você com a tela
deitada</string>
    </resources>
```

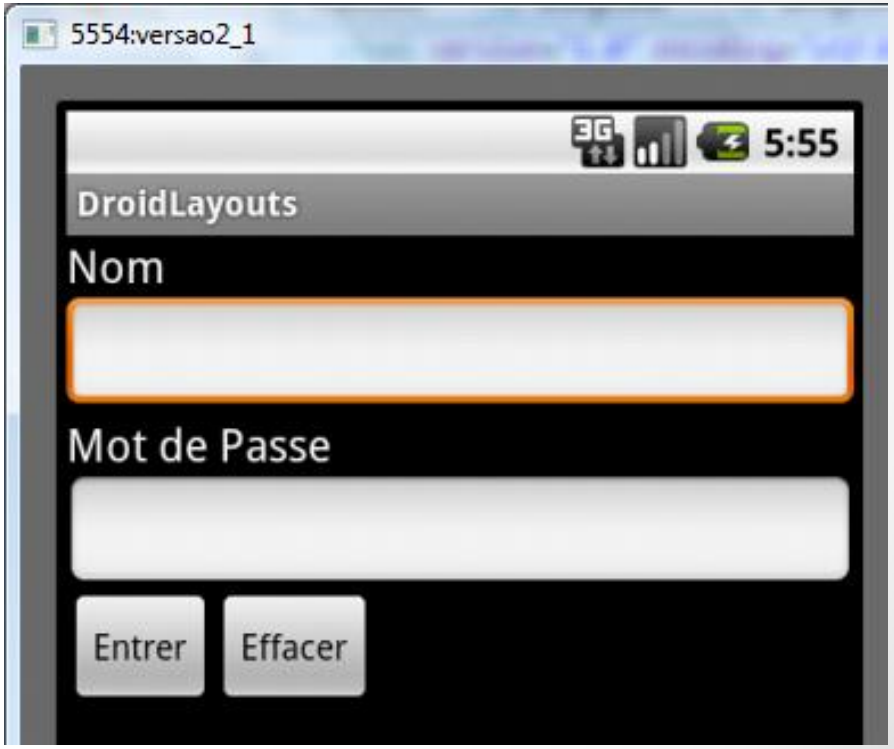
Adaptando a Interface apenas com XML





Exercício

Mudança de Língua



“reconfigurações” no código

- Adição dos qualifiers no AndroidManifest

```
<application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <activity
            android:label="@string/app_name"
            android:name=".DroidLayoutsActivity"
            android:configChanges="orientation">
```

“reconfigurações” no código

- Uso do método onConfigurationChanges

```
@Override
    public void onConfigurationChanged(Configuration _newConfig){
    super.onConfigurationChanged(_newConfig);
    if (_newConfig.orientation==Configuration.ORIENTATION_LANDSCAPE){
        Toast t=Toast.makeText(this, "Deitado!", Toast.LENGTH_LONG);
      t.show();}
    if (_newConfig.orientation==Configuration.ORIENTATION_PORTRAIT){
        Toast t=Toast.makeText(this, “Em pé!", Toast.LENGTH_LONG);
        t.show();
     } }
```

Vamos Testar!