PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Internacionalização e Literais

Professor: Danilo Giacobo

OBJETIVOS DA AULA

- Aprender as vantagens do uso de literais e da internacionalização em aplicativos Android.
- Saber como rodar um aplicativo Android em diferentes idiomas de aparelhos celulares, sem a necessidade de mudar o código-fonte ou recompilar o aplicativo.
- Customizar o aplicativo em relação as suas funcionalidades como o idioma do mesmo.

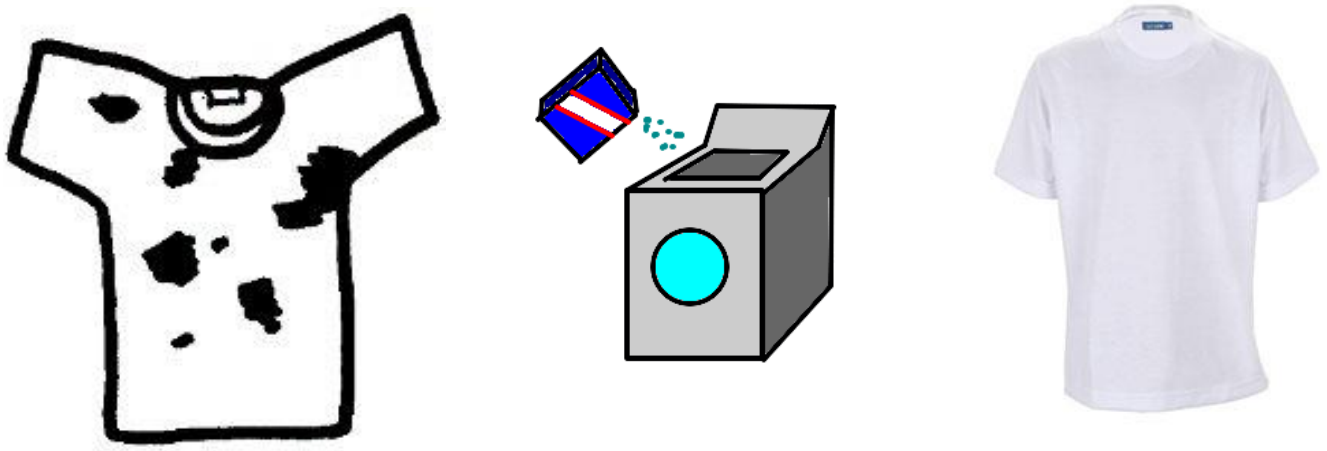
INTRODUÇÃO

- O processo de desenvolvimento de aplicativos evoluiu muito nas últimas décadas.
- Antigamente: ganhar dinheiro de forma isolada com aplicativos simples no mercado nacional.
- Hoje: processo de desenvolvimento mudou, plataformas mudaram e a forma de fazer dinheiro também.
- Uma pessoa hoje pode contar com smartphones, tablets, TV digital, canetas inteligentes, carros com computadores de bordo, laptops, relógios com acesso à Internet, computadores desktops e laptops.
- Esses dispositivos precisam executar aplicativos e este estão disponíveis na rede, de forma gratuita ou paga, ou ainda em portais, como, por exemplo, o Google Play.

INTRODUÇÃO

- Os softwares desenvolvidos para smartphones podem ser acessados e baixados por pessoas de qualquer parte do mundo, que pagarão por estes aplicativos em reais, dólares, euros, libras ou qualquer outra moeda, entretanto, e necessário que esses softwares estejam acessíveis a todos os usuários.
- O termo acessível não se refere apenas em deixar o *apk*, o instalador de um aplicativo Android, disponível para download na Internet. Ser acessível refere-se a tornar o aplicativo usável por diferentes usuários, que falam diferentes idiomas, possuem diferentes culturas, que trabalham com diferentes unidades de medidas.
- Mesmo um aplicativo simples como o do IMC pode e deve ser internacionalizado.
- Até o formato de navegação e leitura de um aplicativo pode causar problemas.

INTRODUÇÃO

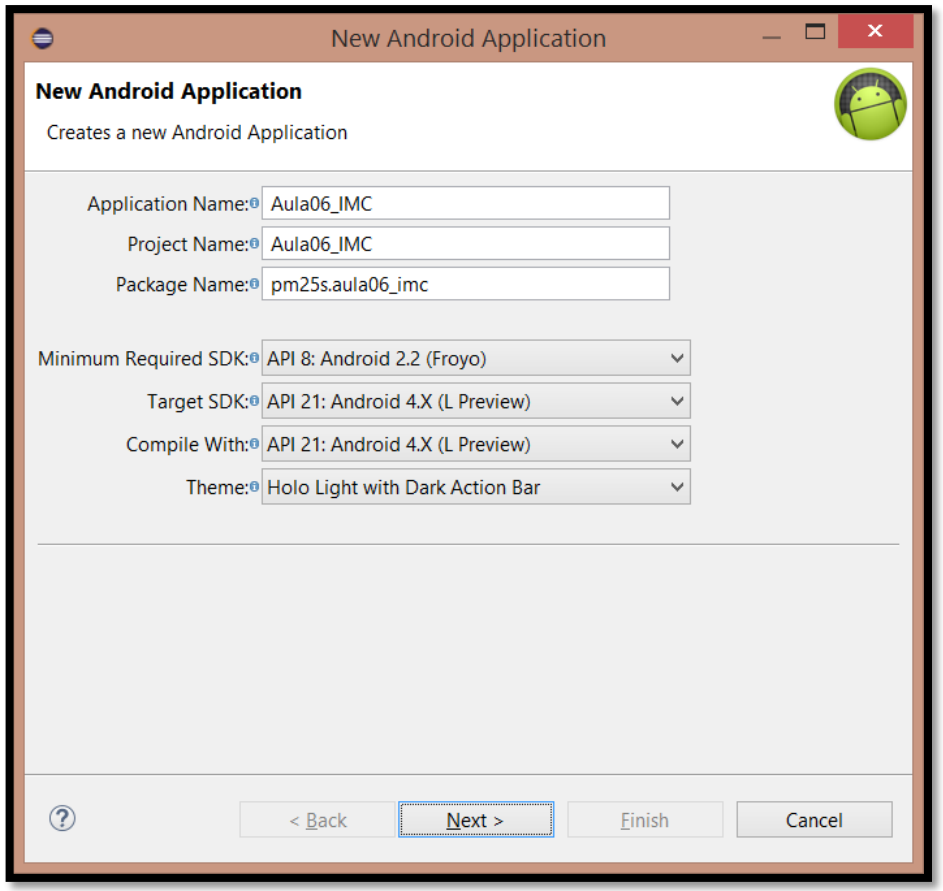


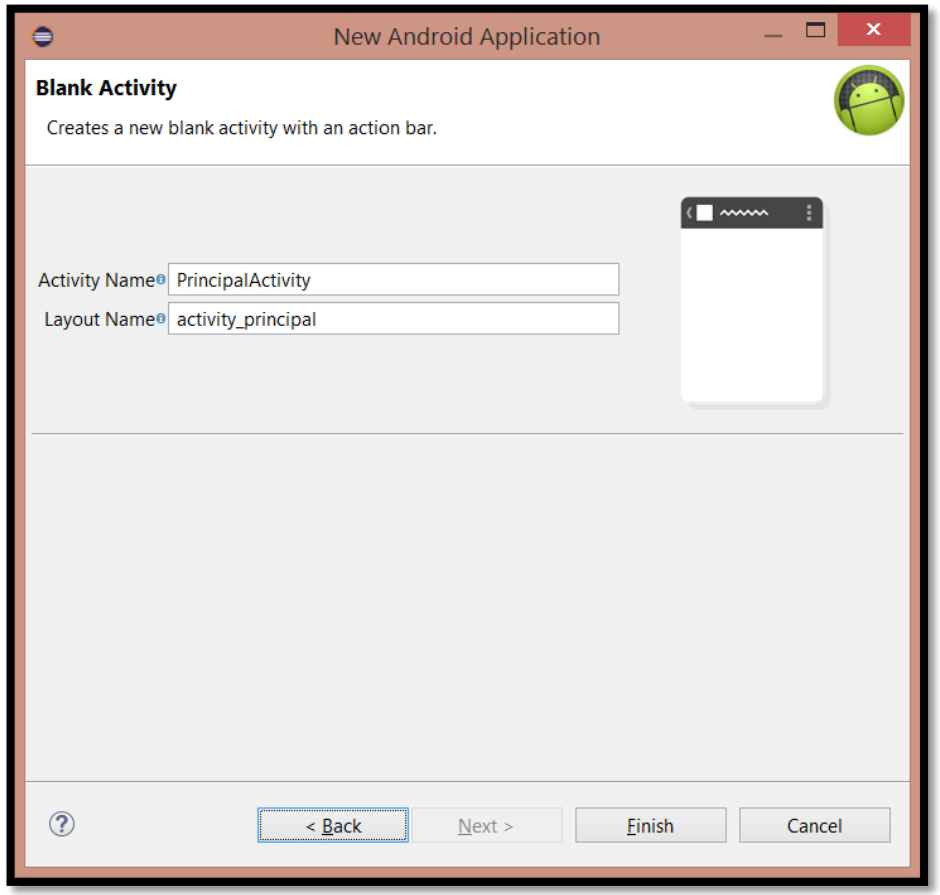
Exemplo típico de imagem que deve ser refeita dependendo da cultura onde será apresentada.

DESENVOLVENDO O APLICATIVO DE EXEMPLO

- Para apresentar os conceitos de internacionalização em Android, será desenvolvido um aplicativo simples de cálculo do IMC, que é o índice de massa corpórea de um indivíduo.
- Para este cálculo, são necessárias informações, tais como, o peso e altura do indivíduo, que no aplicativo serão solicitados via componentes EditText, além de dois botões: um para calcular o IMC e outro para limpar o conteúdo da tela. Por fim, o resultado do IMC será apresentado em um componente TextView.
- O projeto desenvolvido terá o nome de **Aula06_IMC**, e deve ser criado no menu **File > New > Android Application Project**. A tela para a configuração do projeto é apresentada no slide seguinte.

ESTUDO DE CASO - CALCULAR O IMC





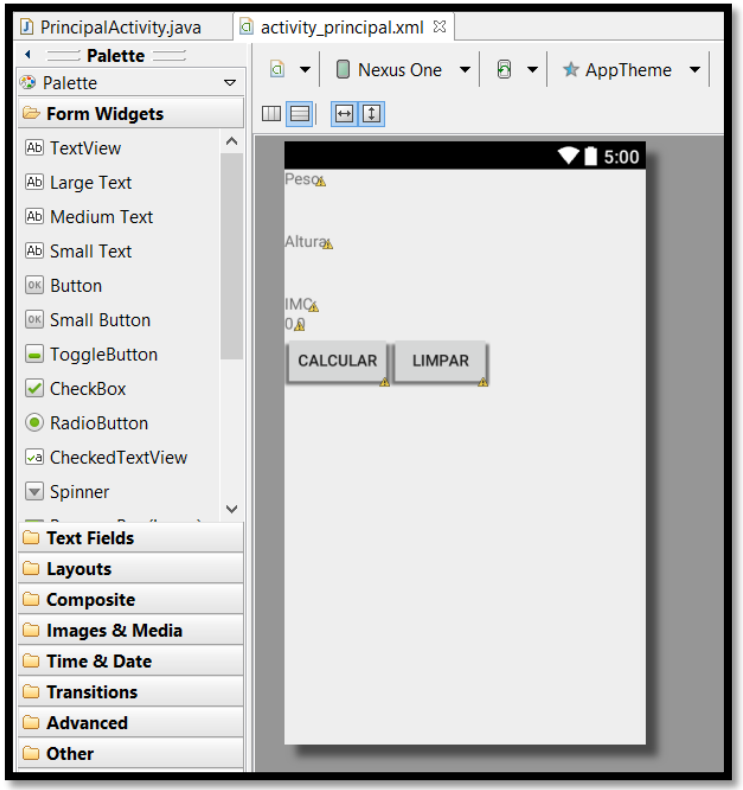
ESTUDO DE CASO - CALCULAR O IMC

- O arquivo activity_principal.xml possui o layout apresentado abaixo:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    tools:context="br.edu.utfpr.imc.PrincipalActivity" >

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

    <EditText
        android:id="@+id/etPeso"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="numberDecimal"
        android:text="" />

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

    <EditText
        android:id="@+id/etAltura"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="numberDecimal"
        android:text="" />

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

    <TextView
        android:id="@+id/tvResult"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="0,0" />

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <Button
            android:id="@+id/btCalcular"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Calcular" />

        <Button
            android:id="@+id/btLimpar"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Limpar" />

    </LinearLayout>

</LinearLayout>
```

ESTUDO DE CASO - CALCULAR O IMC

- Se optarmos pelo modo de visualização Graphical Layout, na parte inferior esquerda do código do activity_principal.xml, veremos o layout conforme figura abaixo:



ESTUDO DE CASO - CALCULAR O IMC

- Embora a criação da interface utilizando literais diretamente no código XML seja mais prática e rápida, esta não é a melhor opção. O próprio plug-in ADT informa como *warning* a utilização de literais, como pode ser observado na imagem abaixo.

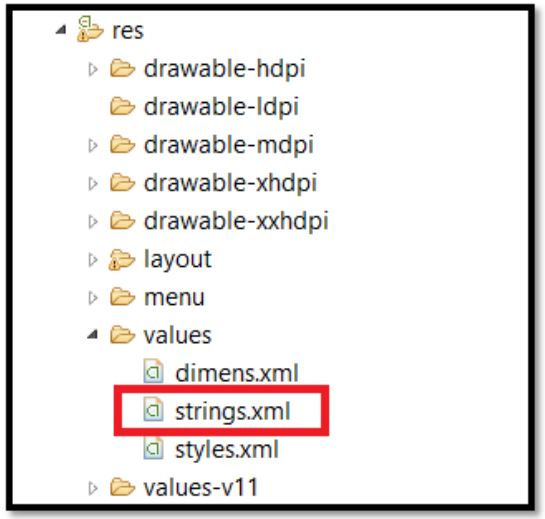
```
 8     <TextView
 9         android:layout_width="wrap_content"
10         android:layout_height="wrap_content"
11         android:text="Peso: " />
12
13     <EditText
14         android:id="@+id/etPeso"
15         android:layout_width="wrap_content"
16         android:layout_height="wrap_content"
17         android:inputType="numberDecimal"
18         android:text="" />
19
20     <TextView
21         android:layout_width="wrap_content"
22         android:layout_height="wrap_content"
23         android:text="Altura: " />
24
25     <EditText
```

TRABALHANDO COM LITERAIS NO ANDROID

- As vantagens de utilizar um “repositório central de literais” em uma aplicação Android são muitas.
- Uma delas é a possibilidade de manter em um mesmo arquivo (XML) todos os textos e mensagens de seu aplicativo, proporcionando assim uma fácil verificação gramatical e ortográfica.
- Outra vantagem está na facilidade de padronizar os termos. Utilizando um repositório de literais, é possível percorrer todos os textos e verificar se algum termo está fora do padrão do aplicativo, deixando o aplicativo com um aspecto mais profissional.
- Por fim, a terceira e maior vantagem de utilizar um repositório de literais é a facilidade para internacionalizar o aplicativo, permitindo traduzi-lo para vários idiomas, modificando somente o arquivo de literais, não havendo a necessidade de modificar o código-fonte.

TRABALHANDO COM LITERAIS NO ANDROID

- Analisando o código digitado anteriormente, temos uma série de literais (“Peso:”, “Altura:”, “IMC:”, “0.0”, “Calcular” e “Limpar”).
- Podemos colocar todo este conteúdo em um repositório de literais,
- No Android, esse repositório costuma ser o arquivo strings.xml, sendo que ele se encontra na pasta res, subpasta values, conforme imagem abaixo:



TRABALHANDO COM LITERAIS NO ANDROID

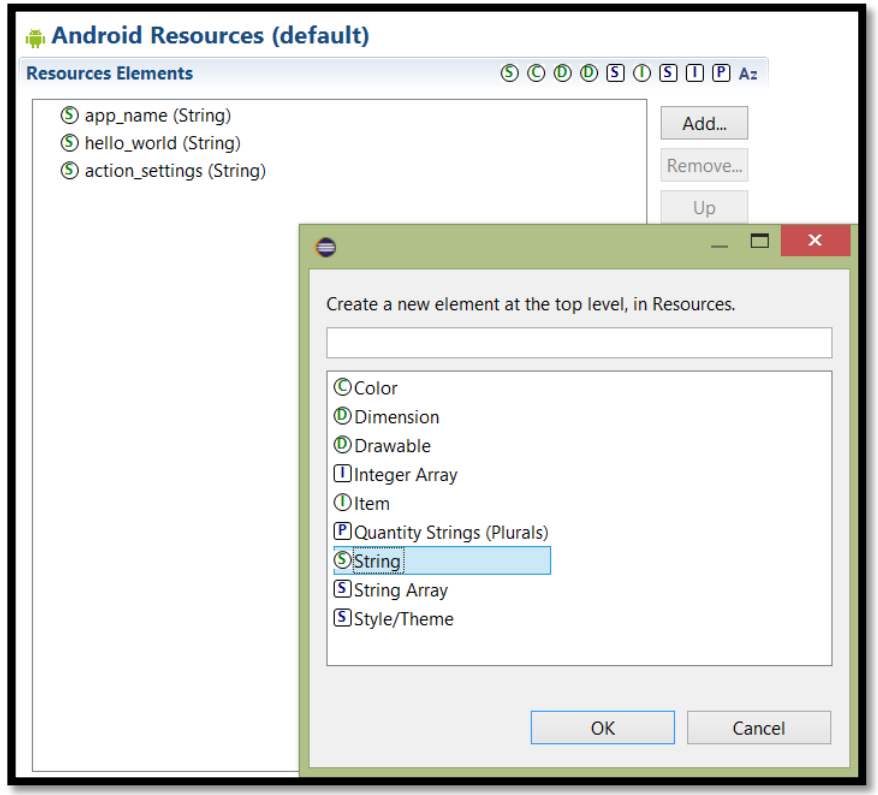
- Abrindo este arquivo, algumas literais do aplicativo já existem, como, por exemplo, o nome da aplicação (app_name).
- Nós iremos editar este arquivo para adicionar novas literais. Isto pode ser feito no editor visual de arquivo XML, no modo de visualização Resource (canto inferior direito do arquivo strings.xml) ou então diretamente no código XML.

PrincipalActivity.java | activity_principal.xml | strings.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3
4     <string name="app_name">Aula06_IMC</string>
5     <string name="hello_world">Hello world!</string>
6     <string name="action_settings">Settings</string>
7
8 </resources>
```

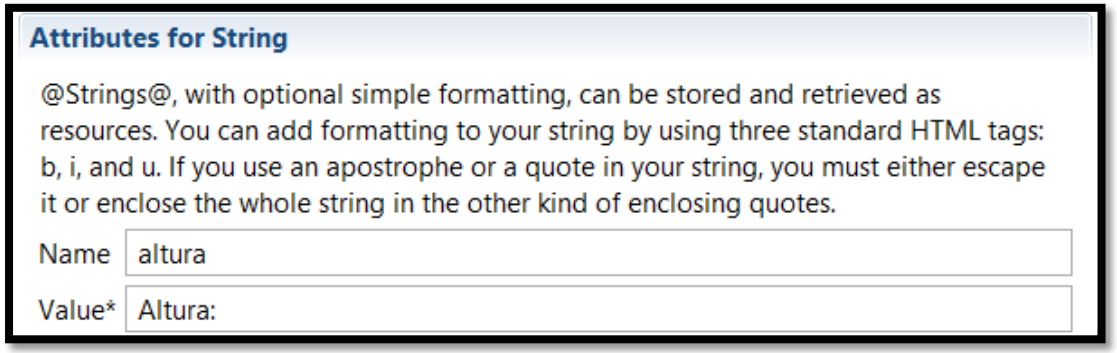
TRABALHANDO COM LITERAIS NO ANDROID

- Para adicionar um literal no modo Resource, basta clicar no botão **Add...**, escolhendo a opção String.



TRABALHANDO COM LITERAIS NO ANDROID

- Na tela apresentada, ao lado direito, deve-se informar o nome da *tag* pelo qual a literal será referenciada dentro do aplicativo (campo Nome), assim como o conteúdo da literal (campo Value), conforme imagem abaixo.




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

    <string name="app_name">Aula06_IMC</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>
    <string name="altura">Altura:</string>

</resources>
```

- Essa mudança reflete-se no conteúdo do arquivo strings.xml.

TRABALHANDO COM LITERAIS NO ANDROID

- Adicione agora as demais literais dentro do arquivo, deixando-o conforme o código abaixo:

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

    <string name="app_name">Aula06_IMC</string>
    <string name="altura">Altura:</string>
    <string name="peso">Peso:</string>
    <string name="imc">IMC:</string>
    <string name="zeros">0.0</string>
    <string name="calcular">Calcular</string>
    <string name="limpar">Limpar</string>
    <string name="erroaltura">Campo [Altura] deve ser preenchido</string>
    <string name="erropeso">Campo [Peso] deve ser preenchido</string>
    <string name="action_settings">Settings</string>

</resources>
```

- Como pode ser observado, além do nome do aplicativo e das literais de tela, duas novas mensagens foram adicionadas para ser usadas posteriormente pelo aplicativo.

TRABALHANDO COM LITERAIS NO ANDROID

Dica: Literais criadas com o aplicativo

- Ao iniciar um aplicativo Android, algumas literais já foram criadas no strings.xml, como, por exemplo, as literais do menu da aplicação e uma literal hello_world, que é apresentada em um TextView adicionado na tela na criação de um projeto novo. Essas literais, bem como seus respectivos componentes (menus e TextView), foram retiradas do programa já que não foram utilizadas.

- Para utilizar o conteúdo do repositório de literais dentro do arquivo activity_principal.xml, é necessário onde se utilizava uma literal, fazer referência ao arquivo strings.xml e ao nome da *tag* da literal, ficando o arquivo activity_principal.xml conforme apresentado no slide seguinte.

TRABALHANDO COM LITERAIS NO ANDROID

```
1  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2      xmlns:tools="http://schemas.android.com/tools"
3      android:layout_width="fill_parent"
4      android:layout_height="fill_parent"
5      android:orientation="vertical"
6      tools:context="pm25s.aula06_imc.PrincipalActivity" >
7
8      <TextView
9          android:layout_width="wrap_content"
10         android:layout_height="wrap_content"
11         android:text="@string/peso" />
12
13     <EditText
14         android:id="@+id/etPeso"
15         android:layout_width="wrap_content"
16         android:layout_height="wrap_content"
17         android:inputType="numberDecimal"
18         android:text="" />
19
20     <TextView
21         android:layout_width="wrap_content"
22         android:layout_height="wrap_content"
23         android:text="@string/altura" />
24
25     <EditText
26         android:id="@+id/etAltura"
27         android:layout_width="wrap_content"
28         android:layout_height="wrap_content"
29         android:inputType="numberDecimal"
30         android:text="" />
```

```
32     <TextView
33         android:layout_width="wrap_content"
34         android:layout_height="wrap_content"
35         android:text="@string/imc" />
36
37     <TextView
38         android:id="@+id/tvResult"
39         android:layout_width="wrap_content"
40         android:layout_height="wrap_content"
41         android:text="@string/zeros" />
42
43     <LinearLayout
44         android:layout_width="wrap_content"
45         android:layout_height="wrap_content"
46         android:orientation="horizontal">
47
48         <Button
49             android:id="@+id/btCalcular"
50             android:layout_width="wrap_content"
51             android:layout_height="wrap_content"
52             android:text="@string/calcular" />
53
54         <Button
55             android:id="@+id/btLimpar"
56             android:layout_width="wrap_content"
57             android:layout_height="wrap_content"
58             android:text="@string/limpar" />
59
60     </LinearLayout>
61
62 </LinearLayout>
```

LITERAIS NO CÓDIGO DA ACTIVITY

- O aplicativo para o cálculo do IMC está gratificante pronto, porém, não foi codificada ainda a classe PrincipalActivity.java, a qual recuperará os componentes visuais e os conteúdos digitados para o cálculo do IMC.
- A lógica da classe PrincipalActivity.java está presente nos dois slides seguintes.
- Como pode ser observado ao longo do código-fonte, os comandos presentes no Java podem ser utilizados em aplicações Android, como, por exemplo, o comando para converter *String* em *double* (Double.parseDouble) ou mesmo o método para elevar um número à segunda potência (Math.pow).

LITERAIS NO CÓDIGO DA ACTIVITY

- A *Activity*, a classe Java responsável pelo funcionamento do aplicativo, é mostrada abaixo:

```java
package br.edu.utfpr.imc;

import java.text.DecimalFormat;

import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class PrincipalActivity extends ActionBarActivity {

    private EditText etPeso;
    private EditText etAltura;
    private TextView tvResult;
    private Button btCalcular;
    private Button btLimpar;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_principal);
```

```java
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_principal);

        etPeso = (EditText) findViewById(R.id.etPeso);
        etAltura = (EditText) findViewById(R.id.etAltura);
        tvResult = (TextView) findViewById(R.id.tvResult);
        btCalcular = (Button) findViewById(R.id.btCalcular);
        btLimpar = (Button) findViewById(R.id.btLimpar);

        btLimpar.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {
                btLimparOnClick();
            }
        });

        btCalcular.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {
                btCalcularOnClick();
            }
        });
    }
```

LITERAIS NO CÓDIGO DA ACTIVITY

- A *Activity*, a classe Java responsável pelo funcionamento do aplicativo, é mostrada abaixo:

```
    private void btLimparOnClick() {
        etPeso.setText("");
        etAltura.setText("");
        tvResult.setText("0,0");
        etPeso.requestFocus();
    }

    private void btCalcularOnClick() {
        if(etPeso.getText().toString().equals("")) {
            Toast.makeText(this, "Campo Peso deve ser preenchido", Toast.LENGTH_LONG).show();
            etPeso.requestFocus();
            return;
        }

        if(etAltura.getText().toString().equals("")) {
            Toast.makeText(this, "Campo Altura deve ser preenchido", Toast.LENGTH_LONG).show();
            etAltura.requestFocus();
            return;
        }

        double peso = Double.parseDouble(etPeso.getText().toString());
        double altura = Double.parseDouble(etAltura.getText().toString());
        double imc = peso / Math.pow(altura, 2);
        tvResult.setText(new DecimalFormat("0.00").format(imc));
    }
}
```

LITERAIS NO CÓDIGO DA ACTIVITY

- O código apresentado anteriormente possui uma série de literais no seu conteúdo, como o conteúdo presente na linha tvResult.setText("0,0"); e nas linhas de código que apresentam mensagens de erro.
- As literais apresentadas podem ser recuperadas também do arquivo strings.xml. Para isso, basta utilizar o comando getString(R.string.nome_da_literal).
- Desta forma, o código dos métodos btLimparOnClick e btCalcularOnClick são apresentados conforme mostra o slide a seguir.
- Com isso, as literais utilizadas pelo código Java também são recuperadas do arquivo strings.xml, sendo este o repositório de literais do aplicativo.

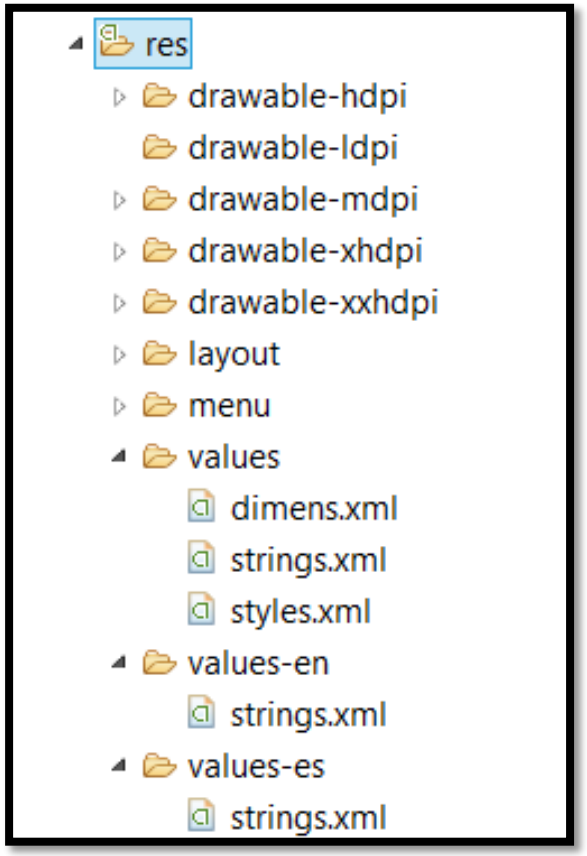
LITERAIS NO CÓDIGO DA ACTIVITY

```
private void btLimparOnCLick() {
    etPeso.setText("");
    etAltura.setText("");
    tvResult.setText(getString(R.string.zeros));
    etPeso.requestFocus();
}

private void btCalcularOnCLick() {
    if(etPeso.getText().toString().equals("")) {
        Toast.makeText(this, getString(R.string.erropeso), Toast.LENGTH_LONG).show();
        etPeso.requestFocus();
        return;
    }

    if(etAltura.getText().toString().equals("")) {
        Toast.makeText(this, R.string.erroaltura, Toast.LENGTH_LONG).show();
        etAltura.requestFocus();
        return;
    }

    double peso = Double.parseDouble(etPeso.getText().toString());
    double altura = Double.parseDouble(etAltura.getText().toString());
    double imc = peso / Math.pow(altura, 2);
    tvResult.setText(new DecimalFormat("0.00").format(imc));
}
```

INTERNACIONALIZANDO APLICAÇÕES ANDROID

- Para internacionalizar o aplicativo desenvolvido até o momento, o primeiro passo é traduzir todas as literais em todos os idiomas desejados.
- Assim, para melhor organizar o arquivo strings.xml, uma dica é criar várias pastas values, uma para cada idioma, assim, a pasta values-pt possuirá as literais no idioma português; já a pasta values-en possuirá as literais em inglês; a pasta values-es em espanhol, e assim por diante.
- Também é aconselhável ter uma pasta values, sendo este o idioma *default* da aplicação, portanto, se o aplicativo rodar em um celular alemão e esse idioma não foi tratado, o conteúdo apresentado para o usuário será o armazenado na pasta values.
- Desta forma, traduzindo o aplicativo para o inglês, espanhol e português, a estrutura de pastas ficará conforme apresentada no slide seguinte.

INTERNACIONALIZANDO APLICAÇÕES ANDROID



Dica: Criando pastas no Android

Para criar um pasta dentro de res, basta clicar com o botão direito na pasta res, escolhendo a opção New > Folder. Na sequencia, o nome da pasta deve ser informado (ex.: values-en). Para facilitar a criação dos arquivos strings.xml, o mesmo pode ser copiado para outra pasta, clicando-o com o botão direito e escolhendo Copiar. Na sequencia, clica-se com o botão direito na pasta de destino, escolhendo a opção Colar.

INTERNACIONALIZANDO APLICAÇÕES ANDROID

- O conteúdo do arquivo strings.xml presenta na pasta values-en é apresentado abaixo:

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

    <string name="app_name">BMI Calculator</string>
    <string name="altura">Height:</string>
    <string name="peso">Weight:</string>
    <string name="imc">BMI:</string>
    <string name="zeros">0.0</string>
    <string name="calcular">Calculate</string>
    <string name="limpar">Clear</string>
    <string name="erroaltura">Field [Height] must be filled</string>
    <string name="erropeso">Field [Weight] must be filled</string>
    <string name="action_settings">Settings</string>

</resources>
```

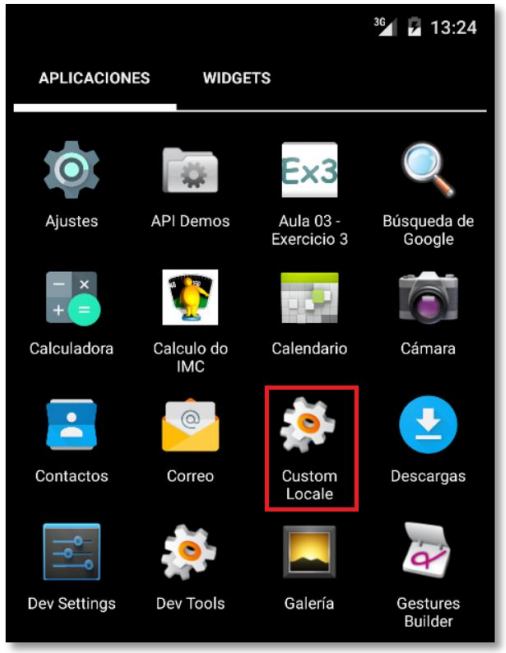
INTERNACIONALIZANDO APLICAÇÕES ANDROID

- O conteúdo do arquivo strings.xml presenta na pasta values-es é apresentado abaixo:

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

    <string name="app_name">Calculo do IMC</string>
    <string name="altura">Altura:</string>
    <string name="peso">Peso:</string>
    <string name="imc">IMC:</string>
    <string name="zeros">0.0</string>
    <string name="calcular">Calcular</string>
    <string name="limpar">Limpiar</string>
    <string name="erroaltura">Campo [Altura] debe ser llenado</string>
    <string name="erropeso">Campo [Peso] debe ser llenado</string>
    <string name="action_settings">Ajustes</string>

</resources>
```

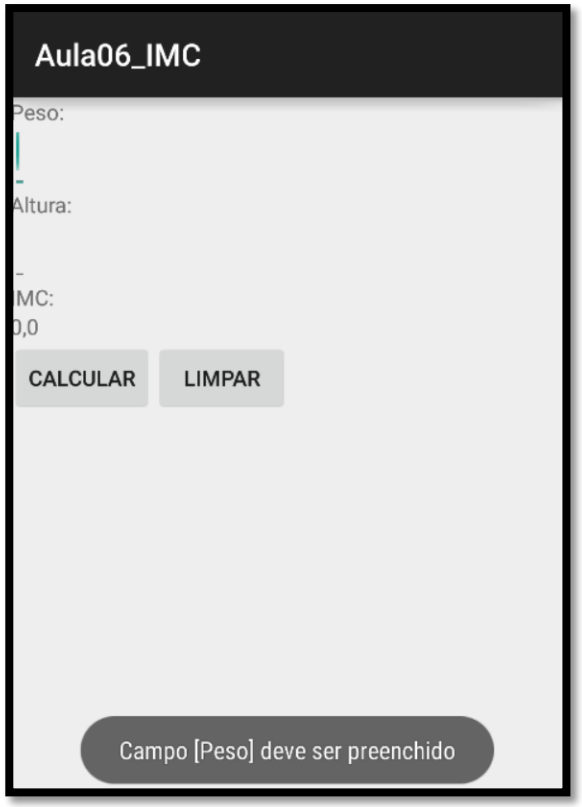
INTERNACIONALIZANDO APLICAÇÕES ANDROID

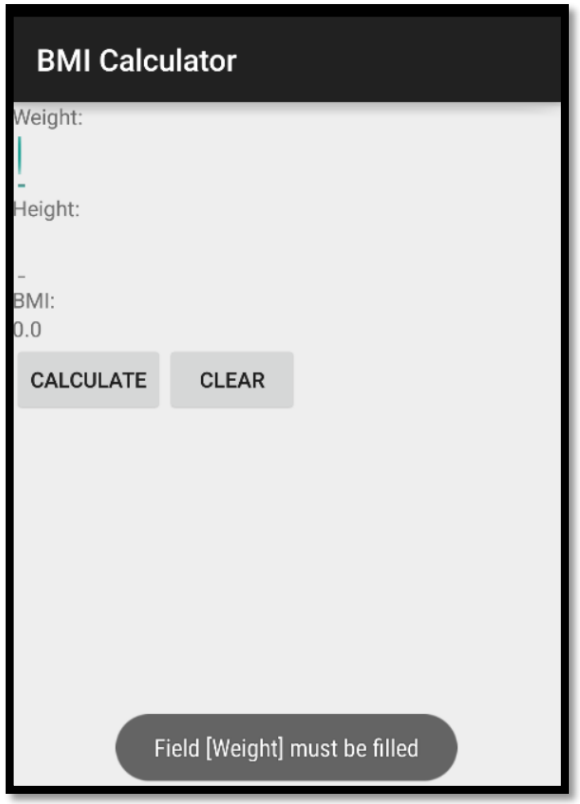
- Após, executando o aplicativo, é visualizado no emulador o mesmo resultado, como se todas as literais estivessem no próprio arquivo de layout. A diferença acontece quando mudamos o idioma do emulador, acessando as configurações do dispositivo Android. Na lista de opções, deve-se escolher o idioma desejado.

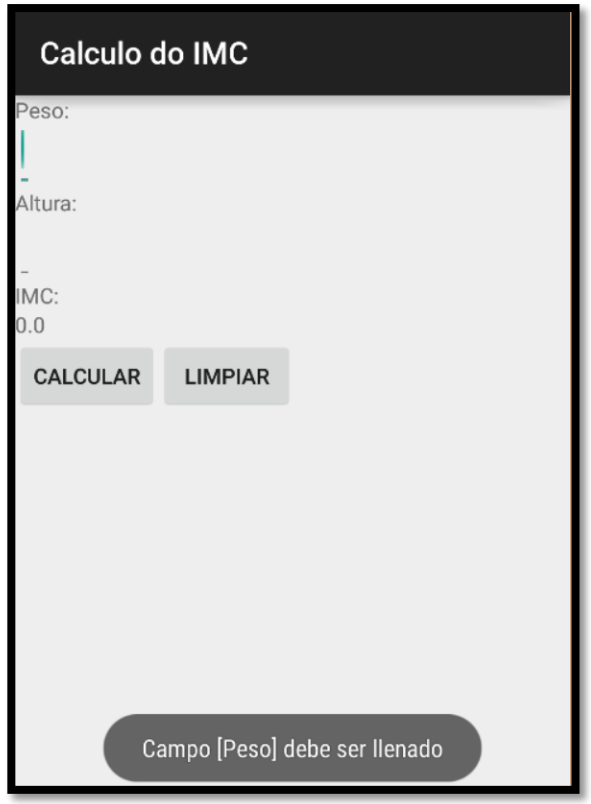


INTERNACIONALIZANDO APLICAÇÕES ANDROID

- Nas imagens abaixo é possível ver o aplicativo sendo executado nos três idiomas definidos.







INTERNACIONALIZANDO APLICAÇÕES ANDROID

- Temos ainda de tratar a internacionalização do presente aplicativo. Será necessário personalizar a fórmula, uma vez que a maioria dos países trabalha com o sistema internacional de medidas (neste, o comprimento é medido em cm e o peso é medido em grama).
- Porém, alguns países como os Estados Unidos não utilizam esse padrão, preferindo polegadas para medir o comprimento e *pounds* para o peso. Assim, a fórmula do IMC para os americanos é um pouco diferente, conforme apresentado na fórmula:

$$BMI = \frac{weight(lb) \times 703}{(height(in))^2}$$

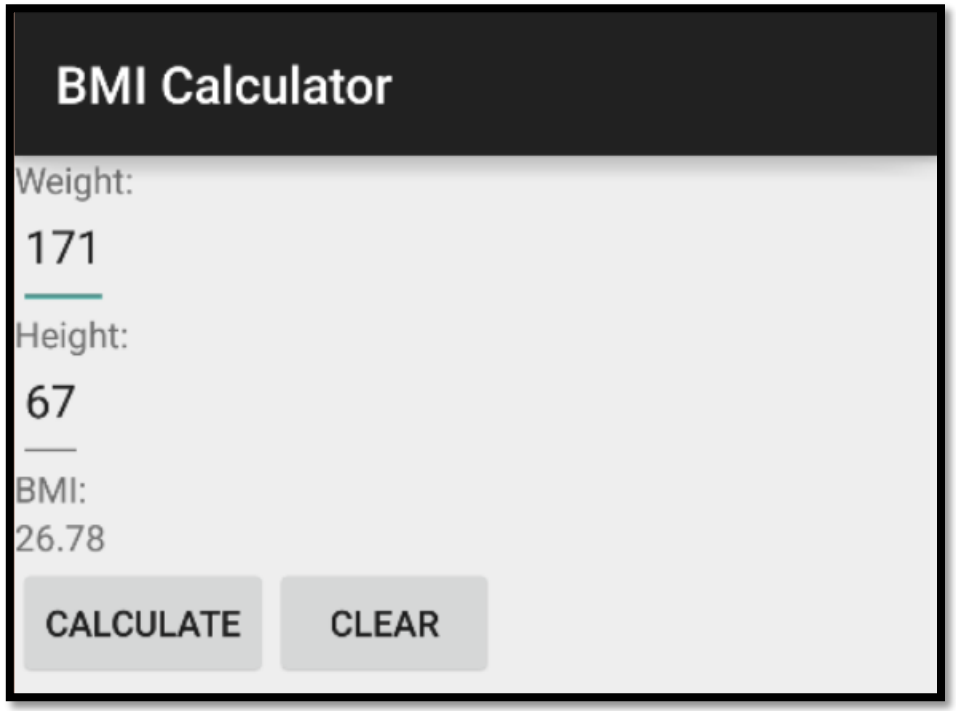
INTERNACIONALIZANDO APLICAÇÕES ANDROID

- Para personalizar a lógica do botão Calcular, pode-se recuperar o idioma do dispositivo durante a execução utilizando o comando Locale.getDefault().getLanguage().
- Assim, a lógica do botão "Calcular" fica conforme código abaixo:

```
private void btCalcularOnCLick() {
    if(etPeso.getText().toString().equals("")) {
        Toast.makeText(this, getString(R.string.erropeso), Toast.LENGTH_LONG).show();
        etPeso.requestFocus();
        return;
    }

    if(etAltura.getText().toString().equals("")) {
        Toast.makeText(this, R.string.erroaltura, Toast.LENGTH_LONG).show();
        etAltura.requestFocus();
        return;
    }

    double peso = Double.parseDouble(etPeso.getText().toString());
    double altura = Double.parseDouble(etAltura.getText().toString());
    double imc = 0;

    if(Locale.getDefault().getLanguage().equals("en"))
        imc = (peso * 703) / (Math.pow(altura, 2));
    else
        imc = peso / Math.pow(altura, 2);

    tvResult.setText(new DecimalFormat("0.00").format(imc));
}
```

INTERNACIONALIZANDO APLICAÇÕES ANDROID

- Usando o aplicativo em português e inglês temos o seguinte resultado:



INTERNACIONALIZANDO APLICAÇÕES ANDROID

- Por fim, pode-se também personalizar a máscara dos campos, tais como, a vírgula como deparador decimal para o aplicativo rodando no Brasil ou o ponto para o aplicativo rodando nos Estados Unidos. Para isso realize as alterações abaixo no método btCalcularOnClick antes de exibir o resultado na tela:

```
NumberFormat nf = NumberFormat.getNumberInstance(Locale.getDefault());
DecimalFormat df = (DecimalFormat) nf;
tvResult.setText(df.format(imc));
```