PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Tratamento de Eventos

Professor: Danilo Giacobo

OBJETIVOS DA AULA

- Apresentar os modelos mais comuns de tratamento de eventos da plataforma Android.
- Aprender a utilizar o modelo de clique simples de um botão
- Aprender a utilizar o modelo de clique longo de um componente visual
- Conhecer os tipos de implementações de eventos na plataforma Android

FORMAS DE TRATAMENTO DE EVENTOS

- A interface de programação do Android, de forma resumida, apresenta oito formas de tratamento de eventos com o usuário:
  - ✓ Clique
  - ✓ Clique longo
  - ✓ Menu de contexto
  - ✓ Evento de toque
  - ✓ Mudança de foco
  - ✓ Evento de tecla
  - ✓ Item selecionado
  - ✓ Eventos automáticos
- Como estudo de caso, será utilizado o exemplo de cálculo do IMC desenvolvido na aula anterior, codificando os eventos de clique nos botões Calcular e Limpar.

FORMAS DE TRATAMENTO DE EVENTOS

- A lógica do tratamento de eventos é feita via código Java (classe *Activity*).
- É necessário atribuir um nome a cada componente que irá participar do tratamento de eventos bem como aqueles que terão seu estado/comportamento alterado após o evento.
- A nomeação dos componentes pode ser feito pela propriedade android:id.
- Esta propriedade deve estar no formato String "@+id/nomedocomponente".
- Esses nomes são mapeados em um arquivo Java, chamado R.java.
- Esse arquivo é criado e gerenciado automaticamente pelo Android, não devendo ser alterado pelo usuário.
- Ele é armazenado na pasta *gen* do projeto.

DECLARAÇÃO E VINCULAÇÃO DOS COMPONENTES

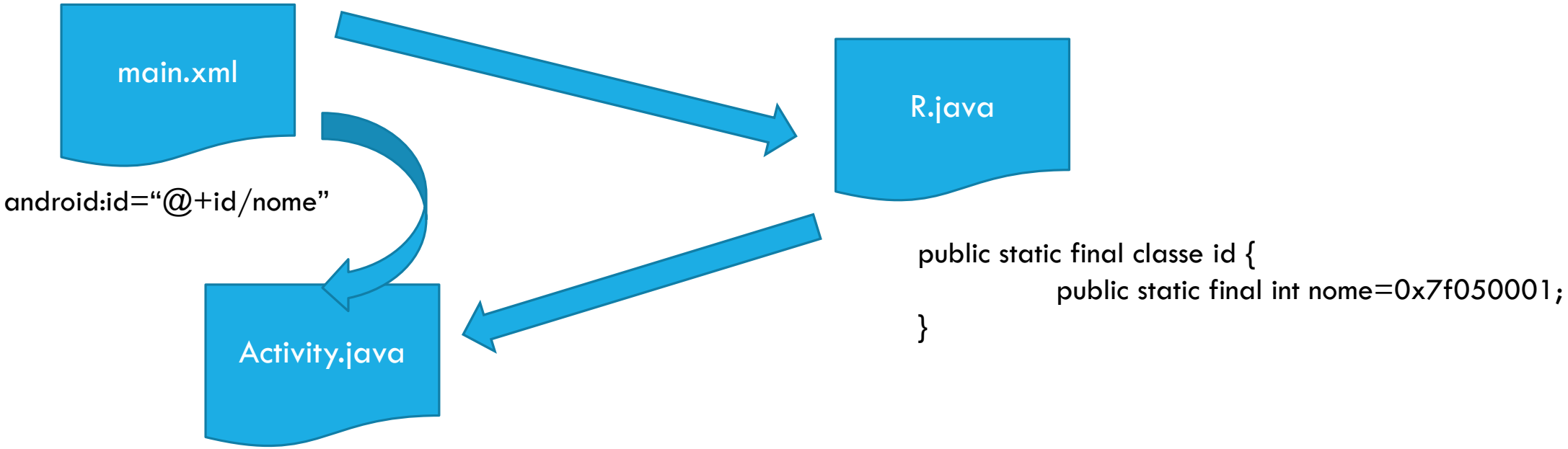
- Para fazer uso dos componentes visuais nomeados no arquivo XML e mapeados no arquivo R.java, é necessário declarar os componentes na classe *Activity* e recuperá-los por meio do comando findViewById.
- No slide seguinte é apresentado o processo descrito acima.
- Você deve importar as classes referentes aos componentes visuais (linhas 5 a 7).
- Cada componente visual é declarado dentro da classe (linhas 11 a 15).
- No método onCreate (linha 18) são referenciados os componentes a partir do comando findViewById (linhas 22 a 26), sendo que o mesmo vincula esse componente ao componente declarado no arquivo XML por meio de R.id.nomecomponente.

DECLARAÇÃO E VINCULAÇÃO DOS COMPONENTES

```java
package pm25s.aula3.imc;

import android.os.Bundle;
import android.app.Activity;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class MainActivity extends Activity {

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

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

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

DECLARAÇÃO E VINCULAÇÃO DOS COMPONENTES

- Uma classe Java pode utilizar os recursos dos componentes visuais declarados em um arquivo XML por meio do arquivo intermediário R.java, conforme figura abaixo:



LISTENERS

- O próximo passo é informar que os componentes botões devem tratar um código se clicados, e isso é feito a partir dos *Listeners*.
- *Listeners* são classes que "escutam" os eventos gerados. Eles podem ser codificados em arquivos separados (arquivos com a extensão .java e que implementam os *Listeners* desejados) ou a partir de classes internas anônimas, sendo que esta última técnica é a mais utilizada.
- Vamos atualizar agora o arquivo .java da *Activity* para incluir o tratamento do evento de clique nos botões da nossa aplicação.

LISTENERS

- Dentro do método onCreate (depois das linhas incluídas anteriormente) adicione as seguintes linhas de código:

```
btCalcular.setOnClickListener(new View.OnClickListener() {

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

btLimpar.setOnClickListener(new View.OnClickListener() {

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

    }
});
```


LISTENERS

- O código abaixo é usado para definir o método btCalcularOnClick().
- Será necessário incluir a linha **import java.text.DecimalFormat;** no começo do código também.

```
protected void btCalcularOnClick() {
    if(etPeso.getText().toString().equals("")) {
        Toast.makeText(getApplicationContext(), "Campo [Peso] deve ser preenchido.", Toast.LENGTH_LONG).show();
        etPeso.requestFocus();
        return;
    }

    if(etAltura.getText().toString().equals("")) {
        Toast.makeText(getApplicationContext(), "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);
    tvResultado.setText(new DecimalFormat("0.00").format(imc));
}
```

LISTENERS

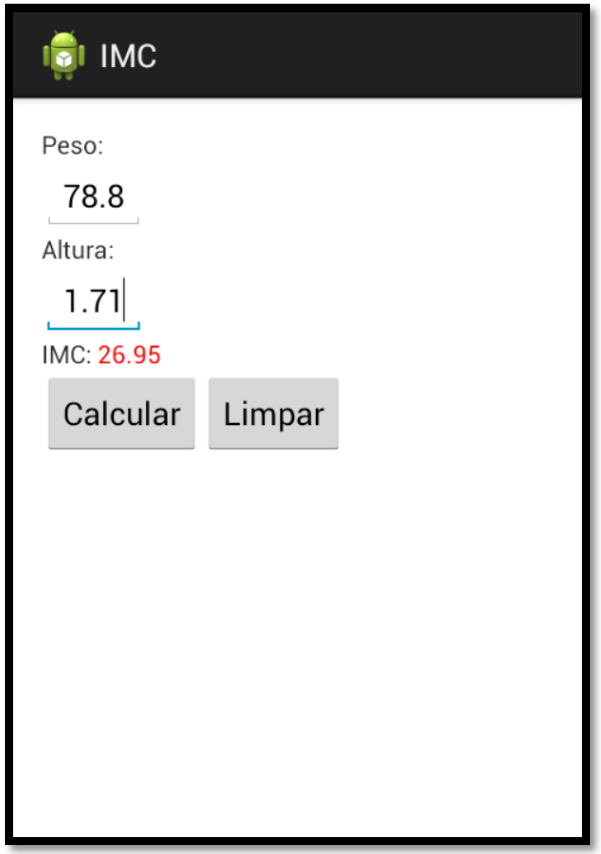
- O código abaixo é usado para definir o método btLimparOnClick().

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

- A fórmula de cálculo do Índice de Massa Corporal é mostrado abaixo:

$$IMC = \frac{\text{peso em kg}}{(\text{altura em m})^2}$$

CÁLCULO DO IMC EM AÇÃO!



SIMPLIFICANDO O TRATAMENTO DO EVENTO

- Na maioria dos programas Android, a interação entre o usuário e o aplicativo acontece por meio do evento de clique em botões, sendo que, neste momento, o programa executará uma tarefa específica. Para simplificar o tratamento deste evento, foi criada a propriedade android:onClick na declaração dos componentes Buttons no arquivo XML.
- Se o programador optar por utilizar esta forma de tratamento, no código da Activity, não será mais necessário criar classes internas anônimas para o View.OnClickListener. Assim, a declaração dos botões deve ser feita conforme mostra o código presente nos slides seguintes.

SIMPLIFICANDO O TRATAMENTO DO EVENTO

- O código abaixo mostra a declaração dos botões com a propriedade onClick.

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

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

- O próximo passo é criar um método público e com uma View de parâmetro para os nomes identificados no método onClick().

SIMPLIFICANDO O TRATAMENTO DO EVENTO

- O código abaixo mostra a codificação dos métodos referenciados via arquivo XML.

```
public void btCalcularOnClick(View v) {
    if(etPeso.getText().toString().equals("")) {
        Toast.makeText(getApplicationContext(), "Campo [Peso] deve ser preenchido.", Toast.LENGTH_LONG).show();
        etPeso.requestFocus();
        return;
    }

    if(etAltura.getText().toString().equals("")) {
        Toast.makeText(getApplicationContext(), "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);
    tvResultado.setText(new DecimalFormat("0.00").format(imc));
}
```

```
public void btLimparOnClick(View v) {
    etPeso.setText("");
    etAltura.setText("");
    tvResultado.setText("0,0");
    etPeso.requestFocus();
}
```

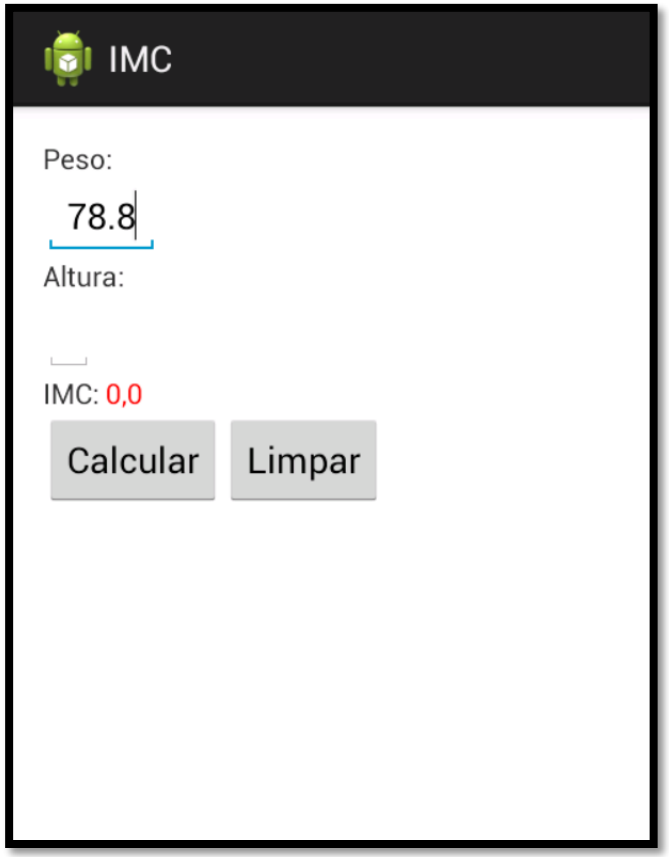
SIMPLIFICANDO O TRATAMENTO DO EVENTO

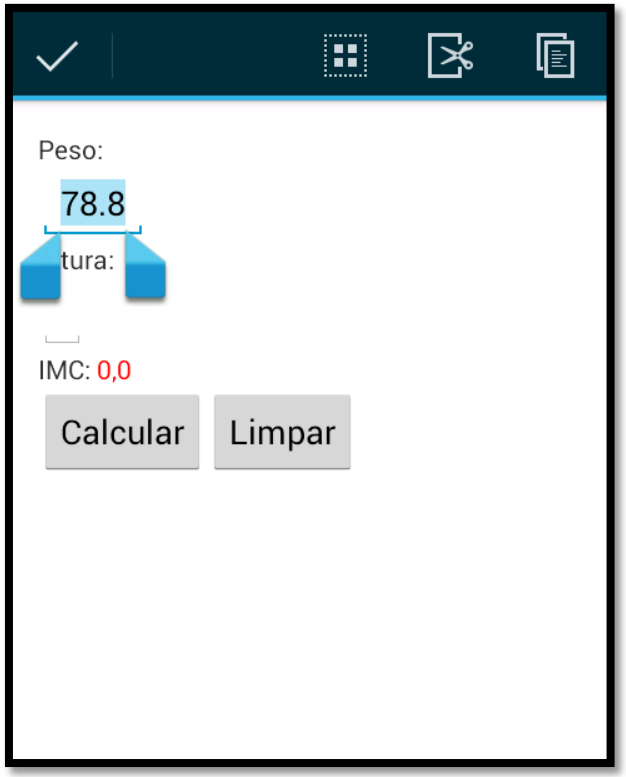
- A diferença básica desta Activity para a anterior é que esta trata os eventos de clique do botão a partir do atributo android:onClick do código XML, o que simplifica consideravelmente o processo, já que na Activity, não é mais necessário declarar os componentes Button, nem mesmo criar classes internas anônimas para tratar o evento de clique (new View.onClickListener).
- Basta apenas codificar os métodos informados pela propriedade android:onClick, devendo estes ser públicos e receber por parâmetro um objeto do tipo View. O código se torna consideravelmente menor nesta nova versão e o funcionamento do aplicativo não se altera.
- A simplificação do tratamento de evento só ocorre para o método onClick de Button, sendo este o modelo de interação mais utilizado pelas aplicações Android. Os demais métodos devem ser codificados no formato tradicional, criando classes internas anônimas para os *listeners* desejados.

ADICIONANDO O EVENTO DE CLIQUE LONGO

- O **clique longo** é um recurso disponível na plataforma Android e analogicamente ele é semelhante ao clique com o botão direito do mouse em uma aplicação *desktop*, apresentando para o usuário opções extras do uso do aplicativo.
- Por padrão, para obter o evento de clique longo é necessário pressionar e manter pressionado em um componente visual por dois segundos.
- Um exemplo típico é quando se clica de modo longo em um EditText, sendo apresentada para o usuário (ao utilizar o emulador) a opção de trabalhar com o conteúdo do texto (área de transferência).
- Este comando não foi tratado pelo programador, ele é implementado automaticamente pela plataforma Android conforme mostra o slide seguinte.

ADICIONANDO O EVENTO DE CLIQUE LONGO





ADICIONANDO O EVENTO DE CLIQUE LONGO

- Para personalizar o clique longo, tratando assim o evento de forma diferente (por exemplo, apresentar uma mensagem informativa via Toast, caso seja clicado de modo longo nos EditTexts), é necessário adicionar o Listener correspondente no código-fonte.

```
etPeso.setOnLongClickListener(new View.OnLongClickListener() {

    @Override
    public boolean onLongClick(View v) {
        Toast.makeText(getApplicationContext(), "Clique longooo no EditText [Peso].", Toast.LENGTH_SHORT).show();
        return true;
    }
});

etAltura.setOnLongClickListener(new View.OnLongClickListener() {

    @Override
    public boolean onLongClick(View v) {
        Toast.makeText(getApplicationContext(), "Clique longooo no EditText [Altura].", Toast.LENGTH_SHORT).show();
        return true;
    }
});
```

EVENTO DE CLIQUE LONGO EM AÇÃO