

# TÓPICOS ESPECIAIS DE PROGRAMAÇÃO ORIENTADA A OBJETOS (TEPOO)

## Aula 4

MSc. Eliana Quiroga

[eliana.quiroga@docente.unip.br](mailto:eliana.quiroga@docente.unip.br)

# Services

- Um dos principais componentes de uma aplicação Android são os *Services*.
- São diferentes das Activities nos seguintes pontos:
  - Não possuem interface com o usuário.
  - Sua execução continua mesmo quando o usuário inicia outra aplicação.
- Por exemplo, um serviço pode:
  - lidar com transações de rede,
  - reproduzir música,
  - executar leitura e escrita de arquivos,
  - interagir com um provedor de conteúdo.

## Services (cont.)

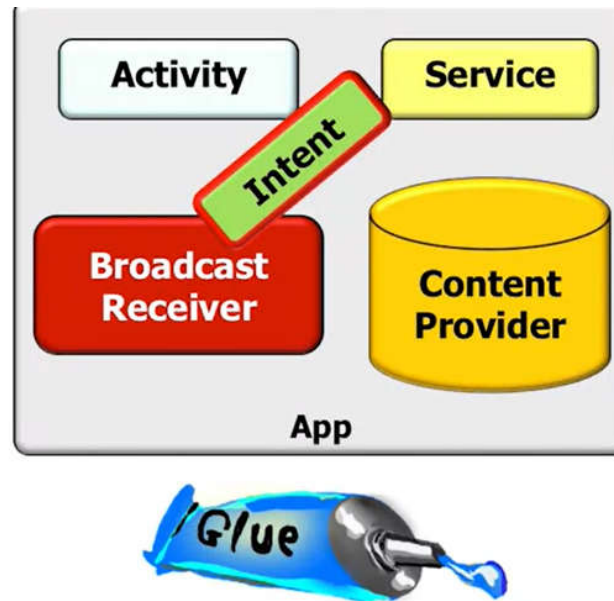
- Existem dois principais tipos de serviços:
  - **Iniciado:** Um serviço é "iniciado" quando um componente do aplicativo (como uma atividade) inicia-o chamando `startService()`.
  - Exemplo: fazer download ou upload de um arquivo pela rede. Quando a operação for concluída, o serviço deverá ser interrompido.
- **Vinculado:** Um serviço é "vinculado" quando um componente do aplicativo chama `bindService()` para vinculá-lo.
- Um serviço vinculado permanece em execução somente enquanto outro componente do aplicativo estiver vinculado a ele.

## Broadcast Receivers (cont.)

- Os Broadcast Receivers são componentes responsáveis por receber e tratar eventos (ou *broadcasts*) provenientes do sistema ou de outras aplicações.
- Por exemplo, estes são alguns broadcasts do sistema que podem ser tratados por qualquer aplicação:
  - `android.intent.action.ACTION_BOOT_COMPLETED`: indica que o telefone acabou de ser ligado
  - `android.intent.action.ACTION_POWER_CONNECTED`: indica que o carregador foi conectado
  - `android.intent.action.ACTION_BATTERY_LOW`: indica que o nível de bateria está baixo.

# Intent

- A Intent é um objeto de mensagem que pode ser usado para solicitar uma ação de outro componente de aplicativo.
- Embora os intents facilitem a comunicação entre componentes de diversos modos, há três casos de uso fundamentais.



## Intent (cont.)

- **Para iniciar uma atividade:**
- [startActivity\(\)](#): a intent descreve a atividade a iniciar e carrega todos os dados necessários.
- [startActivityForResult\(\)](#): sua atividade recebe o resultado como um objeto Intent separado no retorno de chamada de [onActivityResult\(\)](#) da atividade.

## Intent (cont.)

- **Para iniciar um serviço:**
- O Service é um componente que realiza operações em segundo plano sem interface do usuário.
- A Intent descreve o serviço a iniciar e carrega todos os dados necessários.
- [startService\(\)](#): iniciar um serviço para realizar uma operação que acontece uma vez (como baixar um arquivo) passando uma intent.

## Intent (cont.)

- **Para fornecer uma transmissão:**
- Você pode fornecer uma transmissão a outros aplicativos passando um Intent a [sendBroadcast\(\)](#), [sendOrderedBroadcast\(\)](#) ou [sendStickyBroadcast\(\)](#).

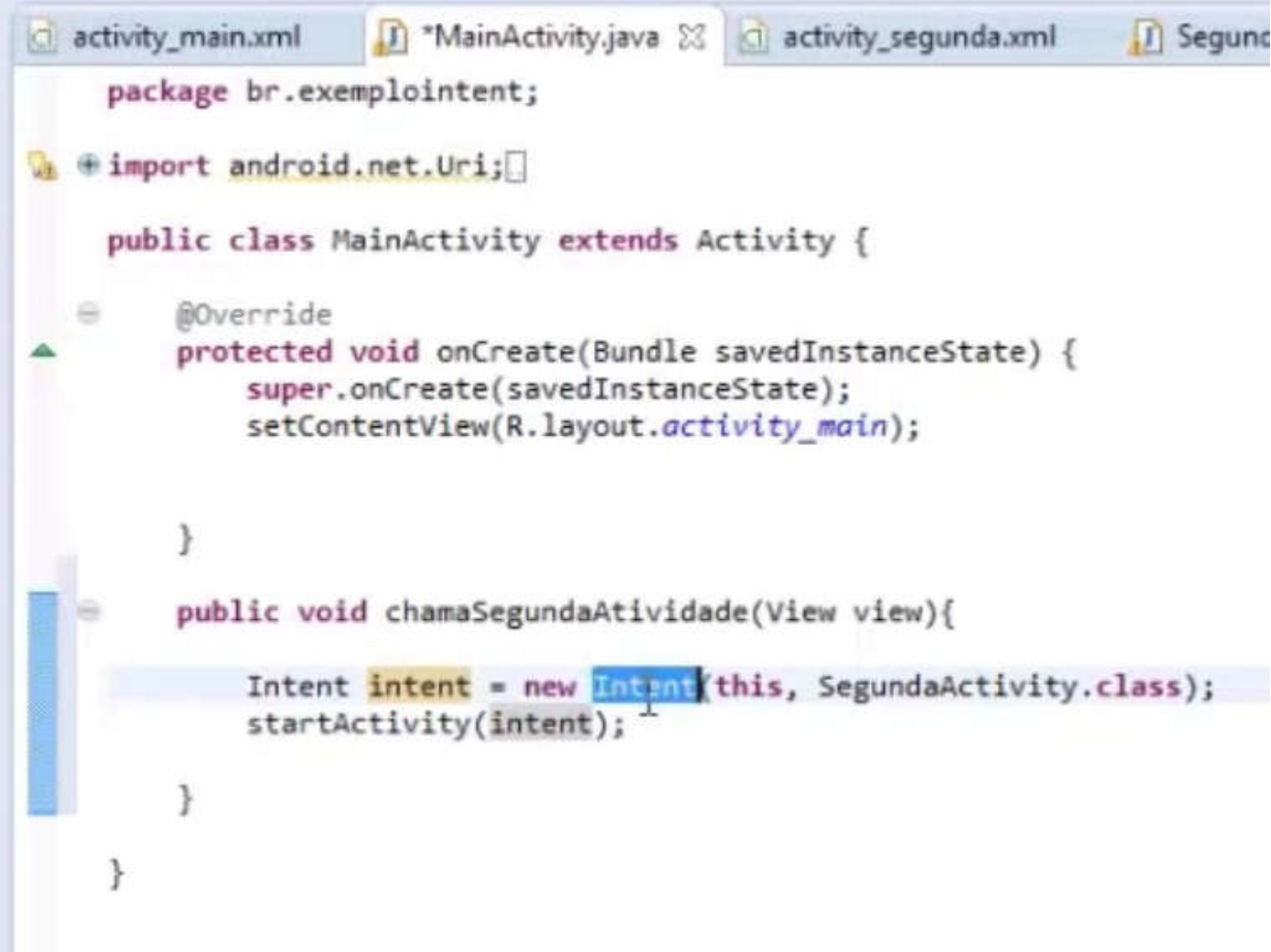


## Intent (cont.)

- Exemplo de Intent para chamar a 2ª activity:
- Criar um botão na activity principal
- No arquivo xml:

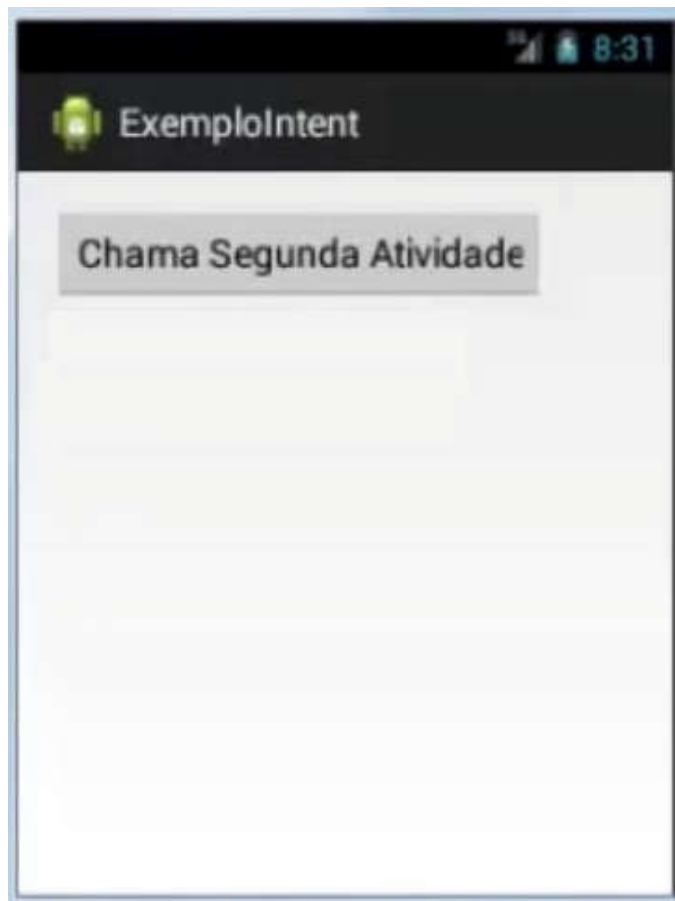
```
<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Chama Segunda Atividade"
    android:onClick="chamaSegundaAtividade" />
```

## Intent (cont.)



```
activity_main.xml  *MainActivity.java  activity_segunda.xml  Segund  
package br.exemplointent;  
  
import android.net.Uri;  
  
public class MainActivity extends Activity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
  
    public void chamaSegundaAtividade(View view){  
        Intent intent = new Intent(this, SegundaActivity.class);  
        startActivity(intent);  
    }  
}
```

## Intent (cont.)



## Relative Layout (cont.)

- Existem 3 formas de definir os parâmetros: `android:layout_width` e `android:layout_height`
  - **match\_parent:** o componente deve ocupar o tamanho definido pelo seu parente (pai). Deve ser utilizada sempre que o layout ocupar toda a tela;
  - **wrap\_content:** o componente deve ocupar apenas o seu tamanho, sem nenhum tipo de refatoração;
  - **Definição manual:** Essa é uma das formas utilizadas para definir um tamanho, mas deve-se tomar muito cuidado, pois as telas dos aparelhos móveis possuem vários tamanhos e isso pode tornar seu aplicativo incompatível com alguns tamanhos de tela.

## Relative Layout (cont.)



```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true" />
```

```
android:layout_below="@+id/textView4"
android:layout_alignParentLeft="true"
android:layout_alignParentStart="true"
android:layout_marginTop="74dp" />
```

```
android:layout_alignTop="@+id/textView5"
android:layout_toRightOf="@+id/textView5"
android:layout_toEndOf="@+id/textView5"
android:layout_marginLeft="138dp"
android:layout_marginStart="138dp" />
```

# Linear Layout

- O linear layout é um dos tipos de layouts mais utilizados no desenvolvimento com Android
- Possibilita organizar seus componentes de duas formas: alinhados horizontalmente ou alinhados verticalmente.
- O layout é criado vazio, portanto o desenvolvedor deve colocar os componentes.

## Linear Layout (cont.)



```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="Ola "
        android:id="@+id/textView" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="Tudo bem?"
        android:id="@+id/textView3" />

</LinearLayout>
```

# Table Layout

- O Table Layout é uma especialização do Linear Layout e é muito utilizado para criar formulários e telas de login.
- Primeiro, o TableLayout é definido como um componente interno.
- A seguir definimos um TableRow e dentro deste os elementos de tela desejados são colocados alinhados horizontalmente dentro da linha declarada.



# Table Layout

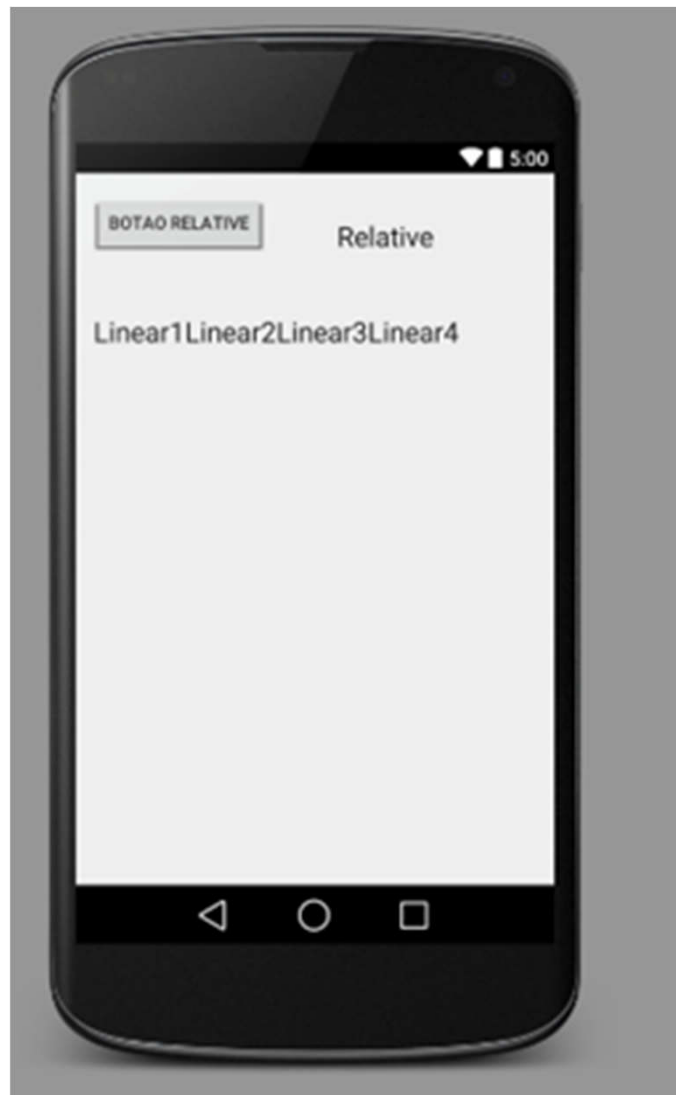
- Lembre-se que todos os elementos de tela devem estar definidos dentro de um TableRow se quiser que eles estejam alinhados na interface gráfica, já que tudo o que estiver dentro dele fará parte de uma mesma linha.



# Grid Layout

- Organiza os objetos em forma de grade

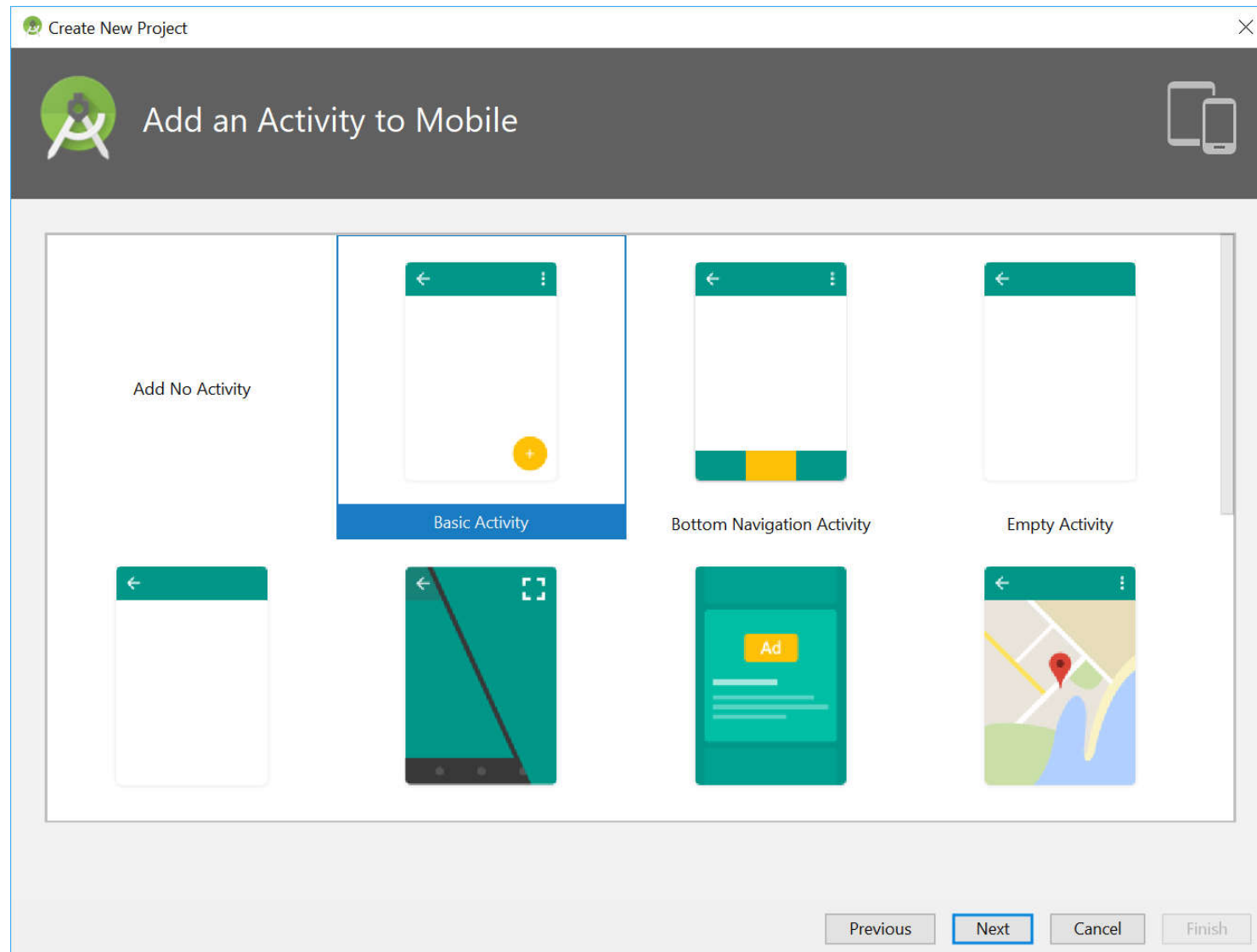
# Combinação de Layouts



# Scrow View

- Usado quando a sequencia de elementos é muito grande
- O scrow view só aceita um filho. Então o que for feito tem que ser feito dentro de um layout

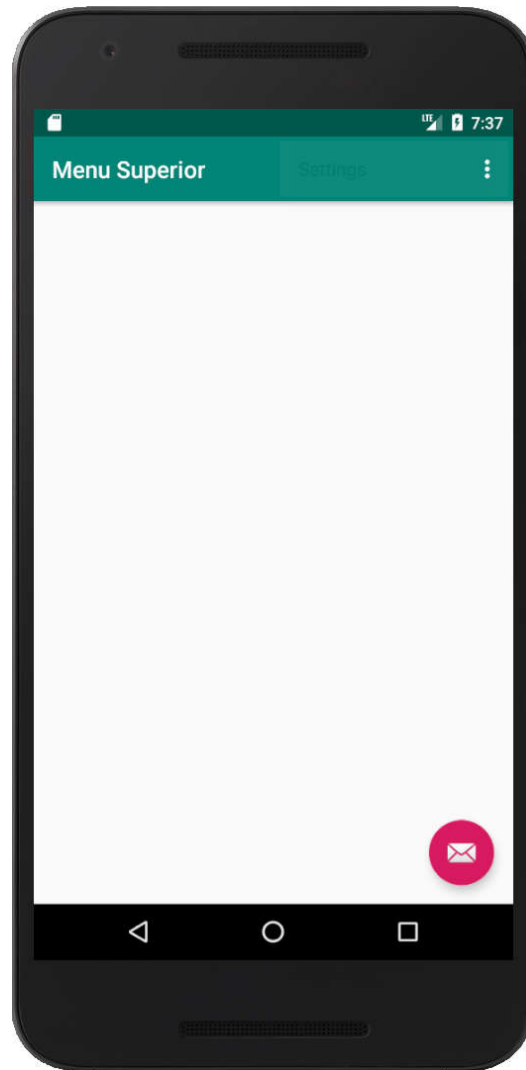
# Menu Superior



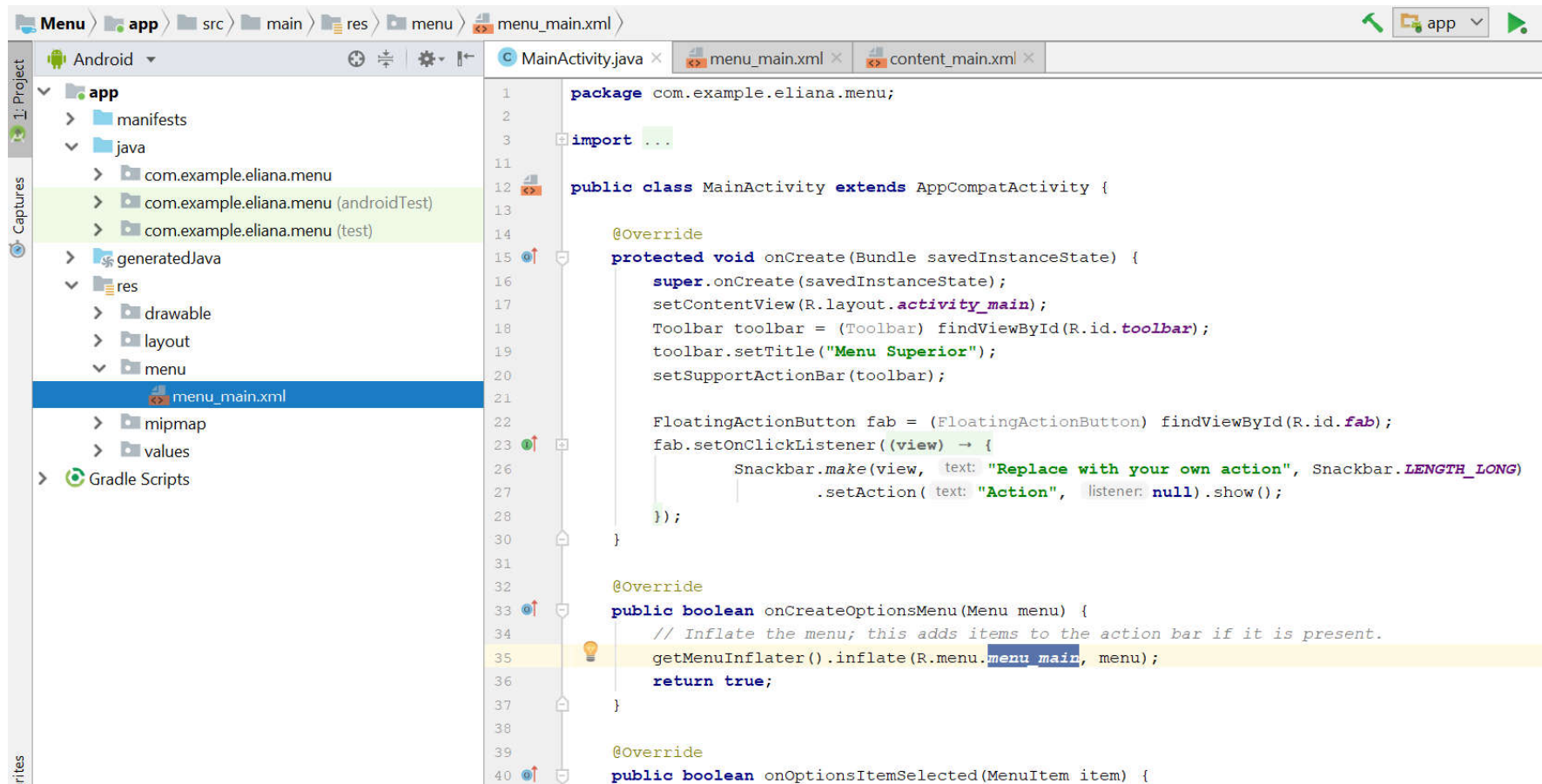
# Menu Superior (cont.)

```
1 package com.example.eliana.menu;
2
3 import ...
4
11
12 public class MainActivity extends AppCompatActivity {
13
14     @Override
15     protected void onCreate(Bundle savedInstanceState) {
16         super.onCreate(savedInstanceState);
17         setContentView(R.layout.activity_main);
18         Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
19         toolbar.setTitle("Menu Superior");
20         setSupportActionBar(toolbar);
21
22         FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);
23         fab.setOnClickListener((view) -> {
24             Snackbar.make(view, text: "Replace with your own action", Snackbar.LENGTH_LONG)
25                 .setAction(text: "Action", listener: null).show();
26         });
27     }
28
29
30
31
32     @Override
33     public boolean onCreateOptionsMenu(Menu menu) {
34         // Inflate the menu; this adds items to the action bar if it is present.
35         getMenuInflater().inflate(R.menu.menu_main, menu);
36         return true;
37     }
38
39
40     @Override
41     public boolean onOptionsItemSelected(MenuItem item) {
42         // Handle action bar item clicks here. The action bar will
```

## Menu Superior (cont.)



# Menu Superior (cont.)



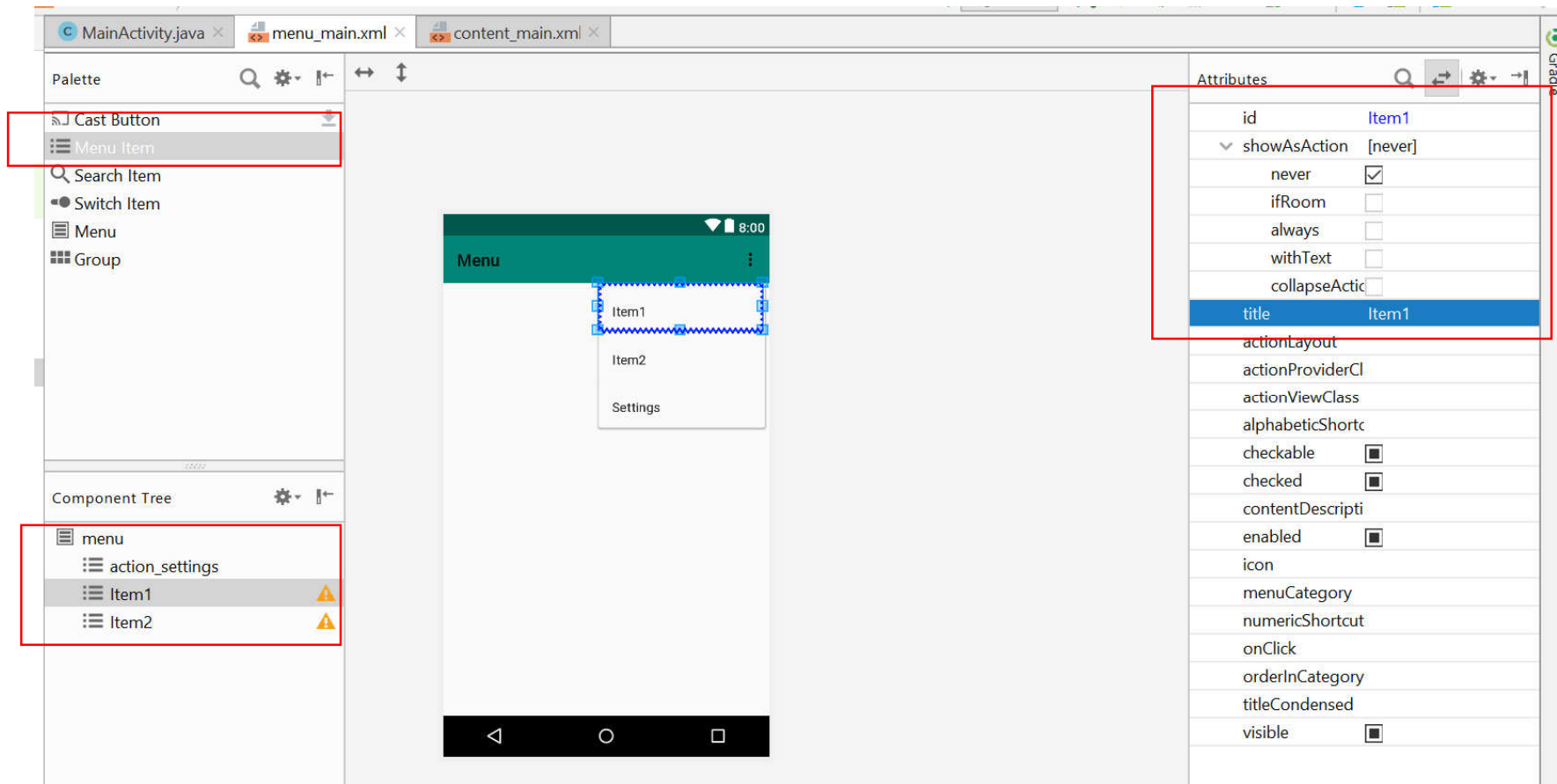


# Menu Superior (cont.)

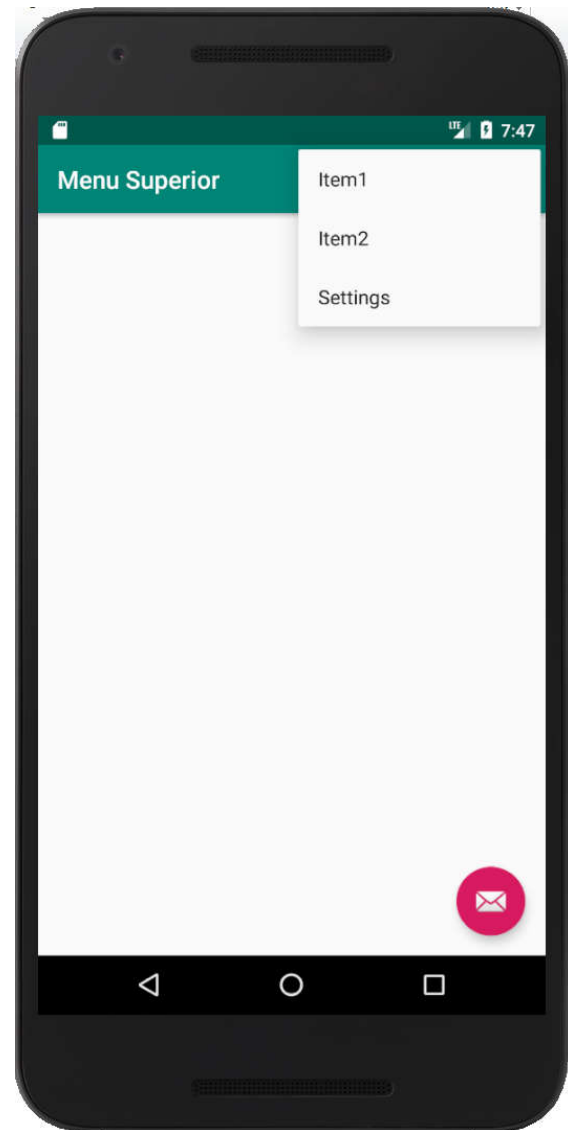
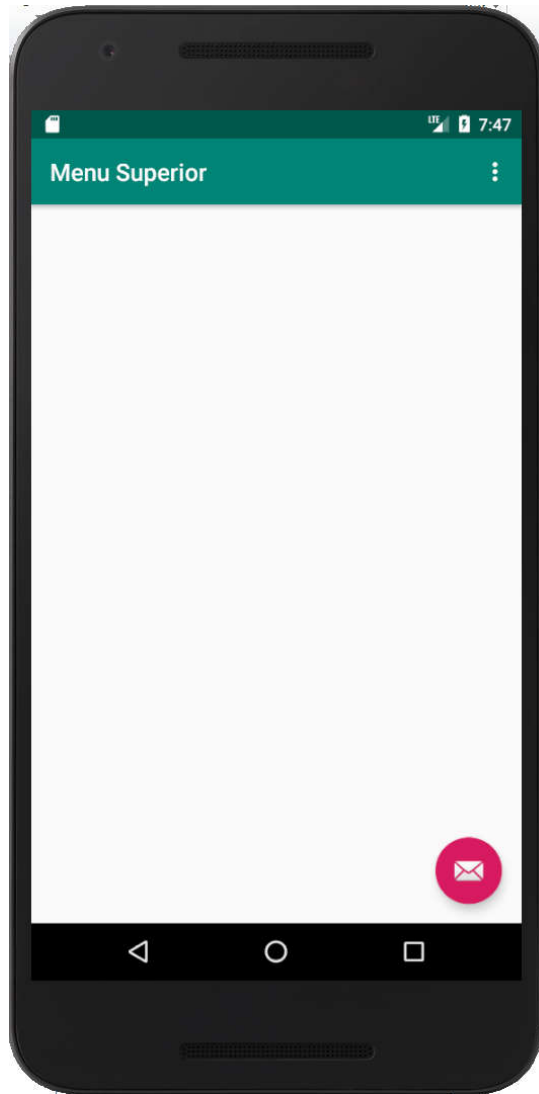
The screenshot displays the Android Studio interface with three tabs at the top: MainActivity.java, menu\_main.xml, and content\_main.xml. The main workspace shows a mobile app preview with a green header bar labeled 'Menu' and a 'Settings' item. The 'Palette' on the left lists UI components, with 'Menu Item' selected. The 'Component Tree' at the bottom shows the hierarchy: menu > action\_settings. The 'Attributes' panel on the right lists properties for the selected 'action\_settings' component. The 'title' attribute is highlighted with a red box and set to '@string/action\_settings'.

Attributes	
id	action_settings
orderInCategory	100
showAsAction	[never]
title	@string/action_settings
actionLayout	
actionProviderClass	
actionViewClass	
alphabeticShortcut	
checkable	<input type="checkbox"/>
checked	<input type="checkbox"/>
contentDescription	
enabled	<input type="checkbox"/>
icon	
menuCategory	
numericShortcut	
onClick	
titleCondensed	
visible	<input type="checkbox"/>

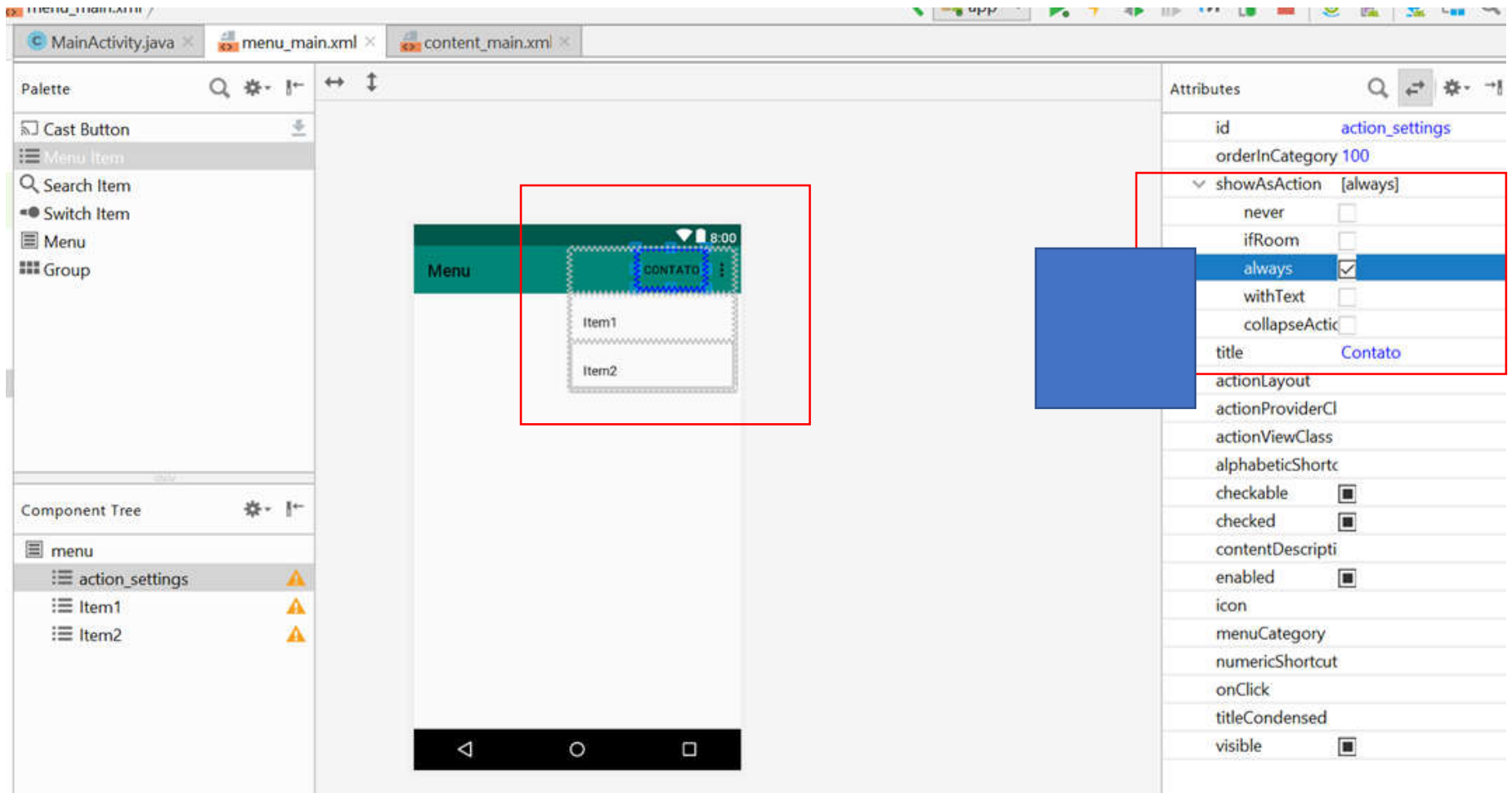
# Menu Superior (cont.)



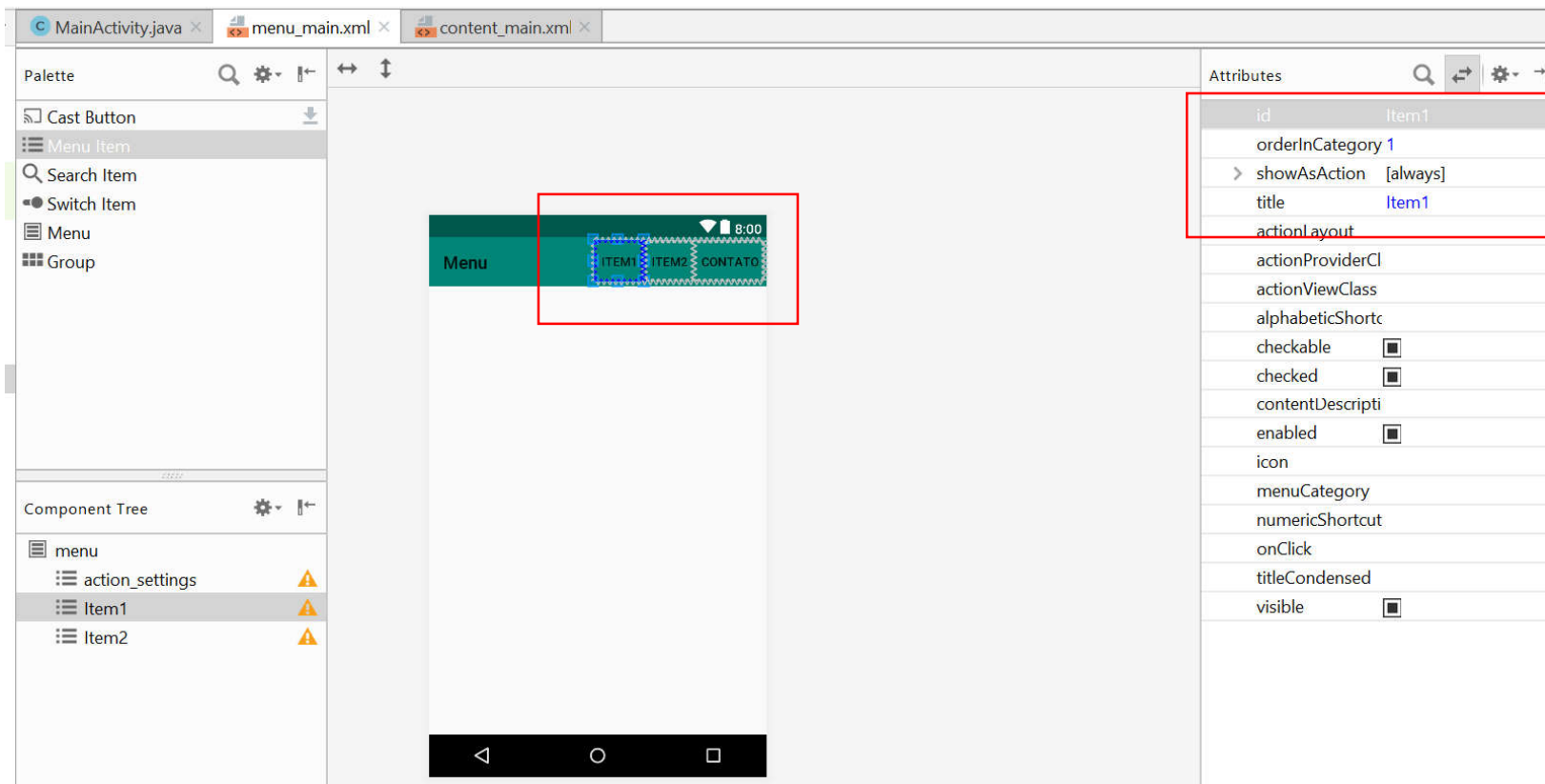
## Menu Superior (cont.)



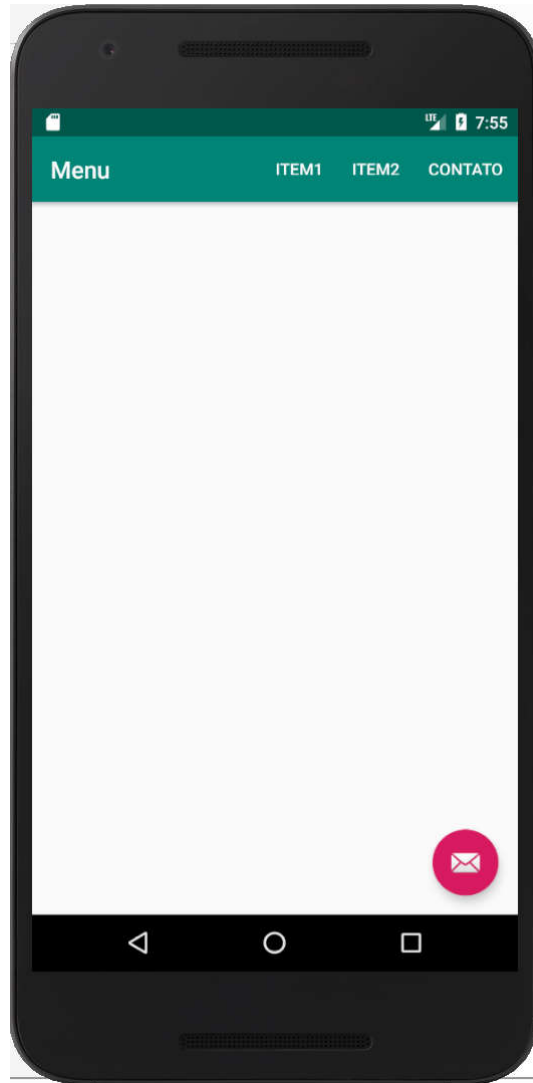
# Menu Superior (cont.)



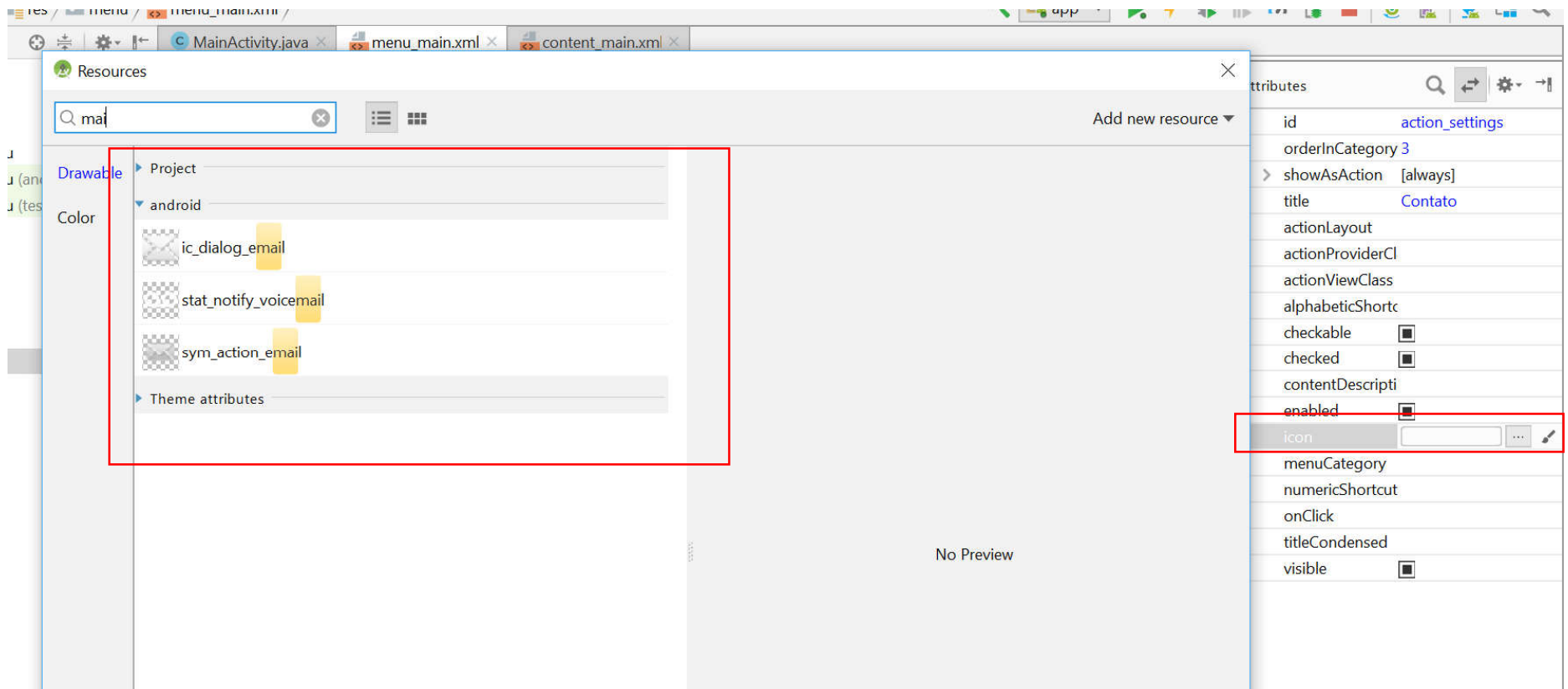
# Menu Superior (cont.)



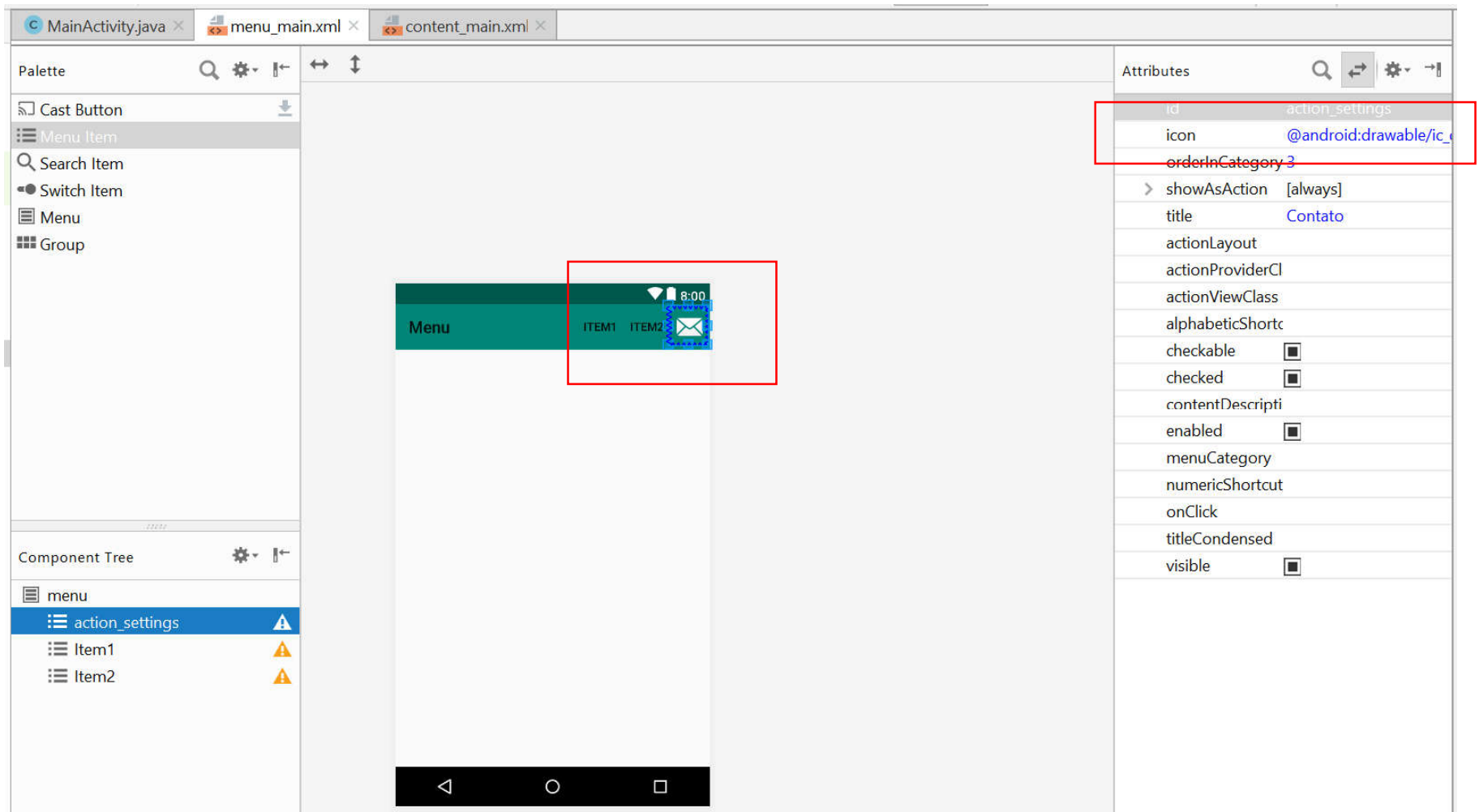
## Menu Superior (cont.)



# Menu Superior (cont.)

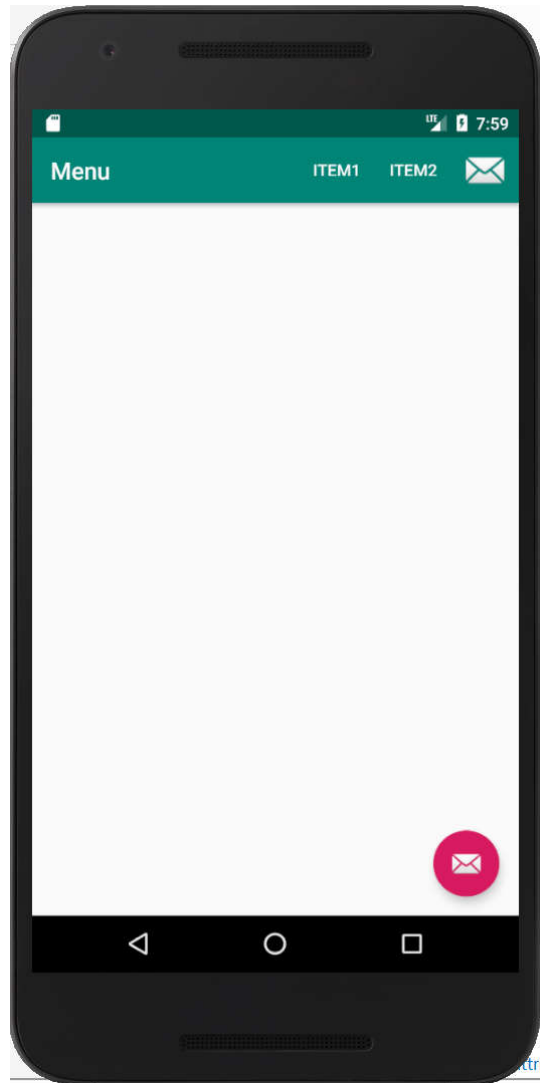


# Menu Superior (cont.)





## Menu Superior (cont.)



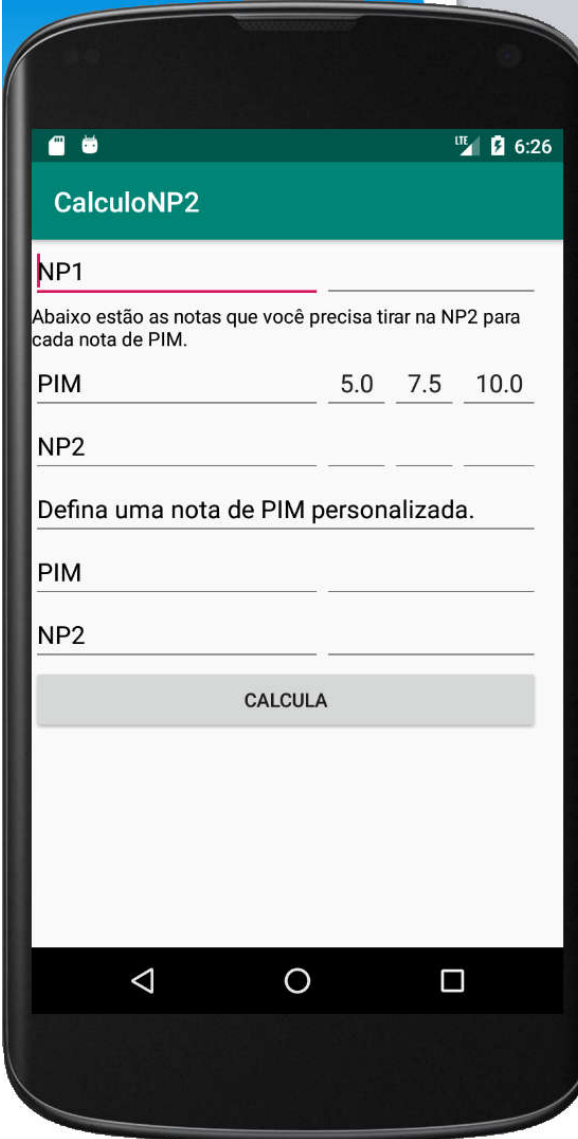
```
example / elliana / menu / MainActivity / app
MainActivity.java x menu_main.xml x content_main.xml x

13
14
15 @Override
16 protected void onCreate(Bundle savedInstanceState) {
17     super.onCreate(savedInstanceState);
18     setContentView(R.layout.activity_main);
19     Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
20     //toolbar.setTitle("Menu Superior");
21     setSupportActionBar(toolbar);
22
23     FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);
24     fab.setOnClickListener((view) -> {
25         Snackbar.make(view, text: "Replace with your own action", Snackbar.LENGTH_LONG)
26             .setAction(text: "Action", listener: null).show();
27     });
28 }
29
30
31
32 @Override
33 public boolean onCreateOptionsMenu(Menu menu) {
34     // Inflate the menu; this adds items to the action bar if it is present.
35     getMenuInflater().inflate(R.menu.menu_main, menu);
36     return true;
37 }
38
39
40 @Override
41 public boolean onOptionsItemSelected(MenuItem item) {
42     // Handle action bar item clicks here. The action bar will
43     // automatically handle clicks on the Home/Up button, so long
44     // as you specify a parent activity in AndroidManifest.xml.
45     int id = item.getItemId();
46
47     //noinspection SimplifiableIfStatement
48     if (id == R.id.action_settings) {
49         return true;
50     }
51
52     return super.onOptionsItemSelected(item);
53 }
```

# Exercício

- **Leiaute de uma calculadora de notas**
- 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.

## Exercício (cont.)



The image shows a smartphone screen with an Android interface. At the top, the status bar shows LTE, signal strength, battery, and the time 6:26. Below the status bar is a green header with the text "CalculoNP2". The main content area is white and contains the following elements:

- A text input field labeled "NP1" with a red underline.
- A paragraph of text: "Abaixo estão as notas que você precisa tirar na NP2 para cada nota de PIM."
- A table with two rows and four columns. The first row is labeled "PIM" and contains the values "5.0", "7.5", and "10.0". The second row is labeled "NP2" and contains empty input fields.
- A text input field labeled "Defina uma nota de PIM personalizada."
- A text input field labeled "PIM".
- A text input field labeled "NP2".
- A gray button labeled "CALCULA".

The bottom of the screen shows the standard Android navigation bar with back, home, and recent apps buttons.