Ficha do Aplicativo

App07 – Recebe Mensagens
Resumo Este é um aplicativo muito simples. Ele foi criado para ser um receptor de Intents Implícitos que depois de instalado em um dispositivo poderá receber intents action.SEND, com dados do tipo text/plain. Deste modo, esse aplicativo poderá responder a Intents iniciados pelo App06 desenvolvido neste curso.
Objetivos de Aprendizagem 1. Editar o arquivo de manifesto do aplicativo para inserir um filtro de intent 2. Tratar o intent no método onCreate() da classe MainActivity
Dinâmica do Aplicativo Não há interação do usuário com este aplicativo, que serve para demonstrar o uso do filtro de intents. Este aplicativo será aberto pelo sistema operacional Android quando for escolhido pelo usuário para receber um intent compatível com a ação action.SEND passando dado no formato text/plain

Lista de Activities do Aplicativo

Nome	Layout
MainActivity.java	activity_main.xml

Resources

strings.xml

{ este arquivo não foi alterado }

colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
    <color name="corFundo">#CB76F9</color>
    <color name="corFundoTexto">#F2E3FC</color>
</resources>
```

AndroidManifest.xml

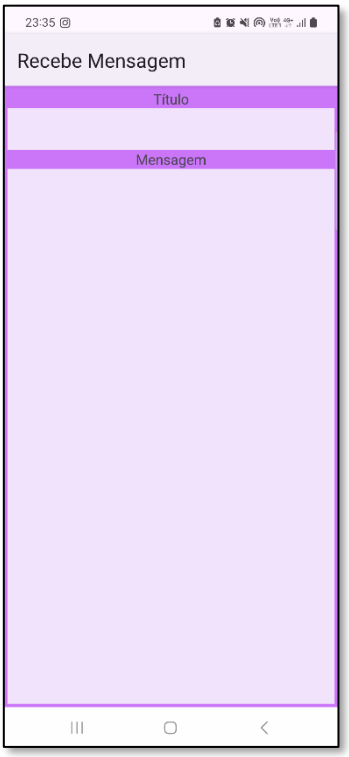
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.fatecsp_ilp506.recebemensagem">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

            <intent-filter>
                <category android:name="android.intent.category.DEFAULT" />
                <action android:name="android.intent.action.SEND" />
                <data android:mimeType="text/plain" />
            </intent-filter>

        </activity>
    </application>

</manifest>
```

Estas linhas devem ser acrescentadas ao manifesto do aplicativo. Elas configuram o filtro de intent, dotando este aplicativo da capacidade de responder a mensagens do tipo action.SEND com dados enviados no formato text/plain

Detalhamento da Activity

Layout: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@color/corFundo"
        android:orientation="vertical"
        android:padding="4dp"
        tools:layout_editor_absoluteX="121dp"
        tools:layout_editor_absoluteY="230dp">

        <TextView
            android:id="@+id/textView1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="@color/corFundo"
            android:gravity="center"
            android:text="Título"
            android:textSize="16sp" />

        <TextView
            android:id="@+id/txtTitulo"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="@color/corFundoTexto"
            android:gravity="center"
            android:text="TextView"
            android:textSize="18sp" />

        <TextView
            android:id="@+id/textView2"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="@color/corFundo"
            android:gravity="center"
            android:text="Mensagem"
            android:textSize="16sp" />

        <TextView
            android:id="@+id/txtMensagem"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:background="@color/corFundoTexto"
            android:text="TextView"
            android:textSize="18sp" />
    </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```



Código: MainActivity.java

```
/* Observação:
   O nome do package está omitido, porque ao fazer o seu projeto esse nome será outro.
   Os imports também estão omitidos. O motivo para isso é que a cada nova versão do Android Studio pode
   haver mudanças nos nomes dos caminhos da biblioteca. */

public class MainActivity extends AppCompatActivity {

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

        Intent intent = getIntent();
        String msg = intent.getStringExtra(Intent.EXTRA_TEXT);
        String titulo = intent.getStringExtra(Intent.EXTRA_SUBJECT);
        TextView txt;
        txt = findViewById(R.id.txtMensagem);
        txt.setText(msg);
        txt = findViewById(R.id.txtTitulo);
        txt.setText(titulo);
    }
```

Notas Técnicas

Arquivo de manifesto – AndroidManifest.xml

O arquivo de manifesto permite ao desenvolvedor configurar uma quantidade muito grande de parâmetros de projeto do aplicativo para Android. Existem parâmetros se aplicam ao projeto como um todo e existem parâmetros que se aplicam a Actvities específicas.

Neste projeto precisaremos aplicar um filtro de Intent para que o sistema operacional Android seja capaz de identificar nosso aplicativo como capaz de responder ações identificadas como action.SEND. Porém isso só não basta. É preciso complementar a especificação do filtro com outros dois parâmetros: a categoria deve ser category.DEFAULT para que o Android saiba que esta Activity é elegível para receber intents implícitas e a como a ação é do tipo SEND, significando que algum dado terá sido enviado, precisamos também informar o tipo de dado. Neste aplicativo vamos receber texto não formatado, então o mimeType deve ser especificado como "text/plain".

Assim, o filtro a ser acrescentado ao manifesto deste aplicativo fica da forma mostrada a seguir

```
<intent-filter>
    <category android:name="android.intent.category.DEFAULT" />
    <action android:name="android.intent.action.SEND" />
    <data android:mimeType="text/plain" />
</intent-filter>
```

Para saber mais sobre Intents, seus usos e filtros, acesse: https://developer.android.com/guide/components/intents-filters

Como tratar a Intent e os dados recebidos

Ao fazer a alteração no arquivo AndroidManifes.xml mostrada acima este aplicativo já estará listado como elegível para receber Intents implícitas. Agora é necessário que o programador escreva o código que irá capturar os dados enviados e os manipule de modo adequado, incorporando-os ao aplicativo.

Isto é feito no método onCreate() cujo código é reproduzido a seguir.

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

    Intent intent = getIntent();
    String msg = intent.getStringExtra(Intent.EXTRA_TEXT);
    String titulo = intent.getStringExtra(Intent.EXTRA_SUBJECT);
    TextView txt;
    txt = findViewById(R.id.txtMensagem);
    txt.setText(msg);
    txt = findViewById(R.id.txtTitulo);
    txt.setText(titulo);
}
```

No código acima a função getIntent() tem papel chave. É ela a responsável por instanciar o objeto da classe Intent e carregá-lo com os dados despachado pela Activity de origem (chamadora).

Na sequência usamos o método Intent.getStringExtra() para recuperar cada um dos dados enviados, carregando-os em objetos String e, por fim, carregamos esses Strings nos TextViews que compõe o layout da Activity.

Tarefa associada a esta aula

Altere o aplicativo **Brincando com Inteiros** feito em aula anterior de modo que ele também seja capaz de responder a intents implícitas caracterizadas por action.SEND e tipo text/plain.

Ele deve receber o texto da mensagem usando a constante EXTRA_TEXT e, caso esse string seja formado exclusivamente por algarismos, deve processar e preencher todos os TextViews do aplicativo.

Para implementar a solução desta tarefa devem ser feitas duas coisas:

- Alterar o AndroidManifest.xml fazendo a inclusão do filtro de Intent para que este aplicativo responda a ação action.SEND com tipo de dado text/plain

```
<intent-filter>
    <category android:name="android.intent.category.DEFAULT" />
    <action android:name="android.intent.action.SEND" />
    <data android:mimeType="text/plain" />
</intent-filter>
```

- No método onCreate() da classe MainActivity deve-se tratar o recebimento da intent. Para isso criei o código do método trataIntent() motrado a seguir, que é chamado em onCreate().

  Na implementação de trataIntent() usamos o método getIntent() para construir um objeto Intent através do qual poderemos recuperar os dados que forem oriundos da chamada de origem. Usando o método Intent.getStringExtra() fazemos essa recuperação de dados e usamos a informação passada do modo que for apropriado.

  Como as caixas de texto – txtGerado e edtTransfere – devem ser carregadas exclusivamente com algarismos (devido à natureza do aplicativo Brincando com Números) fazemos a conversão do string recebido para número, se a conversão for bem sucedida carregamos as caixas de texto e disparamos a execução dos métodos que calculam os resultados. Caso a conversão falhe gerando uma exceção, capturamos a exceção no comando catch, e neste caso, nada fazemos. Assim, se a mensagem recebida através do Intent não contiver um número válido o aplicativo inicia com os componentes vazios.

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

    trataIntent();
}

private void trataIntent() {
    Intent intent = getIntent();
    String msg = intent.getStringExtra(Intent.EXTRA_TEXT);
    try {
        int N = Integer.parseInt(msg);
        TextView txt;
        txt = findViewById(R.id.txtGerado);
        txt.setText(msg);
        txt = findViewById(R.id.edtTransfere);
        txt.setText(msg);
        btnParidadeOnClick(null);
        btnPrimoOnClick(null);
        txtDivisoresOnClick(null);
    }
    catch (Exception e) {

    }
}
```