Ficha do Aplicativo

App06 – O Sistema de Intents
Resumo Este é o primeiro aplicativo no nosso curso com mais de uma Activity. Nele o usuário irá digitar em uma caixa de texto (EditText) e poderá enviar esse texto da MainActivity para uma outra atividade do próprio aplicativo, para outros aplicativos, tais como app de EMail, WhatsApp, Instagram, Google Docs ou qualquer outro que esteja instalado no dispositivo e apto a receber mensagens de texto.
Objetivos de Aprendizagem 1. Uso de um Drawable de Formato para definir o desenho de componentes visuais (no caso, dos botões) 2. Criação de uma segunda activity no mesmo aplicativo. 3. Compreensão e uso de intent, tanto explícitos como implícitos. 4. Envio e recepção de dados através de intents. 5. Integração do seu aplicativo com aplicativos desenvolvidos por outras empresas. 6. Voltar a usar tratamento de exceções (tratamento de erros no app), agora com tipo específico de exceção. 7. Usar novamente o recurso Toast para exibir mensagens que desaparecem após um tempo.
Dinâmica do Aplicativo O usuário digitará um texto em um componente EditText e poderá clicar em um de cinco botões disponíveis. O primeiro botão dispara um intent explícito para a segunda activity do aplicativo. O segundo botão dispara um intent implícito que será tratado pelo sistema operacional e direcionada ao aplicativo capaz de responder à ação a ser realizada – se houver mais de uma, o Android apresentará uma tela de escolha, caso não haja um aplicativo default para o tipo de intent gerado. O terceiro botão dispara um intent implícito que propositalmente inibe o uso de aplicativo padrão, ou seja, o usuário sempre terá a opção de escolher o aplicativo que será iniciado e receberá o intent. O quarto botão direciona o texto digitado, através de intent específico para o WhatsApp, caso ele esteja instalado no dispositivo. Caso não esteja, será levantada uma exceção para tratamento de erro que exibirá uma mensagem ao usuário através de um Toast. O quinto botão direciona o texto digitado para o aplicativo Google Docs com o objetivo de salvar um arquivo no Google Drive.

Lista de Activities do Aplicativo

Nome	Layout
MainActivity.java	activity_main.xml
RecebeIntent.java	activity_recebe_intent.xml

Resources

strings.xml

```
<resources>
    <string name="app_name">O Sistema de Intents</string>
    <string name="tit_activity_envia_msg">Envia Mensagem</string>
    <string name="tit_activity_recebe_msg">Recebe Mensagem</string>
    <string name="btniniciaintent">Inicia Intent</string>
    <string name="btnenviageral">Envia Geral</string>
    <string name="btnsempreescolhe">Sempre Escolhe</string>
    <string name="btnenviawhatsapp">Envia para o WhatsApp</string>
    <string name="btnenviagoogledocs">Envia para o Google Docs</string>
    <string name="lblcxescolha">Escolha o Aplicativo</string>
    <string name="errowhatsapp">WhatsApp não Instalado</string>
    <string name="errogoogledocs">Conexão com o Google Docs falhou</string>
</resources>
```

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="corTelaFundo">#FFFFE5AB</color>
    <color name="corTextoFundo">#FFFAF0C8</color>
    <color name="corBotaoFundo">#FFA0A0FF</color>
    <color name="corBotaoBorda">#FF104DF7</color>
    <color name="corRcbTelaFundo">#FFC4C9F7</color>
    <color name="corRcbTextoFundo">#FFF8F6E9</color>
</resources>
```

fundoedittext.xml (drawable tipo "shape" – veja as notas técnicas)

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <solid android:color="@color/corTextoFundo" />
    <stroke
        android:width="1dp"
        android:color="@color/corBotaoBorda"
        />
    <corners android:radius="4dp" />
</shape>
```

fundobotao.xml (drawable tipo "ripple" – veja as notas técnicas)

```
<?xml version="1.0" encoding="utf-8"?>
<ripple xmlns:android="http://schemas.android.com/apk/res/android"
    android:color="@color/white">

    <item>
        <shape android:shape="rectangle">
            <solid android:color="@color/corBotaoFundo" />
            <stroke
                android:color="@color/corBotaoBorda"
                android:width="4dp"/>
            <corners android:radius="30dp" />

        </shape>
    </item>

</ripple>
```

Detalhamento da Activity

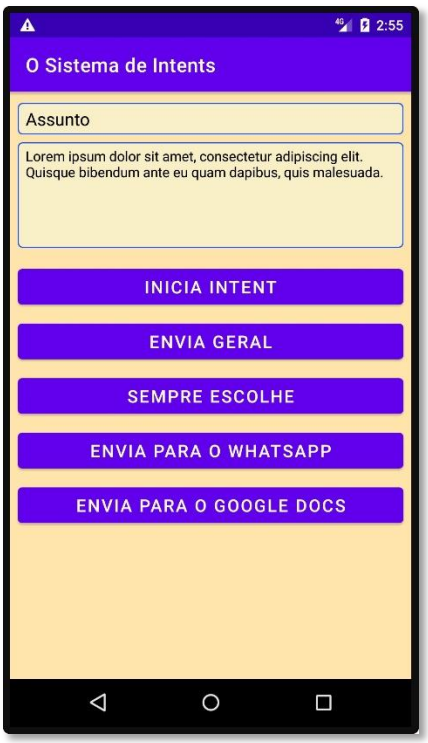
Layout: activity_main.xml

```
<?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"
    android:background="@color/corTelaFundo"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/edtTitulo"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginEnd="8dp"
        android:background="@drawable/fundoedittext"
        android:ems="10"
        android:hint="assunto"
        android:inputType="textPersonName"
        android:paddingStart="8dp"
        android:paddingTop="4dp"
        android:paddingEnd="8dp"
        android:paddingBottom="4dp"
        android:text="Assunto"
        android:textColor="@color/black"
        android:textColorLink="@color/purple_700"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.01999998" />

    <EditText
        android:id="@+id/edtMensagem"
        android:layout_width="0dp"
        android:layout_height="108dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:background="@drawable/fundoedittext"
        android:gravity="top|start"
        android:hint="texto"
        android:inputType="textMultiLine"
        android:lines="5"
        android:minLines="3"
        android:paddingStart="8dp"
        android:paddingTop="4dp"
        android:paddingEnd="8dp"
        android:paddingBottom="4dp"
        android:scrollbars="vertical"
        android:text="Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque bibendum ante eu quam dapibus,
quis malesuada."
        android:textColor="@color/black"
        android:textColorLink="@color/purple_700"
        android:textSize="14sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/edtTitulo" />

    <Button
        android:id="@+id/btnIniciaIntent"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="8dp"
        android:onClick="btnIniciaIntentOnClick"
        android:text="@string/btniniciaintent"
        android:textAllCaps="true"
        android:textSize="18sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.502"
```



```
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/edtMensagem" />

    <Button
        android:id="@+id/btnEnviaGeral"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:onClick="btnEnviaGeralOnClick"
        android:text="@string/btnenviageral"
        android:textAllCaps="true"
        android:textSize="18sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnIniciaIntent" />

    <Button
        android:id="@+id/btnSempreEscolhe"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:onClick="btnSempreEscolheOnClick"
        android:text="@string/btnsempreescolhe"
        android:textAllCaps="true"
        android:textSize="18sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnEnviaGeral" />

    <Button
        android:id="@+id/btnEnviaWhatsApp"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:onClick="btnEnviaWhatsAppOnClick"
        android:text="@string/btnenviawhatsapp"
        android:textAllCaps="true"
        android:textSize="18sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnSempreEscolhe" />

    <Button
        android:id="@+id/btnEnviaGoogleDocs"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:onClick="btnEnviaGoogleDocsOnClick"
        android:text="@string/btnenviagoogledocs"
        android:textAllCaps="true"
        android:textSize="18sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnEnviaWhatsApp" />

</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 {

    private EditText edtTitulo;
    private EditText edtMensagem;
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        edtTitulo = findViewById(R.id.edtTitulo);
        edtMensagem = findViewById(R.id.edtMensagem);
    }

    public void btnIniciaIntentOnClick(View view) {
        Intent intent = new Intent(this, RecebeIntent.class);
        intent.putExtra(RecebeIntent.EXTRA_MSG, edtMensagem.getText().toString());
        intent.putExtra(RecebeIntent.EXTRA_ASSUNTO, edtTitulo.getText().toString());
        startActivity(intent);
    }

    public void btnEnviaGeralOnClick(View view) {
        Intent intent = new Intent(Intent.ACTION_SEND);
        intent.setType("text/plain");
        intent.putExtra(Intent.EXTRA_TEXT, edtMensagem.getText().toString());
        intent.putExtra(Intent.EXTRA_SUBJECT, edtTitulo.getText().toString());
        if (intent.resolveActivity(getPackageManager()) != null)
            startActivity(intent);
    }

    public void btnSempreEscolheOnClick(View view) {
        Intent intent = new Intent(Intent.ACTION_SEND);
        intent.setType("text/plain");
        intent.putExtra(Intent.EXTRA_TEXT, edtMensagem.getText().toString());
        intent.putExtra(Intent.EXTRA_SUBJECT, edtTitulo.getText().toString());
        String tituloEscolha = getString(R.string.lblcxescolha);
        Intent intentEscolhido = Intent.createChooser(intent, tituloEscolha);
        if (intent.resolveActivity(getPackageManager()) != null)
            startActivity(intentEscolhido);
    }

    public void btnEnviaWhatsAppOnClick(View view) {
        PackageManager pm = getPackageManager();
        try {
            Intent intent = new Intent(Intent.ACTION_SEND);
            intent.setType("text/plain");
            intent.putExtra(Intent.EXTRA_TEXT, edtMensagem.getText().toString());
            intent.putExtra(Intent.EXTRA_SUBJECT, edtTitulo.getText().toString());
            // Verifica se o pacote existe (app instalado). Se não estiver desvia para o bloco catch
            PackageInfo info = pm.getPackageInfo("com.whatsapp", PackageManager.GET_META_DATA);
            intent.setPackage("com.whatsapp");
            startActivity(intent);
        }
        catch (PackageManager.NameNotFoundException e) {
            Toast.makeText(this, R.string.errowhatsapp, Toast.LENGTH_SHORT).show();
        }
    }

    public void btnEnviaGoogleDocsOnClick(View view) {
        PackageManager pm = getPackageManager();
        try {
            Intent intent = new Intent(Intent.ACTION_SEND);
            intent.setType("text/plain");
            intent.putExtra(Intent.EXTRA_TEXT, edtMensagem.getText().toString());
            intent.putExtra(Intent.EXTRA_SUBJECT, edtTitulo.getText().toString() + ".txt");
            // Verifica se o pacote existe (app instalado). Se não estiver desvia para o bloco catch
            PackageInfo info = pm.getPackageInfo("com.google.android.apps.docs", PackageManager.GET_META_DATA);
            intent.setPackage("com.google.android.apps.docs");
            startActivity(intent);
        }
        catch (PackageManager.NameNotFoundException e) {
            Toast.makeText(this, R.string.errogoogledocs, Toast.LENGTH_SHORT).show();
        }
    }

}
```

Layout: activity_recebe_intent.xml

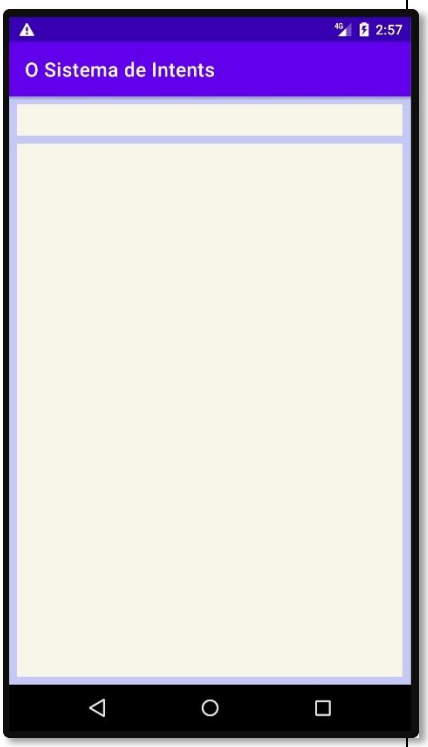
```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"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

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@color/corRcbTelaFundo"
        android:orientation="vertical"
        tools:layout_editor_absoluteX="86dp"
        tools:layout_editor_absoluteY="107dp">

        <TextView
            android:id="@+id/txtTitulo"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginStart="8dp"
            android:layout_marginTop="8dp"
            android:layout_marginEnd="8dp"
            android:background="@color/corRcbTextoFundo"
            android:gravity="center"
            android:textColor="@color/black"
            android:textSize="24sp" />

        <TextView
            android:id="@+id/txtMensagem"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_marginStart="8dp"
            android:layout_marginTop="8dp"
            android:layout_marginEnd="8dp"
            android:layout_marginBottom="8dp"
            android:background="@color/corRcbTextoFundo"
            android:paddingStart="4dp"
            android:paddingEnd="4dp"
            android:textColor="@color/black"
            android:textSize="18sp" />
    </LinearLayout>


</androidx.constraintlayout.widget.ConstraintLayout>
```



Código: RecebeIntentActivity.java

```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 RecebeIntent extends AppCompatActivity {

    public static final String EXTRA_MSG = "msg";
    public static final String EXTRA_ASSUNTO = "assunto";

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

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

Notas Técnicas

Drawable Resource – Conceito

Como vimos no App04 deste curso, no ambiente Android, conceitualmente, Drawables são recursos (*resources*). Este termo se refere de maneira geral a elementos gráficos possíveis de serem desenhados na tela dos dispositivos. No código Java, drawables podem ser acessados através da função getDrawable() ou em arquivos XML podem ser atribuídos a outros recursos através de atributos como android:drawable, android:background ou android:icon. Existe uma grande variedade de drawables e a lista completa pode ser consultada no link ao final desta seção

Vimos que a primeira ideia que vem à mente quando se fala em algo que possa ser desenhado são as imagens de bitmaps dos arquivos PNG ou JPG. Mas, também vimos que existem tipos de drawables que são arquivos texto no formato XML. Nestes casos o Android interpreta o conteúdo do XML e executa o desenho. Pode-se pensar nesses XML como uma receita passo a passo, com atributos devidamente configurados para que o Android seja capaz interpretá-los e executar o desenho.

Neste aplicativo serão definidos dois arquivos: fundoesittext.xml e fundobotao.xml. Cada um destes arquivos define um Drawable, porém de tipos diferentes: enquanto fundoesittext.xml define um Formato (*shape*), o fundobotao.xml define um Efeito Visual (*ripple*). Nos links ao final desta seção podem ser obtidos mais detalhes.

O resource ***shape*** permite definir um elemento com uma forma geométrica dentre quatro opções: retângulo, elipse, linha ou anel. Além disso é possível definir cores para o fundo e borda, estilo de borda, preenchimento gradiente, arredondamento de cantos, entre outros possíveis atributos. A documentação do Android descreve por completo todas as possibilidades, bem como deixa claro quais são aplicáveis em cada caso (p.ex. arredondamento de cantos se aplica somente a retângulos, não tendo qualquer efeito nas outras formas).

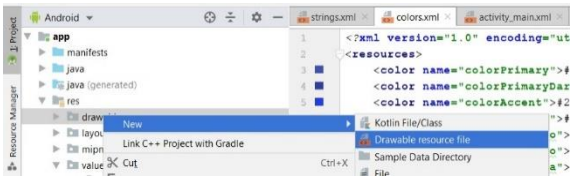
O resource ***ripple*** permite criar um efeito visual que é disparado quando, em um componente visual do layout, ocorre uma transição de estado devido à ação do usuário, como pro exemplo um toque em um botão.

Para saber mais sobre:

Drawable resources: https://developer.android.com/guide/topics/resources/drawable-resource

Ripple resources: https://developer.android.com/reference/android/graphics/drawable/RippleDrawable

Como criar e aplicar o Drawable deste aplicativo



No painel esquerdo, navegue pela estrutura de pastas do seu aplicativo, localize a pasta **app → res → drawable** e clique sobre a mesma com o botão direito do mouse.

No menu que irá abrir aponte o menu **new → Drawable resource file** e clique. Será aberta uma janela onde fornecemos o nome do resource – que será fundoesittext (digite sem a extensão xml) e no campo *root element* deve ser especificado o tipo *shape* (para o drawable fundobotao especifique *ripple*).

É regra do Android que nomes de drawables só podem conter letras minúsculas, algarismos e o caractere underscore ' _ '.

Pronto, ao executar esses passos o resource estará criado, porém vazio.

Agora devemos escrever dentro dele as tags e atributos XML necessários, conforme o tipo de drawable que se está criando. Para saber o que colocar neste arquivo é necessário que o iniciante consulte a documentação indicada acima. No caso deste aplicativo basta copiar o texto XML desse resource, que está na página 2, e colar no Android Studio.

A segunda parte é usar o drawable. Para isso, vá ao layout activity_main.xml e aplique-o ao atributo background dos botões. Você verá que imediatamente a aparência padrão do botão é substituída por uma nova aparência comandada pelo drawable.

Então, mãos à obra, crie os dois drawables do app.

Para que o ripple funcione corretamente ao ser aplicado aos botões é preciso trocar o componente usado no XML do layout. Troque Button por android.widget.Button.

A partir do Android Studio 4.1.xx o componente Button refere-se ao botão padrão do Material Design, que tem suas próprias regras de desenho e para obter no botão do aplicativo os efeitos produzidos pelo ripple e pelo shape é preciso usar o componente Button convencional

Troque isto...

```
<Button
    android:id="@+id/btnEnviaGeral"
    android:layout_width="0dp"
```

...por isto

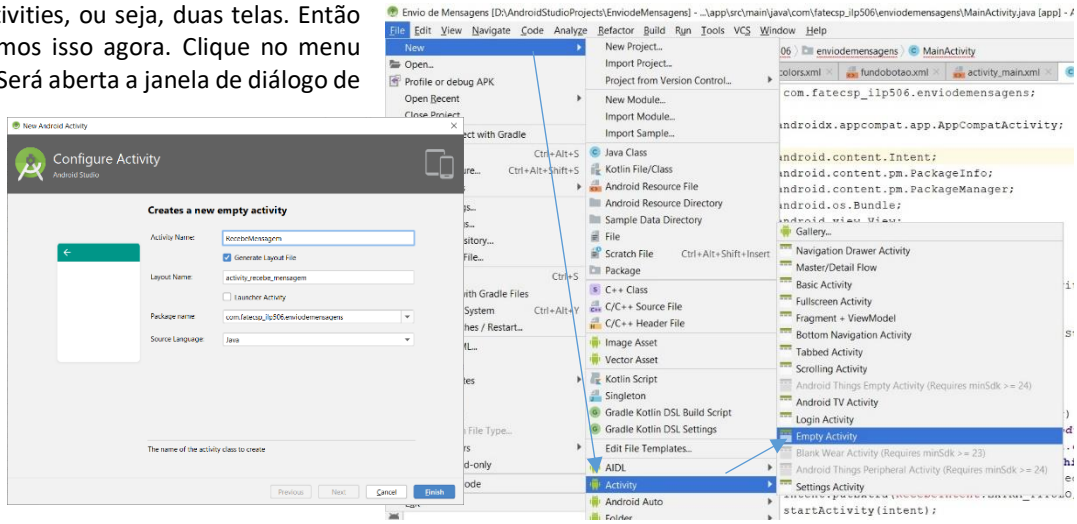
```
<android.widget.Button
    android:id="@+id/btnEnviaGeral"
    android:layout_width="0dp"
```

Criação da segunda Activity do Aplicativo

Este aplicativo tem duas Activities, ou seja, duas telas. Então precisamos criar a segunda e faremos isso agora. Clique no menu **New → Activity → Empty Activity**. Será aberta a janela de diálogo de configuração da Activity, na qual fornecemos o nome para ela.

O Android Studio criará os arquivos XML e Java da Activity.

Feito isto, basta criar o layout da tela e implementar o código conforme indicado acima.



Intent

Na plataforma Android a classe Intent é utilizada para prover um modo consistente de comunicação entre componentes que constituem os aplicativos. São eles: Activities, Broadcasts e Services. Esta classe contém a descrição de uma operação a ser executada e pode ser amplamente usada para diversas operações e interações entre aplicativos, sistema operacional e recursos dos dispositivos. São três as formas mais gerais de uso:

- Para iniciar uma atividade (Activity)

Como já foi falado mencionado, uma Activity representa uma tela em um aplicativo. É possível iniciar uma nova instância de uma Activity passando um Intent como parâmetro da função startActivity(). O Intent descreve a atividade a iniciar e carrega todos os dados necessários.

- Para fornecer uma transmissão (Broadcast)

Transmissão é uma mensagem que qualquer aplicativo pode receber. O sistema fornece diversas transmissões para eventos do sistema, como quando o sistema inicializa ou o dispositivo inicia o carregamento. Você pode fornecer uma transmissão a outros aplicativos passando um Intent a uma das funções sendBroadcast() ou sendOrderedBroadcast().

- Para iniciar um serviço (Service)

O Service é um componente que realiza operações em segundo plano. Devido a isso, não necessitam de interface do usuário, ou seja, não necessita uma tela, com seu layout XML. Serviços servem para tarefas como acessar uma rede (através de um plano de dados ou WiFi), fazer o download de um arquivo ou para reproduzir um arquivo de áudio. Nas versões anteriores ao Android 5.0 (API nível 21) é possível iniciar um serviço usando os métodos da classe Service. Pode-se iniciar um serviço passando um Intent à função startService(). O Intent descreve o serviço a iniciar e carrega todos os dados necessários. Com o Android 5.0 (API nível 21) e posteriores existe um recurso mais avançado para tais tarefas. Este recurso é a API chamada JobScheduler.

Neste Aplicativo vamos trabalhar com intents para iniciar Activities e Broadcasts.

Tipos de Intents

Há dois tipos de Intents: Explícitos e Implícitos

Os **Intents Explícitos** especificam qual aplicativo atenderá ao Intent, fornecendo o nome do pacote do aplicativo de destino ou o nome da classe de um componente totalmente qualificado. Normalmente, usa-se um Intent explícito para iniciar um componente no próprio aplicativo porque se sabe o nome de classe da atividade ou do serviço que se quer iniciar. Por exemplo, iniciar uma nova atividade em resposta a uma ação do usuário ou iniciar um serviço para fazer o download de um arquivo em segundo plano.

Os **Intents Implícitos** não nomeiam nenhum componente específico, mas declaram uma ação geral a realizar, o que permite que um componente de outro aplicativo a processe. Por exemplo, se você quiser exibir ao usuário uma localização em um mapa, pode usar um Intent implícito para solicitar que outro aplicativo capaz exiba uma localização especificada no mapa.

Intent para iniciar uma Activity específica (Intent Explícito)

Um Intent que será usado para iniciar uma Activity envolve duas telas do aplicativo: uma chamadora (que é a tela de onde parte o Intent) e a chamada (que é a tela que irá receber o Intent).

O código básico para realizar a chamada pode ser visto no método onClickIniciaIntent() presente no código de MainActivity.java. Esse código está reproduzido nas duas linhas a seguir:

```
Intent intent = new Intent(this, RecebeIntent.class);
startActivity(intent);
```

Isto é o necessário e a primeira linha declara e instancia o objeto da classe Intent. Na instanciação é preciso fornecer os parâmetros **this** que é o ponteiro do objeto da tela chamadora e a identificação de classe da tela chamada: **RecebeIntent.class** .

Na segunda linha a função startActivity() é usada para lançar a atividade.

Por sua vez, a Activity que recebe o Intent dessa forma não necessita conter qualquer código específico para tratar o Intent, ou seja, a Activity simplesmente é iniciada pelo Android e nada mais é necessário, a menos que se queira passar dados da Activity de origem para a Activity de destino.

Intent para iniciar uma Activity com o uso de Extras

Ao olhar atentamente o código do método btnIniciaIntentOnClick() de MainActivity, você perceberá que ele não contém apenas as duas linhas destacadas acima. Ele contém duas linhas a mais que servem para o fornecimento de parâmetros Extras ao Intent, de modo que informações possam ser passadas da Activity chamadora para a chamada.

```
intent.putExtra(RecebeIntent.EXTRA_MSG, edtMensagem.getText().toString());
intent.putExtra(RecebeIntent.EXTRA_ASSUNTO, edtTitulo.getText().toString());
```

A forma de passagem destes parâmetros usa o conceito de pares de dados constituídos de chave e valor. As chaves acima são as constantes públicas EXTRA_MSG e EXTRA_ASSUNTO definidas na classe RecebeIntent. Neste caso são passados dois strings, cada um associado à sua chave, respectivamente: o texto digitado no EditText edtMensagem e o texto digitado no EditText edtTitulo.

No lado da Activity chamada, quando há passagem de parâmetros, os mesmos devem ser recuperados e usados adequadamente. Assim, no código do método RecebeIntent.onCreate() é necessário acrescentar as linhas destacadas a seguir:

```
Intent intent = getIntent();
String msg = intent.getStringExtra(EXTRA_MSG);
String titulo = intent.getStringExtra(EXTRA_ ASSUNTO);
TextView txtMensagem = findViewById(R.id.txtMensagem);
txtMensagem.setText(msg);
TextView txtTitulo = findViewById(R.id.txtTitulo);
txtTitulo.setText(titulo);
```

Onde o método getIntent() é o responsável por recuperar o Intent que ativou a Activity, o método Intent.getStringExtra() recupera os valores passados, usando as chaves EXTRA_MSG e EXTRA_ASSUNTO para acesso aos mesmos e as linhas seguintes colocam esses valores (que são strings) nos TextViews existentes na Activity RecebeIntent.

Intent para iniciar uma Activity não especificada (Intent Implícito)

O código dos segundo e terceiro botões deste aplicativo executam a tarefa de iniciar um Intent de forma implícita. O código dos dois botões dispara um Intent implícito e ficará por conta do sistema operacional Android apresentar ao usuário uma lista de opções de aplicativos capazes de responder à ação desejada.

Note que neste caso, é usada outra versão do construtor, versão essa que possui apenas um parâmetro que especifica uma ação. No caso está sendo usada ACTION_SEND. Especificar a ação apenas não basta. Em linhas gerais, cada ação requer o fornecimento de informações adicionais. Por isso, é usado neste código o método Intent.setType(). Neste caso o parâmetro “text/plain” informa ao

Android que está sendo passado um texto sem formatação. E, por fim, é usada a constante EXTRA_TEXT para identificar o string passado.

```
Intent intent = new Intent(Intent.ACTION_SEND);
intent.setType("text/plain");
intent.putExtra(Intent.EXTRA_TEXT, edtMensagem.getText().toString());
if (intent.resolveActivity(getPackageManager()) != null)
   startActivity(intent);
```

A diferença entre o segundo e o terceiro botões é que no terceiro botão é implementado um código que força o Android a apresentar uma lista de aplicativos capazes de responder à ação desejada no Intent, cabendo a escolha final ao usuário do dispositivo.

No código do segundo botão não necessariamente será apresentada uma lista de opções (mas pode ocorrer). Se em um dispositivo específico estiver configurado um aplicativo padrão para responder à ação desejada, então o Android irá ativar esse aplicativo padrão. Caso não haja um aplicativo padrão, então o Android apresentará as opções.

Uma intent implícita pode ocasionar erro e cancelar a execução do seu aplicativo, caso não haja no dispositivo algum aplicativo capaz de processá-la. Por isso deve-se, nesses casos, verificar a disponibilidade de aplicativos capazes de responder à sua intent. Isso deve ser feito antes de executar o startActivity(). A função Intent.resolveActivity() faz essa verificação e retorna **null** caso não haja um aplicativo nessas condições.

Intent para executar um aplicativo específico

O quarto botão tem por objetivo direcionar o Intent para um aplicativo específico. É óbvio que esse código irá falhar caso tal aplicativo não esteja instalado no dispositivo. E se essa falha ocorrer o aplicativo chamador (que é o que estamos escrevendo) irá gerar um erro e será fechado pelo Android. Isso não pode acontecer. Por isso no código do método onClickBtnEnviaWhatsApp() é usado o mecanismo de tratamento de exceções try-catch visto no código a seguir.

```
public void btnEnviaWhatsAppOnClick(View view) {
   PackageManager pm = getPackageManager();
   try {
      EditText edtMensagem = findViewById(R.id.edtMensagem);
      Intent intent = new Intent(Intent.ACTION_SEND);
      intent.setType("text/plain");
      intent.putExtra(Intent.EXTRA_TEXT, edtMensagem.getText().toString());
      intent.putExtra(Intent.EXTRA_SUBJECT, edtTitulo.getText().toString()); // ← este código não terá efeito no app, pois o WhatsApp não o utiliza
      PackageInfo info = pm.getPackageInfo("com.whatsapp", PackageManager.GET_META_DATA);
      // Verifica se o pacote existe (app instalado). Se não estiver desvia para o bloco catch
      intent.setPackage("com.whatsapp");
      startActivity(Intent.createChooser(intent, "Share with"));
   }
   catch (PackageManager.NameNotFoundException e) {
      Toast.makeText(this, R.string.errowhatsapp, Toast.LENGTH_SHORT).show();
   }
}
```

A ideia aqui é bastante simples: queremos passar o texto digitado em edtMensagem para o aplicativo WhatsApp e vamos usar a classe PackageManager para identificar se o mesmo está instalado no dispositivo. Caso esteja, o Intent é instanciado e disparado. Caso não esteja, ocorrerá um erro no momento da execução do métido PackageManager.getPackageInfo() e por consequência desse erro a execução será desviada para a cláusula catch, que exibirá um Toast (mensagem em um balão temporário na tela).

Para saber mais sobre intents: https://developer.android.com/guide/components/intents-filters?hl=pt-br

Algo semelhante é feito no quinto botão – referente ao Google Docs. Analise o código do método `btnEnviaGoogleDocsOnClick()` e verifique que é análogo a este do WhatsApp, porém neste caso o EXTRA_SUBJECT terá um uso, que será o nome do arquivo a ser gravado no Google Drive. Por este motivo a extensão .TXT foi acrescentada.

Intent para executar um aplicativo específico – atualização para o Android 11

Tudo o que foi explicado acima não vai funcionar imediatamente nas versões 11 e superiores do Android.

A partir do Android 11, API 30 do sistema, foi introduzida uma nova política de segurança, denominada "política de visibilidade de pacotes" que faz com que o seu aplicativo não enxergue automaticamente os pacotes de outros aplicativos instalados no dispositivo. Segundo essa nova política o sistema fará uma filtragem dos pacotes que poderão ser visualizados pelo seu app.

Em outras palavras, o que foi mostrado na seção acima sobre o seu app enviar mensagens ao WhatsApp ou ao Google Docs não funcionará no Android 11, simplesmente porque o seu app não "enxergará" a existência desses pacotes no dispositivo e a função getPackageInfo() dará erro e disparará a exceção.

Esse recurso aumenta a segurança de um podo geral pois impede que aplicativos maliciosos façam um inventário de todos os pacotes instalados em um dispositivo e enviem (silenciosamente) a um servidor, por exemplo.

Porém, também foi providenciada uma forma de permitir o acesso do seu aplicativo a pacotes específicos. É possível listar no manifesto do aplicativo quais pacotes sua aplicação vai acessar. Para isso é preciso criar a seção <queries> no arquivo AndroidManifest.xml como mostrado a seguir:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.ilp506.osistemadeintents">

    <queries>
        <package android:name="com.whatsapp" />
        <package android:name="com.google.android.apps.docs" />
    </queries>

    <application
        android:allowBackup="true"
        android:label="@string/app_name"
        android:icon="@drawable/ic_icone_app"
        android:roundIcon="@drawable/ic_icone_app"
        android:supportsRtl="true"
        android:theme="@style/Theme.OSistemaDeIntents">
        <activity android:name=".RecebeIntent"></activity>
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

Visibilidade de pacotes a partir do Android 11
Esta seção deve ser incluída, listando os pacotes que sua aplicação pretende acessar.

Feito isso, seu aplicativo passará a enxergar os pacotes desejados e instalados no dispositivo.

Para saber mais acesse os links:

https://developer.android.com/training/package-visibility

https://developer.android.com/training/package-visibility/declaring

https://developer.android.com/training/package-visibility/use-cases