

# Roteiro de Laboratório 05

José Cassiano Grassi Gunji

## Abrindo Aplicativos Externos

Neste roteiro, vamos dar continuidade ao aplicativo desenvolvido no roteiro anterior. Vamos continuar o aplicativo modificando a `activity_mostre_mensagem.xml`.

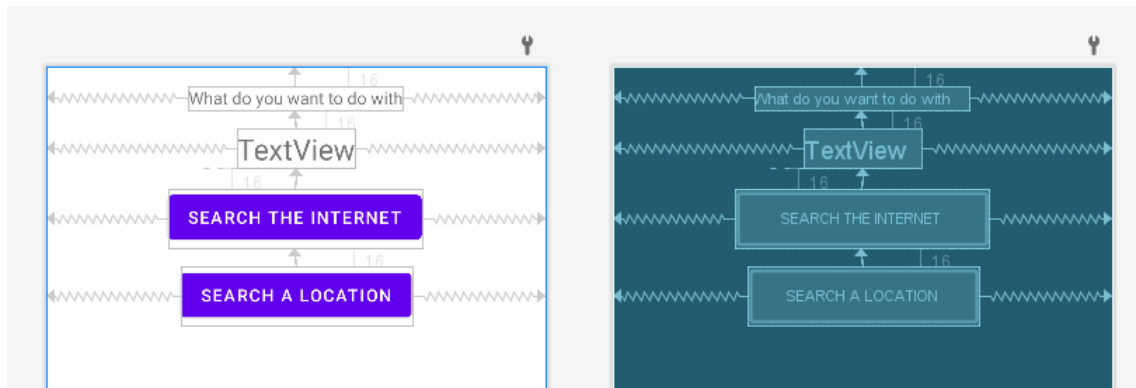


Figura 1: Novo layout da `activity_mostre-mensagem.xml`.

Altere o seu layout de modo a ficar semelhante ao mostrado na Figura 1. Crie os novos textos no arquivo `strings.xml` como fizemos anteriormente. Compare com a Figura 2. Note que adicionamos dois novos recursos, `erroIntent` e `escolhaApp`, que serão usados mais adiante.

```
<resources>
    <string name="app_name">Intent Example</string>
    <string name="digiteAlgo">Type something</string>
    <string name="enviar">Send</string>
    <string name="pergunta">What do you want to do with</string>
    <string name="internet">Search the Internet</string>
    <string name="mapa">Search a location</string>
    <string name="erroIntent">I couldn\'t launch the app.</string>
    <string name="escolhaApp">Choose the App to open.</string>
</resources>
```

Figura 2: Novos recursos no arquivo `strings.xml`.

Não se esqueça de fazer as traduções destes usando a ferramenta de tradução.



<div> + -  Show All Keys Show All Locales  ? </div>				
Key	Resource Folder	Untranslatable	Default Value	Portuguese (pt) in Brazil (BR)
app_name	app/src/main/res	<input type="checkbox"/>	Intent Example	Exemplo Intent
digiteAlgo	app/src/main/res	<input type="checkbox"/>	Type something	Digite algo
enviar	app/src/main/res	<input type="checkbox"/>	Send	Enviar
pergunta	app/src/main/res	<input type="checkbox"/>	What do you want to do with	O que você quer fazer com
internet	app/src/main/res	<input type="checkbox"/>	Search the Internet	Procurar na Internet
mapa	app/src/main/res	<input type="checkbox"/>	Search a location	Procurar um local
erroIntent	app/src/main/res	<input type="checkbox"/>	I couldn't launch the app.	Não consegui abrir o app.
escolhaApp	app/src/main/res	<input type="checkbox"/>	Choose the App to open.	Escolha o App para abrir.

Figura 3: Traduções dos recursos de texto.

Abra o arquivo `app->java->br.edu.cruzeirosul.exemplointent->MostreMensagemActivity.java`. Nele, crie o atributo *mensagem* e os métodos *procureNaInternet()* e *procureUmLocal()* como mostrado na Figura 4.

```

1  package br.edu.cruzeirodosul.exemplointent;
2
3  import ...
12
13  public class MostreMensagemActivity extends AppCompatActivity {
14
15      private String mensagem;
16
17      @Override
18      protected void onCreate(Bundle savedInstanceState) {
19          super.onCreate(savedInstanceState);
20          setContentView(R.layout.activity_mostre_mensagem);
21
22          // Obtém a intent que iniciou esta activity e extrai o string
23          Intent intent = getIntent();
24          mensagem = intent.getStringExtra(MainActivity.EXTRA_MESSAGE);
25
26          // Captura o TextView do layout e define o string como seu Text.
27          TextView textView = findViewById(R.id.textview);
28          textView.setText(mensagem);
29      }
30
31      public void procureNaInternet(View view){
32
33      }
34
35      public void procureUmLocal(View view){
36
37      }
38  }

```

Figura 4: Novos métodos e atributo na classe MostreMensagemActivity.

Note que alteramos a linha 24 também. Na versão anterior, a variável mensagem era declarada ali. Como agora mensagem é um atributo (para que possa ser usado facilmente pelos outros métodos), precisamos retirar a declaração String dali.

Volte ao editor de layout do *activity\_mostre\_mensagem.xml*. Clique no primeiro botão e mude seu atributo *onClick* para *procureNaInternet*. Clique no segundo botão e mude agora seu atributo *onClick* para *procureUmLocal*.

Na classe *MostreMensagemActivity.java*, vamos implementar o método *procureNaInternet()* como apresentado na Figura 5.

```

31 public void procureNaInternet(View view){
32     Uri pesquisa = Uri.parse("https://www.google.com/search?q=" + mensagem);
33     Intent intentPesquisa = new Intent(Intent.ACTION_VIEW, pesquisa);
34     try {
35         startActivity(intentPesquisa);
36     } catch(ActivityNotFoundException e){
37         String erro = getResources().getString(R.string.erroIntent);
38         Toast toast = Toast.makeText(context: this, erro, Toast.LENGTH_SHORT);
39         toast.show();
40     }
41 }

```

Figura 5: Implementação do método `procureNaInternet()`.

Na linha 32 estamos criando um objeto da classe `Uri` que representa o link que será enviado para o aplicativo de navegação na Internet padrão do dispositivo.

Na linha 33, criamos um *intent* implícito. Um *intent* implícito não especifica qual a classe que ele irá abrir. Esta escolha fica por conta do sistema operacional, que irá procurar um aplicativo capaz de atender a este *intent*. O *intent* que fizemos no roteiro anterior é um *intent* explícito, porque ele especifica qual a classe que ele deve abrir, que é esta *activity* que estamos editando agora.

Como queremos apenas exibir a página especificada na `Uri`, vamos usar a ação `ACTION_VIEW` ao criar o *intent*.

Para saber mais quais *intents* podem ser criados:

<https://developer.android.com/guide/components/intents-common>

Para saber as ações possíveis para um *intent*:

<https://developer.android.com/reference/android/content/Intent#standard-activity-actions>

Como o lançamento de um *intent* pode não funcionar, por exemplo, se não houver qualquer aplicativo instalado capaz de atender este *intent*, devemos fazer o seu lançamento dentro de um bloco *try*.

No bloco *catch*, estamos exibindo uma mensagem por meio de um *toast*, que é uma mensagem temporária apresentada na tela do usuário. Para garantir que a mensagem esteja no idioma correto, estamos usando um recurso de texto que foi criado como mostrado na Figura 2. Para recuperá-los usamos o método `getResources().getString()` como mostrado na linha 37.

Vamos agora implementar o método `procureUmLocal()` como mostrado na Figura 6.

```

43 public void procureUmLocal(View view){
44     Uri local = Uri.parse("geo:0,0?q=" + mensagem);
45     Intent intentLocal = new Intent(Intent.ACTION_VIEW, local);
46     String titulo = getResources().getString(R.string.escolhaApp);
47     Intent intentChooser = Intent.createChooser(intentLocal, titulo);
48     try {
49         startActivity(intentChooser);
50     } catch (ActivityNotFoundException e){
51         String erro = getResources().getString(R.string.erroIntent);
52         Toast toast = Toast.makeText(context, this, erro, Toast.LENGTH_SHORT);
53         toast.show();
54     }
55 }

```

Figura 6: Implementação do método `procureUmLocal()`.

Este método segue uma estratégia semelhante, com as seguintes mudanças: Na linha 44, a Uri segue o formato “geo:...”, que indica que trata-se de uma informação geográfica. Com isto, o sistema operacional sabe que apenas aplicativos que declaram ser capazes de tratar informações geográficas podem receber este *intent*.

Também estamos supondo que existem mais aplicativos instalados que sejam capazes de tratar informações geográficas. Assim, podemos fazer com que o sistema operacional apresente sempre um menu para que o usuário escolha qual o aplicativo a ser usado para receber este *intent*. Isto é feito criando-se o *intentChooser* como mostrado na linha 47. Note que na linha anterior, 46, estamos obtendo a tradução correta do título a ser apresentado no menu de escolha do aplicativo.

Teste seu aplicativo tanto em emuladores, mas não deixe de testar também em um dispositivo real.

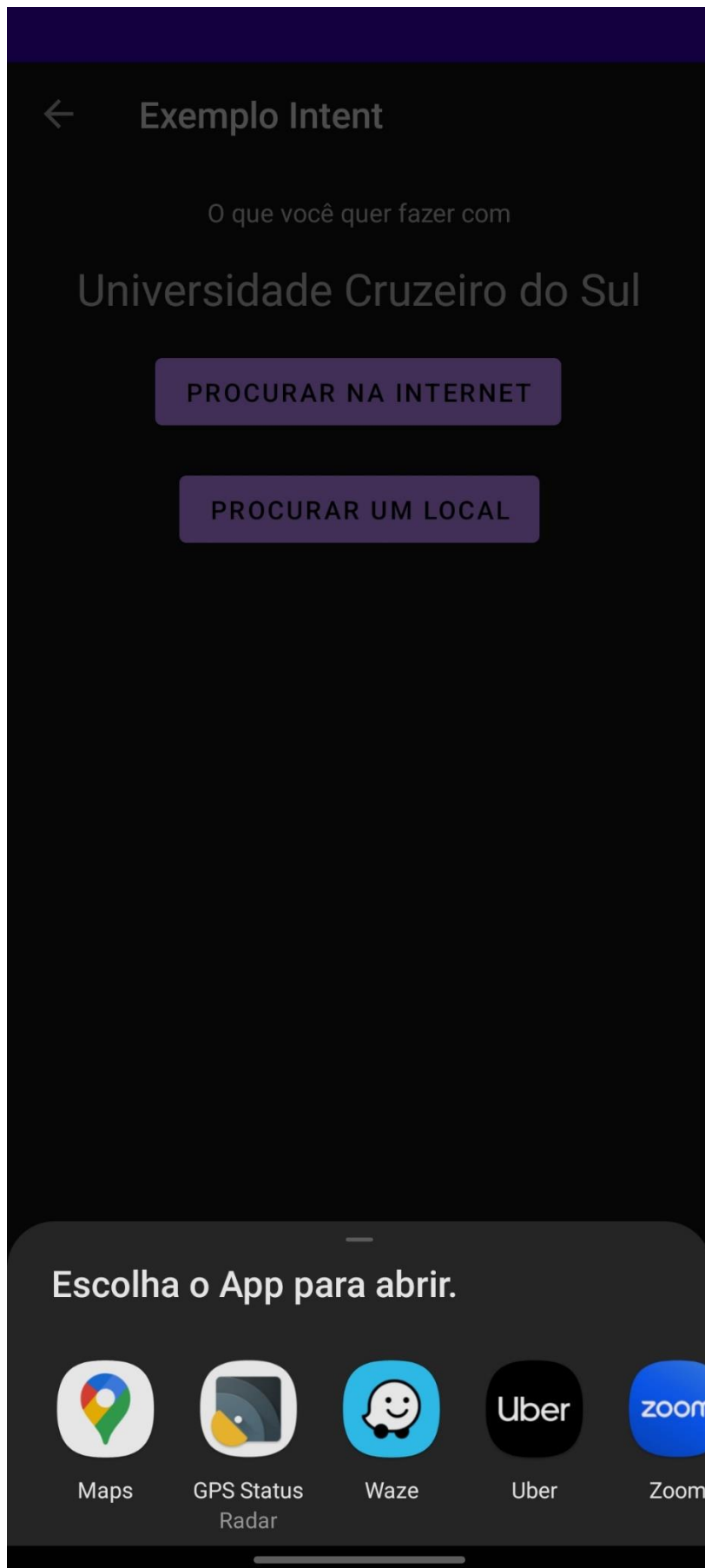


Figura 7: Menu de seleção de aplicativo para atender o intent.

## Referências

<https://developer.android.com/training/basics/firstapp>

DEITEL, Paul et al. Android para Programadores: uma abordagem baseada em aplicativos, 1ª Edição, 2013.