

Activity, Fragment e Intent

Bruno Lopes Alcantara Batista

Prazer em conhecer,

Bruno Lopes A Batista

Me. Ciência da Computação - UECE

Esp. em Desenvolvimento Web com JavaEE - FJN

Bel. em Sistemas de Informação - FJN

Professor Pesquisador - LARCES

Gerente de Projetos - LARCES

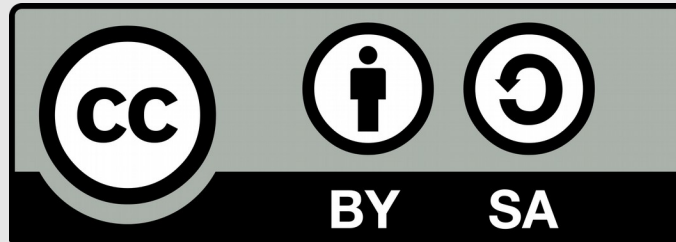
Gerente de Projetos - Athomustec

bruno@larces.uece.br

<http://www.larces.uece.br/bruno>



Copyleft



Esta obra está licenciado com uma Licença
[Creative Commons Atribuição-Compartilha Igual 4.0 Internacional](https://creativecommons.org/licenses/by-sa/4.0/)

Agenda

- ☐ Activity

- ☐ Ciclo de Vida

- ☐ Let's try it!

- ☐ Layout

- ☐ Let's try it!

- ☐ Intent

- ☐ Introdução

- ☐ Let's try it!

- ☐ Revisitando a Activity

- ☐ Intent implícita e explícita

Agenda

- ☐ Fragment
 - ☐ Introdução
 - ☐ Exemplo
 - ☐ Ciclo de Vida
 - ☐ Let's try it!
 - ☐ Fragments Dinâmicos
- ☐ Trabalho #1



Activity

Activity

- ❑ É uma tela na qual o usuário pode interagir.
- ❑ Uma aplicação geralmente possui várias Activities.
- ❑ Uma aplicação Android possui uma Activity principal (ponto de início da aplicação).
- ❑ A plataforma Android possui um Activity Manager.
- ❑ Sua função é de gerenciar o ciclo de vida de todas as Activities.

Activity

- Em “javanês”, uma Activity é uma classe Java que estende a Classe Activity (ou uma classe filha).

```
1 package br.edu.fa7.welcome;
2
3 import ...
4
5
6
7
8
9 public class MainActivity extends Activity {
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_main);
15     }
16
17 }
18
```


Activity

- Toda Activity deve ser declarado no arquivo de manifesto.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3          package="br.edu.fa7.welcome" >
4
5      <application
6          android:allowBackup="true"
7          android:icon="@mipmap/ic_launcher"
8          android:label="Welcome"
9          android:theme="@style/AppTheme" >
10         <activity
11             android:name=".MainActivity"
12             android:label="Welcome" >
13             <intent-filter>
14                 <action android:name="android.intent.action.MAIN" />
15
16                 <category android:name="android.intent.category.LAUNCHER" />
17             </intent-filter>
18         </activity>
19     </application>
20
21 </manifest>
22
```

Activity

- Durante esse curso, representaremos uma activity visualmente da seguinte maneira:





Activity

Ciclo de Vida

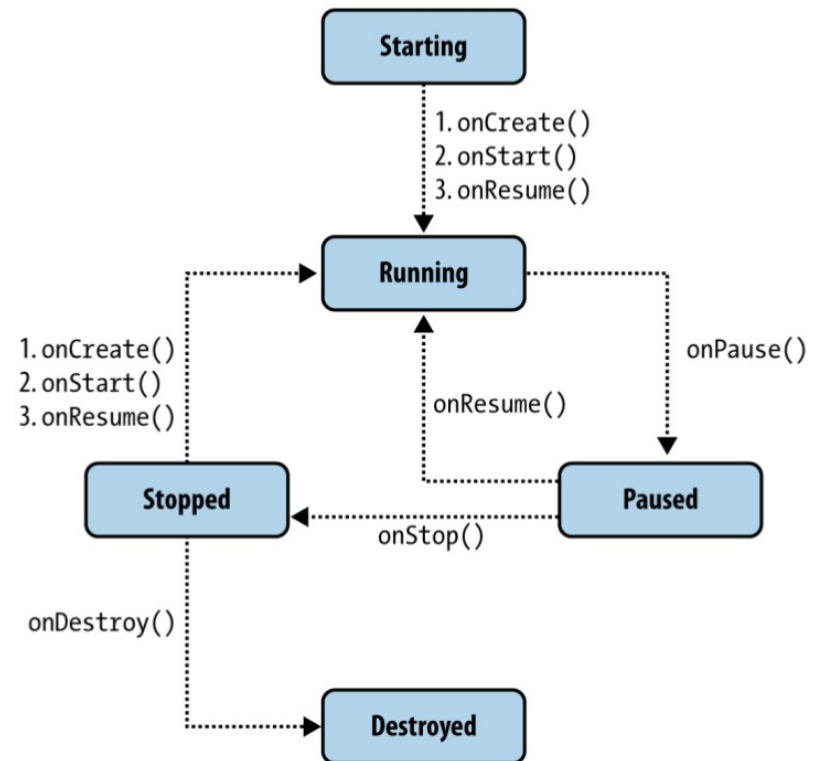
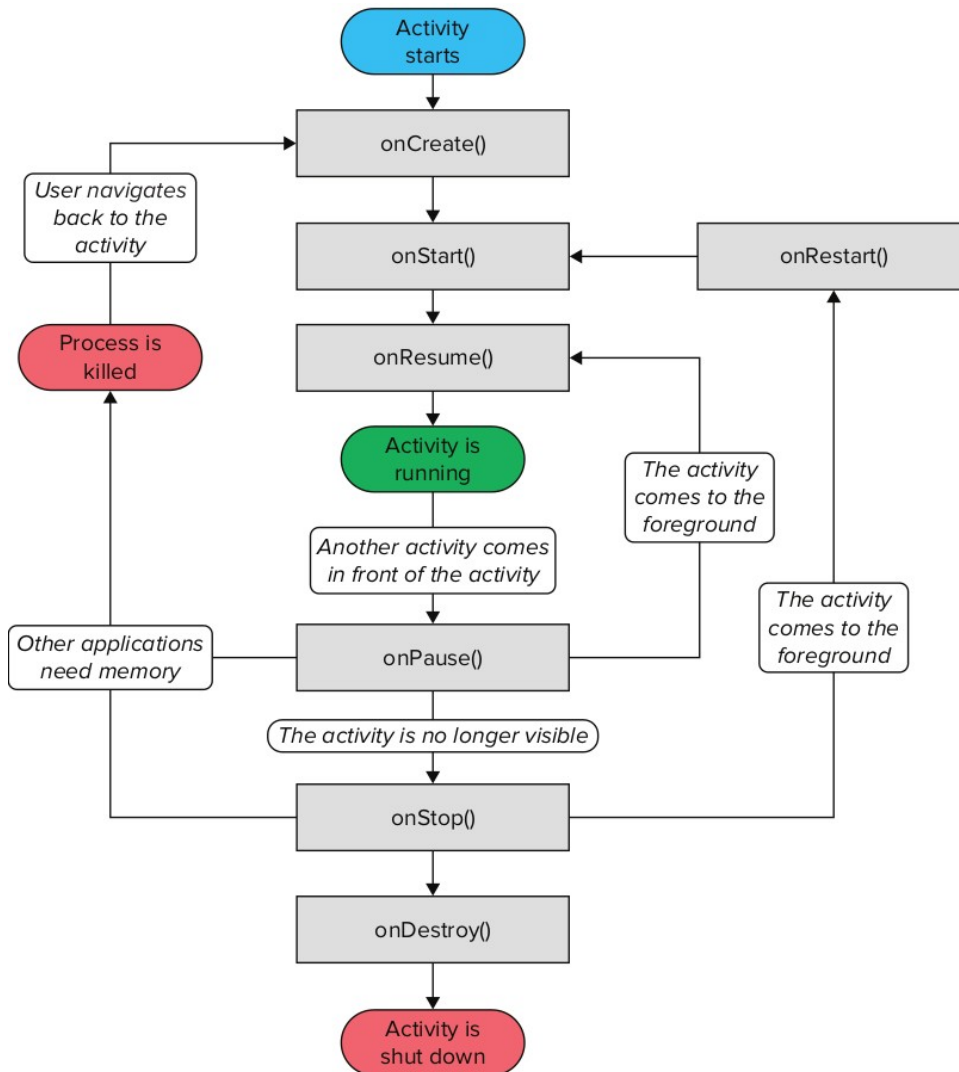
Activity

Ciclo de Vida

- ☐ Entender o ciclo de vida de uma Activity é de suma importância.
- ☐ Garante o bom funcionamento da Aplicação.
- ☐ O ciclo de vida é composto por sete estados:
 - ☐ onCreate
 - ☐ onRestart
 - ☐ onStart
 - ☐ onResume
 - ☐ onPause
 - ☐ onStop
 - ☐ onDestroy

Activity

Ciclo de Vida





Activity

Let's try it!

Activity

Let's try it!

- ☐ Criar um aplicativo Android chamado **Welcome**
- ☐ Criar uma Activity
- ☐ Implementar todos os métodos do ciclo de vida.
- ☐ Imprimir no console uma mensagem avisando a passagem por cada estado do ciclo de vida.

- ☐ OBS: Utilize a classe Log para imprimir uma mensagem no console:

```
Log.i ("Welcome", "Mensagem") ;
```



Activity

Layout

Activity

Layout



Como criamos um
layout para uma
Activity no
Android?

Activity

Layout

- ❑ Podemos criar um layout uma Activity de duas maneiras:
 - ❑ Via Java
 - ❑ Via XML
- ❑ O resultado de ambas as abordagens é o mesmo, entretanto:
 - ❑ Para a parte estática do layout, utilize XML
 - ❑ Para a parte dinâmica do layout, utilize Java

Activity

Layout

```
11
12 public class MainActivity extends Activity {
13
14     @Override
15     protected void onCreate(Bundle savedInstanceState) {
16         super.onCreate(savedInstanceState);
17
18         LinearLayout layout = new LinearLayout(this);
19         layout.setOrientation(LinearLayout.VERTICAL);
20         layout.setLayoutParams(new ViewGroup.LayoutParams(LinearLayout.LayoutParams.MATCH_PARENT, LinearLayout.LayoutParams.MATCH_PARENT));
21
22         TextView titleView = new TextView(this);
23         ViewGroup.LayoutParams lparams = new ViewGroup.LayoutParams(ViewGroup.LayoutParams.WRAP_CONTENT, ViewGroup.LayoutParams.WRAP_CONTENT);
24         titleView.setLayoutParams(lparams);
25         titleView.setText("Hello World");
26         layout.addView(titleView);
27
28         setContentView(layout);
29     }
30
31 }
32
```

Activity

Layout

```
1  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
2      xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
3      android:layout_height="match_parent" android:paddingLeft="16dp"
4      android:paddingRight="16dp"
5      android:paddingTop="16dp"
6      android:paddingBottom="16dp" tools:context=".MainActivity">
7
8      <TextView android:text="Hello world!" android:layout_width="wrap_content"
9          android:layout_height="wrap_content" />
10
11 </RelativeLayout>
12
```

```
12  public class MainActivity extends Activity {
13
14      @Override
15      protected void onCreate(Bundle savedInstanceState) {
16          super.onCreate(savedInstanceState);
17          setContentView(R.layout.activity_main);
18      }
19
20  }
21
```

Activity

Layout



Activity

Let's try it!

Activity

Let's try it!

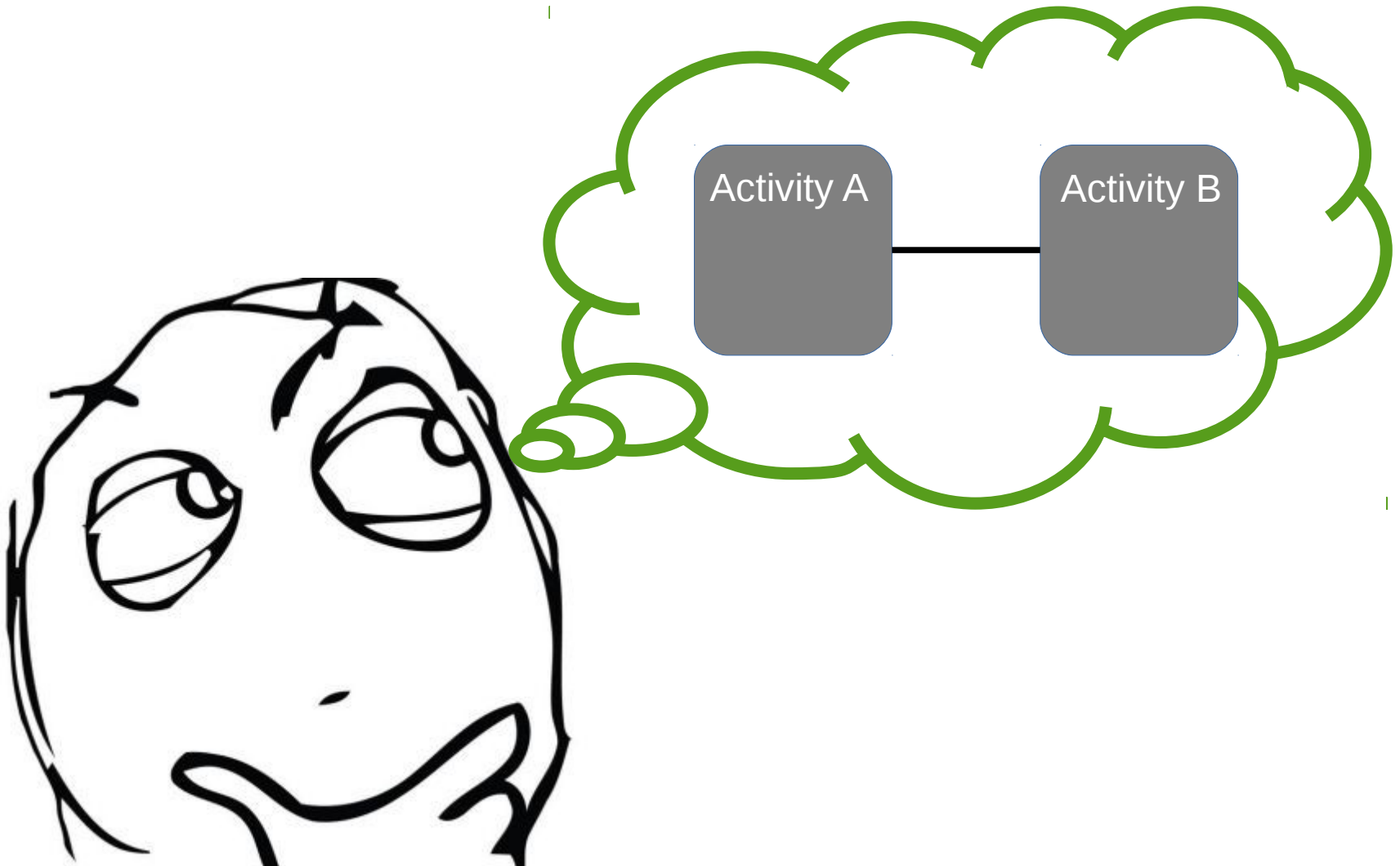
- ☐ No aplicativo **Welcome**
 - ☐ Implemente o layout mostrado via Java
 - ☐ Adicione um botão via Java
 - ☐ Implemente o layout mostrado via XML
 - ☐ Adicione um botão via XML

- ☐ OBS: A classe que representa um botão no código Java é a classe **Button**



Intent

Intent



Intent

Introdução

- ❑ Aplicações Android são compostas por várias Activities.
- ❑ Como podemos interliga-las?
- ❑ No Android utilizamos uma Intent para esse propósito.
- ❑ Intents são mensagens que são enviadas ao Android.
- ❑ Intents podem:
 - ❑ Abrir uma Activity
 - ❑ Iniciar, parar ou vincular um Service
 - ❑ Ouvir mensagens de broadcast

Intent

Introdução

- ❑ Intents são assíncronas!
- ❑ Podem ser explícitas ou implícitas:
 - ❑ **Explícita:** o componente que enviou uma Intent especifica claramente qual componente deve recebê-la
 - ❑ **Implícita:** o componente que enviou uma Intent não especifica claramente qual componente deve recebê-la.
- ❑ Uma Intent pode carregar consigo dados adicionais.

Intent

Let's try it!

Intent

Let's try it!

- ❑ Criar uma nova Activity chamada NewActivity no projeto **Welcome**
- ❑ Adicionar um botão na MainActivity e capturar o evento de clique do botão
- ❑ Abrir a NewActivity Quando o Botão for pressionado

- ❑ **OBS 1:** Para “ouvir” o clique do botão, utilize o listener `setOnClickListener()` do objeto do botão;
- ❑ **OBS 2:** Para abrir uma Activity use o método `startActivity();`

Intent

Revisitando Activity

Intent

Revisitando Activity

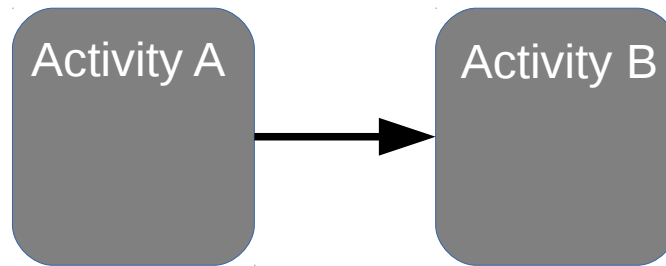


Beleza, aprendi como abrir uma Activity. Mas como faço para passar parâmetros entre elas?

Intent

Revisitando Activity

- Tomemos esse cenário como exemplo





- Para enviar dados da Activity A para Activity B podemos utilizar a Intent que utilizamos para iniciar a Activtiy B.
 - Utilizando um objeto Bundle
 - Utilizando os métodos da própria Intent

Intent

Revisitando Activity



□ Via Bundle (Activity A):

```
12
13  public class MainActivity extends Activity {
14
15     @Override
16      protected void onCreate(Bundle savedInstanceState) {
17         super.onCreate(savedInstanceState);
18         setContentView(R.layout.activity_main);
19
20         Bundle b = new Bundle();
21         b.putChar("a", 'a');
22         b.putByte("b", (byte) 12);
23         b.putInt("c", 2345);
24         b.putString("s", "Hello World");
25
26         Intent it = new Intent(this, NewActivity.class);
27         it.putExtras(b);
28         startActivity(it);
29     }
30 }
31
32
33
```

Intent

Revisitando Activity




□ Via Bundle (Activity B):

```
12
13  public class MainActivity extends Activity {
14
15     @Override
16      protected void onCreate(Bundle savedInstanceState) {
17         super.onCreate(savedInstanceState);
18         setContentView(R.layout.activity_main);
19
20         Bundle b = new Bundle();
21         b.putChar("a", 'a');
22         b.putByte("b", (byte) 12);
23         b.putInt("c", 2345);
24         b.putString("s", "Hello World");
25
26         Intent it = new Intent(this, NewActivity.class);
27         it.putExtras(b);
28         startActivity(it);
29     }
30 }
31
32
33
```

Intent

Revisitando Activity

□ Via métodos da Intent (Activity A):

```
13  public class MainActivity extends Activity {  
14  
15     @Override  
16   protected void onCreate(Bundle savedInstanceState) {  
17         super.onCreate(savedInstanceState);  
18         setContentView(R.layout.activity_main);  
19  
20         Intent it = new Intent(this, NewActivity.class);  
21         it.putExtra("a", 'a');  
22         it.putExtra("b", (byte) 12);  
23         it.putExtra("c", 2345);  
24         it.putExtra("s", "Hello World");  
25         startActivity(it);  
26  
27     }  
28  
29 }  
30
```

Intent

Revisitando Activity

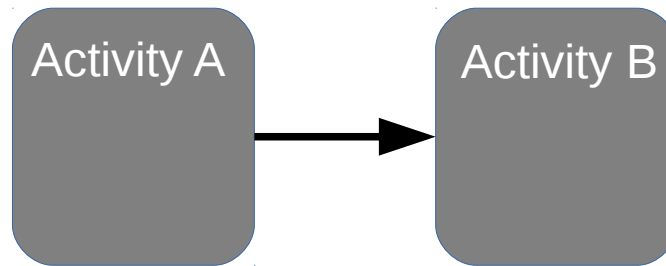
□ Via métodos da Intent (Activity B):

```
10 public class NewActivity extends Activity {  
11  
12     @Override  
13     protected void onCreate(Bundle savedInstanceState) {  
14         super.onCreate(savedInstanceState);  
15         setContentView(R.layout.activity_new);  
16  
17         Bundle bundle = getIntent().getExtras();  
18         char a = bundle.getChar("a");  
19         byte b = bundle.getBytes("b");  
20         int c = bundle.getInt("c");  
21         String s = bundle.getString("s");  
22     }  
23 }  
24  
25
```

Intent

Revisitando Activity

- Relembrando o cenário



- Para retornar dados da Activity B para Activity A (leve em consideração que A anteriormente invocou B) é um pouco mais complicado.
- Ao invés de usar o método `startActivity()`, usaremos o método **`startActivityForResult()`**.
- E devemos implementar na Activity A o método **`onActivityResult()`**

Intent

Revisitando Activity

□ Activity A

```
12
13 public class MainActivity extends Activity {
14
15     private static final short NEW_ACTIVITY_ID = 1;
16
17     @Override
18     protected void onCreate(Bundle savedInstanceState) {
19         super.onCreate(savedInstanceState);
20         setContentView(R.layout.activity_main);
21
22         Intent it = new Intent(this, NewActivity.class);
23         startActivityForResult(it, NEW_ACTIVITY_ID);
24     }
25
26     @Override
27     protected void onActivityResult(int requestCode, int resultCode, Intent data) {
28         super.onActivityResult(requestCode, resultCode, data);
29         switch (requestCode){
30             case NEW_ACTIVITY_ID:
31                 if(resultCode == RESULT_OK){
32                     // TODO: Código para quando ocorrer tudo OK!
33                 } else {
34                     // TODO: Código para quando algo der errado!
35                 }
36             }
37         }
38     }
39 }
40
41
```

Intent

Revisitando Activity

□ Activity B

```
10 public class NewActivity extends Activity {  
11  
12     @Override  
13     protected void onCreate(Bundle savedInstanceState) {  
14         super.onCreate(savedInstanceState);  
15         setContentView(R.layout.activity_new);  
16  
17         Bundle b = new Bundle();  
18         b.putString("user.name", "Bruno Lopes");  
19         b.putString("user.email", "bruno@larces.uece.br");  
20  
21         Intent it = new Intent();  
22         it.putExtras(b);  
23  
24         setResult(RESULT_OK, it);  
25         finish();  
26     }  
27 }  
28  
29
```

Finaliza a Activity

Intent

Explicita e Implícita

Intent

Explícita e Implícita

- Ao chamar uma nova Activity nos exemplos anteriores, utilizamos uma Intent Explícita.

```
17
18
19
20
21
22
23
24
25
26
```

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

    Intent it = new Intent(this, NewActivity.class);
    startActivity(it);
}
```

- Podemos chamar outras Activities de forma implícita

Intent

Explicita e Implícita



Intent

Explícita e Implícita

```
15 public class MainActivity extends Activity {
16
17     private static final short NEW_ACTIVITY_ID = 1;
18
19     @Override
20     protected void onCreate(Bundle savedInstanceState) {
21         super.onCreate(savedInstanceState);
22         setContentView(R.layout.activity_main);
23
24         Intent sendIntent = new Intent(Intent.ACTION_SEND);
25         sendIntent.putExtra(Intent.EXTRA_TEXT, "Lorem ipsum dolor sit amet. ");
26         sendIntent.setType("text/plain");
27
28         if(sendIntent.resolveActivity(getPackageManager()) != null){
29             startActivity(sendIntent);
30         }
31     }
32 }
33
34
```

Intent.ACTION_SEND == "android.intent.action.SEND"

Intent

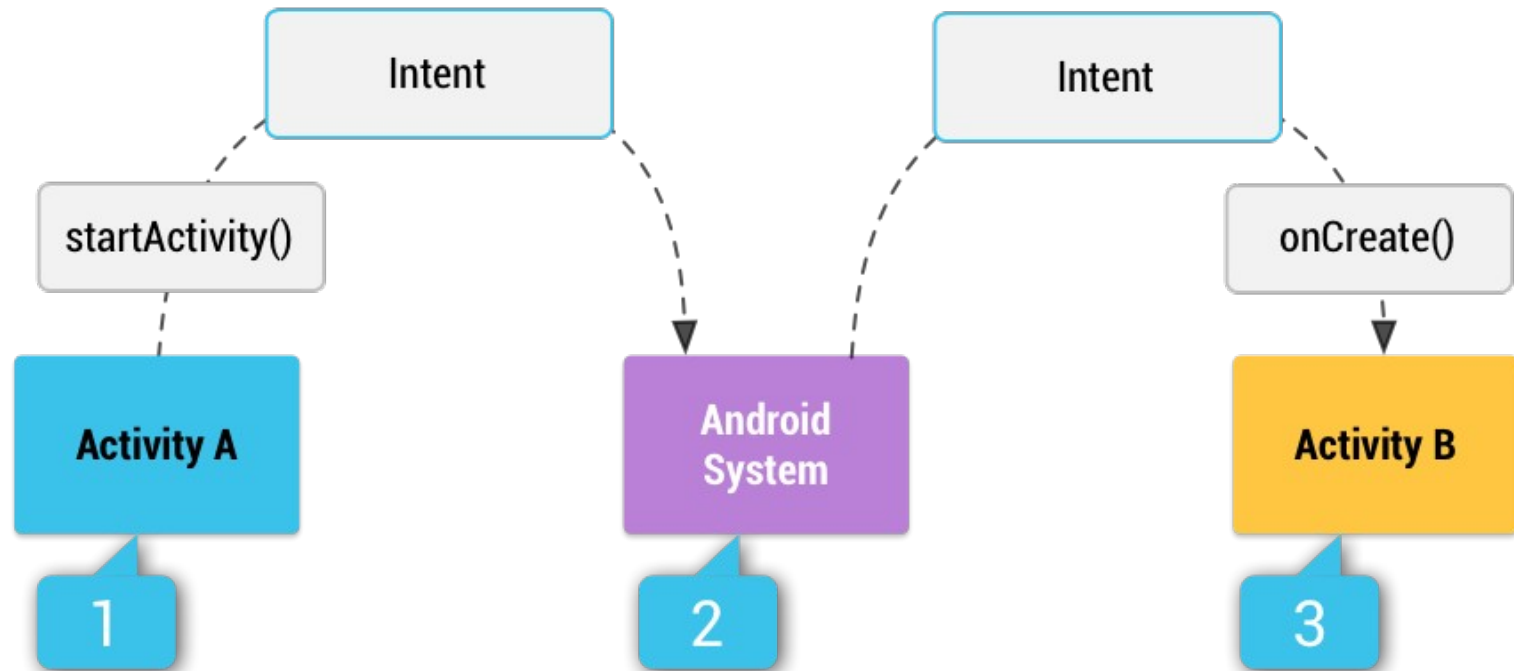
Explícita e Implícita

```
15 public class MainActivity extends Activity {
16
17     private static final short NEW_ACTIVITY_ID = 1;
18
19     @Override
20     protected void onCreate(Bundle savedInstanceState) {
21         super.onCreate(savedInstanceState);
22         setContentView(R.layout.activity_main);
23
24         Intent sendIntent = new Intent(Intent.ACTION_SEND);
25         sendIntent.putExtra(Intent.EXTRA_TEXT, "Lorem ipsum dolor sit amet. ");
26         sendIntent.setType("text/plain");
27
28         if(sendIntent.resolveActivity(getPackageManager()) != null){
29             startActivity(sendIntent);
30         }
31     }
32 }
33
34
```

`Intent.ACTION_SEND == "android.intent.action.SEND"`

Intent

Explícita e Implícita



Intent

Explícita e Implícita

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="br.edu.fa7.welcome" >
4
5      <application
6          android:allowBackup="true"
7          android:icon="@mipmap/ic_launcher"
8          android:label="Welcome"
9          android:theme="@style/AppTheme" >
10
11          <activity
12              android:name=".MainActivity"
13              android:label="Welcome" >
14              <intent-filter>
15                  <action android:name="android.intent.action.MAIN" />
16                  <category android:name="android.intent.category.LAUNCHER" />
17              </intent-filter>
18          </activity>
19
20          <activity android:name=".NewActivity">
21              <intent-filter>
22                  <action android:name="android.intent.action.SEND"/>
23                  <category android:name="ANDROID.INTENT.CATEGORY.DEFAULT"/>
24              </intent-filter>
25          </activity>
26
27      </application>
28
29  </manifest>
30
```



Fragment

Fragment



Fragment

Introdução

- ❑ Como trabalhar com um layout modular em uma Activity?
- ❑ Tarefa muito difícil de realizar com uma Activity!
- ❑ Mas como gerenciar melhor o espaço de tela de um tablet?
- ❑ No Android 3.0 foi criado o conceito de Fragment.
- ❑ Um Fragment se comporta como uma “mini-activity”
- ❑ Possui seu próprio ciclo de vida
- ❑ A Activity enxerga o Fragment como uma View (componente).

Fragment

Introdução

- ❑ A Activity enxerga o Fragment como uma View (componente).
- ❑ Sempre um ou mais Fragments são embarcados dentro de uma Activity.
- ❑ Durante esse curso, representaremos um fragment visualmente da seguinte maneira:



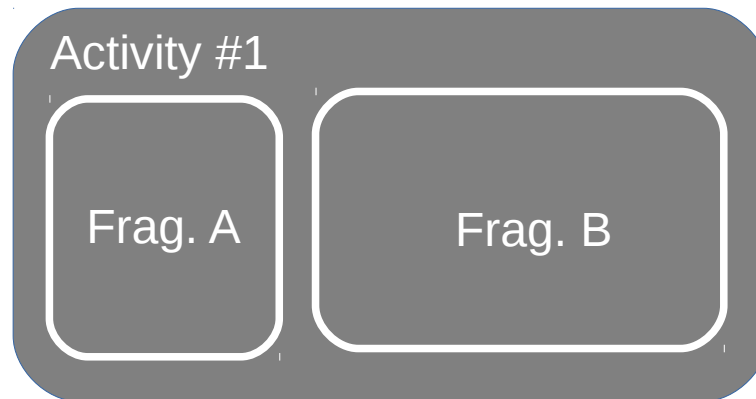
Fragment

Exemplo

Fragment

Exemplo

- ❑ Vamos trabalhar com o seguinte cenário:



- ❑ Vamos criar dois fragmentos e embarcá-los em uma Activity.

Fragment

Exemplo

❑ Criar o layout do Fragment #1

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      android:layout_width="match_parent"
4      android:layout_height="match_parent"
5      android:orientation="vertical"
6      android:background="#FFFE00">
7
8      <TextView
9          android:layout_width="match_parent"
10         android:layout_height="match_parent"
11         android:text="Este é o Fragment #1"
12         android:textColor="#000000"
13         android:textSize="35sp"
14         android:layout_gravity="center"
15         android:gravity="center"/>
16
17  </LinearLayout>
18
```

Fragment

Exemplo

❑ Criar o layout do Fragment #2

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      android:layout_width="match_parent"
4      android:layout_height="match_parent"
5      android:orientation="vertical"
6      android:background="#00FF00">
7
8      <TextView
9          android:layout_width="match_parent"
10         android:layout_height="match_parent"
11         android:text="Este é o Fragment #2"
12         android:textColor="#000000"
13         android:textSize="35sp"
14         android:layout_gravity="center"
15         android:gravity="center"/>
16
17  </LinearLayout>
18
```

Fragment

Exemplo

❑ Criar a classe do Fragment #1

```
1  package br.edu.fa7.welcome;
2
3  import android.animation.Animator;
4  import android.app.Fragment;
5  import android.os.Bundle;
6  import android.view.LayoutInflater;
7  import android.view.View;
8  import android.view.ViewGroup;
9
10 /**
11  * Created by bruno on 22/07/15.
12  */
13 public class Fragment1 extends Fragment {
14
15     @Override
16     public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
17         return inflater.inflate(R.layout.fragment1, container, false);
18     }
19 }
20
```

Fragment

Exemplo

❑ Criar a classe do Fragment #2

```
1  package br.edu.fa7.welcome;
2
3  import ...
9
10 /**
11  * Created by bruno on 22/07/15.
12  */
13 public class Fragment2 extends Fragment {
14
15     @Override
16     public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
17         return inflater.inflate(R.layout.fragment2, container, false);
18     }
19 }
20
```


Fragment

Exemplo

□ Activity principal

```
1  package br.edu.fa7.welcome;
2
3  +import ...
13
14
15  public class MainActivity extends Activity {
16
17      private static final short NEW_ACTIVITY_ID = 1;
18
19      @Override
20      protected void onCreate(Bundle savedInstanceState) {
21          super.onCreate(savedInstanceState);
22          setContentView(R.layout.main_activity_fragment);
23      }
24  }
25
```

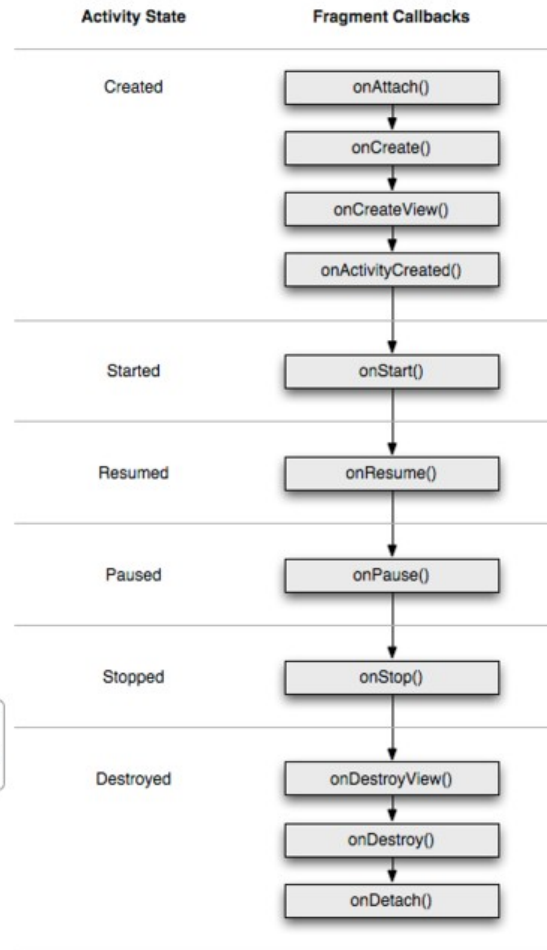
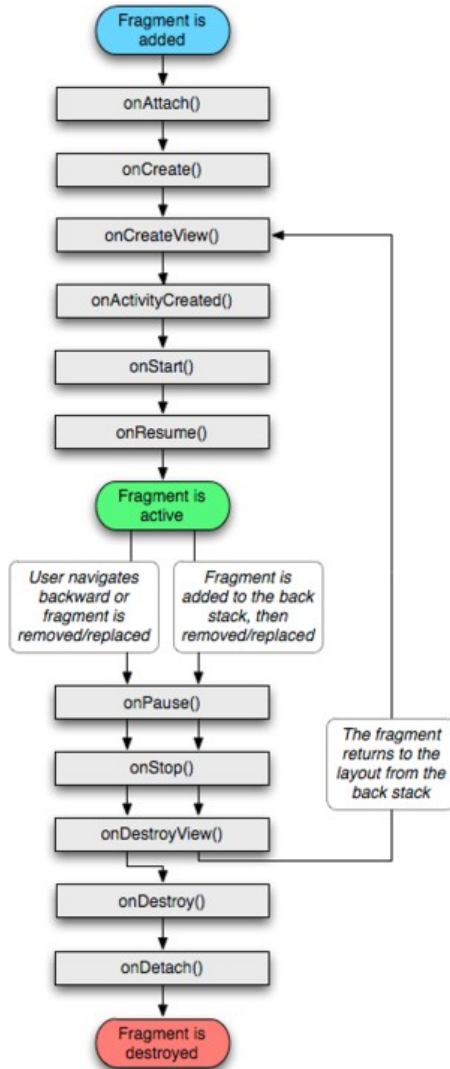


Fragment

Ciclo de Vida

Fragment

Ciclo de vida



Fragment

Let's try it!

Fragment

Let's try it!

- ❑ Criar um aplicativo Android chamado **FragmentLifecycle**
- ❑ Criar uma Activity e dois Fragments
- ❑ Implementar todos os métodos do ciclo de vida em ambos os Fragments.
- ❑ Imprimir no console uma mensagem avisando a passagem por cada estado do ciclo de vida.
- ❑ OBS: Utilize a classe Log para imprimir uma mensagem no console:

```
Log.info("Welcome", "Mensagem");
```

Fragment

Fragments Dinâmicos

Fragment

Fragments Dinâmicos

- ❑ É possível adicionar fragments de forma dinâmica.
- ❑ Esse é o verdadeiro poder do Fragment.
- ❑ Adicionar um Fragment dinamicamente é uma tarefa fácil.
- ❑ Na Activity necessitamos de uma instancia de *FragmentManager* e uma de *FragmentTransaction*.

Fragment

Fragments Dinâmicos

- ❑ No arquivo XML de layout da Activity do exemplo anterior, vamos comentar a declaração dos Fragments

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3   xmlns:tools="http://schemas.android.com/tools"
4   android:layout_width="match_parent"
5   android:layout_height="match_parent"
6   android:orientation="horizontal">
7
8   <!--
9   <fragment
10     android:layout_width="0px"
11     android:layout_weight="1"
12     android:layout_height="match_parent"
13     android:name="br.edu.fa7.welcome.Fragment1"
14     android:id="@+id/fragment1"
15     tools:layout="@layout/fragment1" />
16
17   <fragment
18     android:layout_width="0px"
19     android:layout_weight="1"
20     android:layout_height="match_parent"
21     android:name="br.edu.fa7.welcome.Fragment2"
22     android:id="@+id/fragment2"
23     tools:layout="@layout/fragment2" />
24   -->
25
26 </LinearLayout>
27
```


Fragment

Fragments Dinâmicos

- No arquivo XML de layout da Activity do exemplo anterior, vamos comentar a declaração dos Fragments

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3   xmlns:tools="http://schemas.android.com/tools"
4   android:layout_width="match_parent"
5   android:layout_height="match_parent"
6   android:orientation="horizontal">
7
8   <!--
9   <fragment
10     android:layout_width="0px"
11     android:layout_weight="1"
12     android:layout_height="match_parent"
13     android:name="br.edu.fa7.welcome.Fragment1"
14     android:id="@+id/fragment1"
15     tools:layout="@layout/fragment1" />
16
17   <fragment
18     android:layout_width="0px"
19     android:layout_weight="1"
20     android:layout_height="match_parent"
21     android:name="br.edu.fa7.welcome.Fragment2"
22     android:id="@+id/fragment2"
23     tools:layout="@layout/fragment2" />
24   -->
25
26 </LinearLayout>
27
```

Fragment

Fragments Dinâmicos

□ Na classe da Activity, fazemos:

```
19
20 public class MainActivity extends Activity {
21
22     private static final short NEW_ACTIVITY_ID = 1;
23
24     @Override
25     protected void onCreate(Bundle savedInstanceState) {
26         super.onCreate(savedInstanceState);
27
28         // Recupera uma instancia do fragmentManager e FragmentTransaction
29         fragmentManager = getFragmentManager();
30         fragmentTransaction = fragmentManager.beginTransaction();
31
32         // Recupera as informações do display
33         WindowManager windowManager = getWindowManager();
34         Display display = windowManager.getDefaultDisplay();
35
36         // Recupera as medidas do display
37         Point sizeDisplay = new Point();
38         display.getSize(sizeDisplay);
39
40         // Verifica se o dispositivo esta em modo retrato ou paisagem
41         if(sizeDisplay.x > sizeDisplay.y){
42             // Modo paisagem
43             Fragment1 fragment1 = new Fragment1();
44             fragmentTransaction.replace(android.R.id.content, fragment1);
45         } else {
46             // Modo retrato
47             Fragment2 fragment2 = new Fragment2();
48             fragmentTransaction.replace(android.R.id.content, fragment2);
49         }
50
51         // Aplica o fragment na Activity
52         fragmentTransaction.commit();
53     }
54 }
55
56
57
```

Trabalho #1

Trabalho #1

- ❑ Criar um aplicativo Android que contenha 3 botões, onde cada botão deve executar uma Intent Implícita que solicitará um serviço Android e exibir a resposta da Intent, caso tenha, da melhor forma apropriada.
- ❑ As Intents devem ser escolhidas, a seu critério, a partir do link.
`http://developer.android.com/guide/components/intents-common.html`
- ❑ O código fonte deve ser enviado para o e-mail `brunolopesjn@gmail.com` com o assunto **Trabalho #1** até o dia 17/08.

Obrigado!

A word cloud featuring the phrase 'thank you' in numerous languages and scripts. The words are arranged in a circular pattern around the central text 'thank you'. The colors of the words vary, including shades of blue, red, green, yellow, and orange. The font sizes are also varied, with 'thank you' being the largest and most prominent.

thank you

danke 謝謝

ngiyabonga

teşekkür ederim

dank je

спасибо

gracias

tapadh leat

moichchakkeram

go raibh maith agat

arigatō

takk

dakujem

merci

ευχαριστώ

grazie

kop khun krap

sukriya

terima kasih

감사합니다

sagolun

dziękuję

hvala

maururu

bedankt

obrigado