



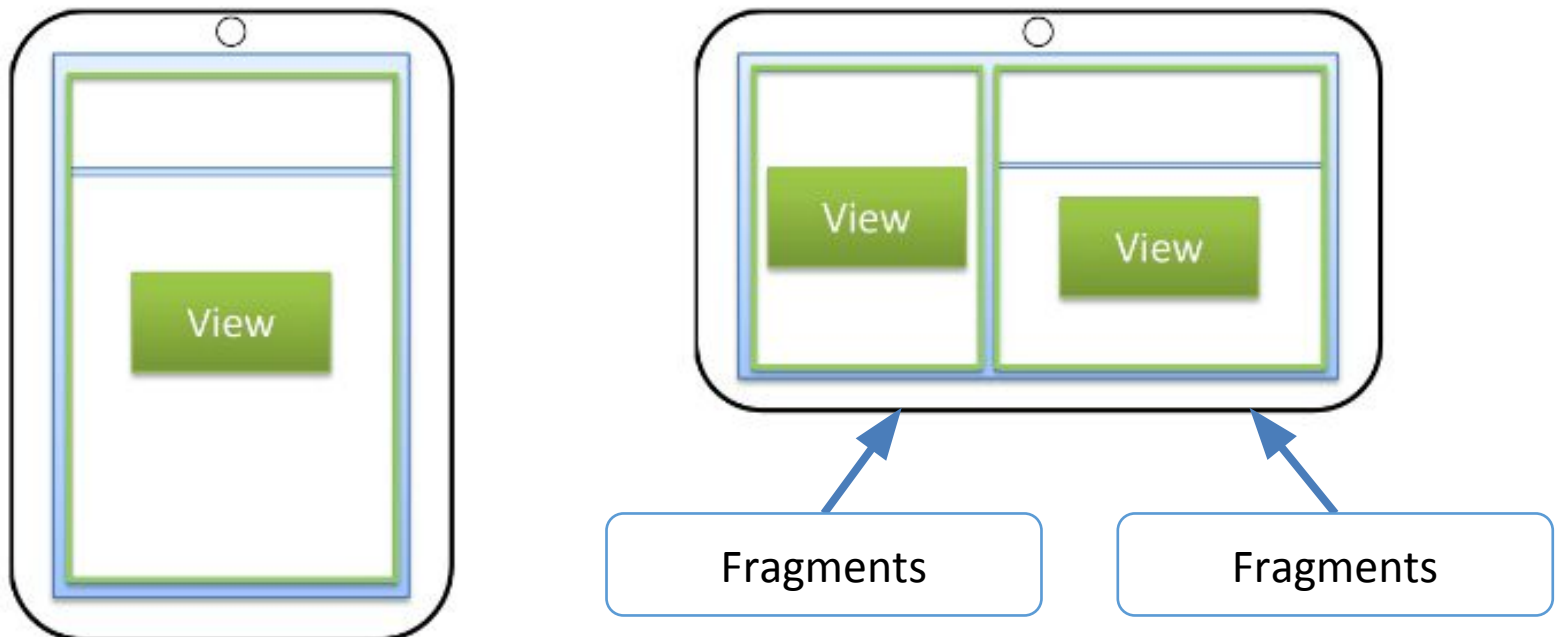
Fragments

Professor Emerson Alencar

emerson@imd.ufrn.br

FRAGMENTS - SURGIMENTO

- ▶ Uma Activity é associada a uma view
- ▶ Surgiram a partir da versão 3.0 do Android
- ▶ Com o surgimento de Telas maiores, passou a existir a necessidade de dividir a tela



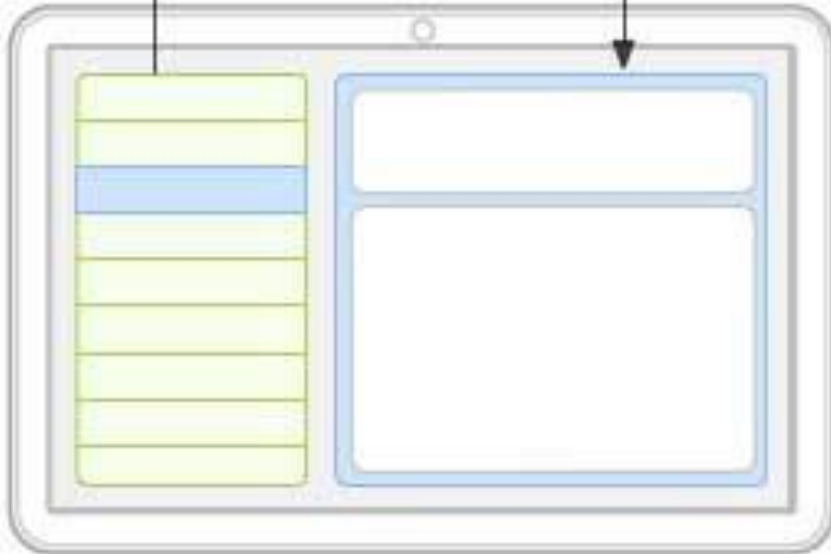
FRAGMENTS - SURGIMENTO

- ▶ Um Fragment é um componente
 - Que gerencia sua própria view
 - Gerencia os eventos da sua view
 - Tem seu próprio ciclo de vida
 - Está atrelado a uma activity
- ▶ Uma activity pode ter diversos Fragments associados a ela
- ▶ Um Fragment deve ser criado de forma modular
 - Assim ele pode ser reaproveitado em várias activities

FRAGMENTS

TABLET

A seleção de um item atualiza o Fragment B



A activity A contém Fragment A e Fragment B

SMARTPHONE

A seleção de um item inicia a Activity B



Activity A contém um Fragment A



Activity B contém um Fragment B

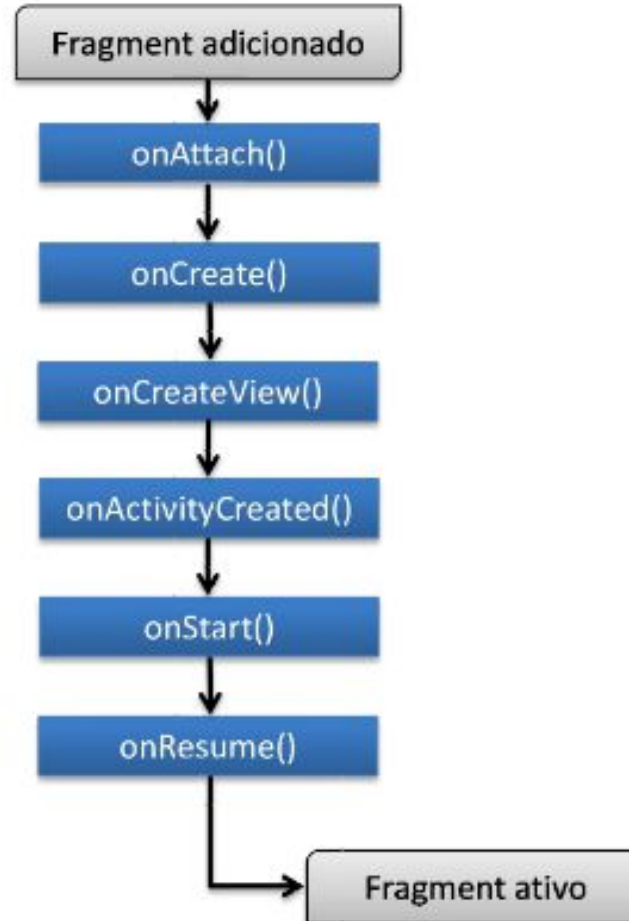
FRAGMENTS - CRIAÇÃO

- ▶ Um Fragment é uma classe que herda de **Fragment**

```
8
9 class MyFragment : Fragment(){
10
11
12     override fun onCreate(savedInstanceState: Bundle?) {
13         super.onCreate(savedInstanceState)
14     }
15
16     override fun onPause() {
17         super.onPause()
18     }
19
20     override fun onCreateView(inflater: LayoutInflater,
21                               container: ViewGroup?, savedInstanceState: Bundle?): View? {
22
23         return inflater.inflate(R.layout.fragment_layout, container, attachToRoot: false)
24     }
25 }
```

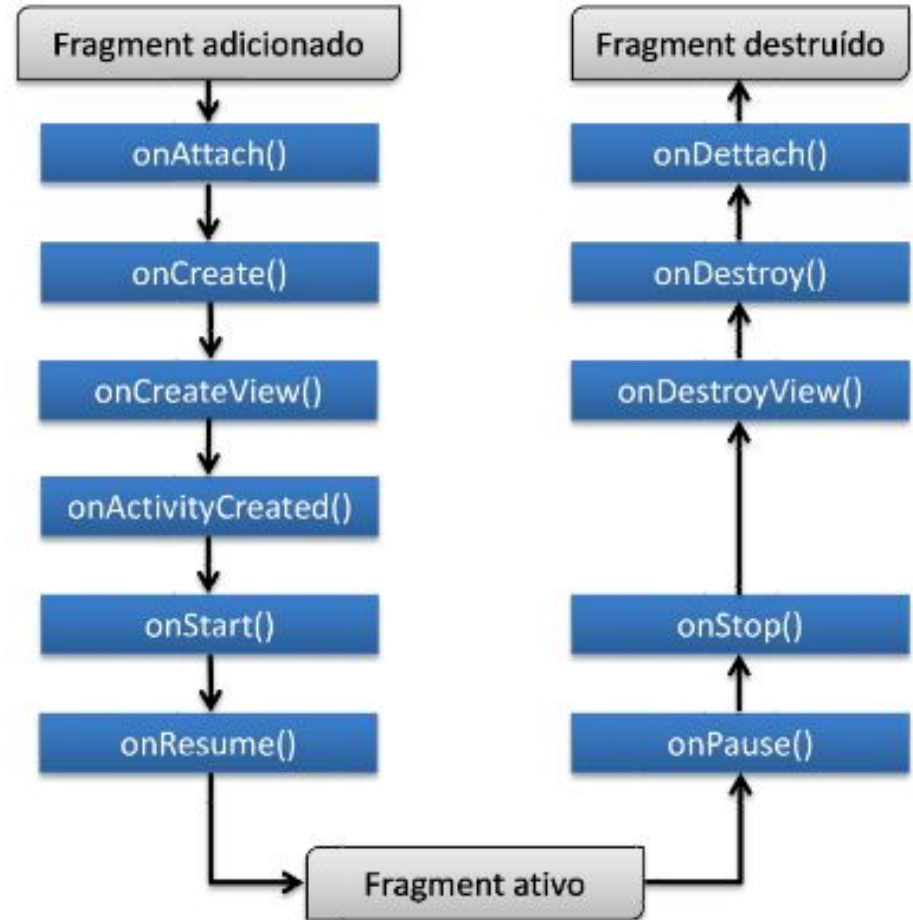
FRAGMENTS - CICLO DE VIDA

- ▶ Bastante semelhante ao de uma activity
- ▶ Atrelado ao ciclo de vida da activity associada



FRAGMENTS - CICLO DE VIDA

- ▶ Bastante semelhante ao de uma activity
- ▶ Atrelado ao ciclo de vida da activity associada



FRAGMENTS - ADICIONANDO

- ▶ Fragments podem ser adicionados a activities de duas formas
 - Estática
 - O fragment é declarado diretamente dentro do arquivo de layout da activity
 - Dinâmica
 - O fragment é adicionado via programação a partir de um ViewGroup existente

FRAGMENTS - ESTÁTICOS

activity_main.xml

```
3
4 <LinearLayout
5     xmlns:android="http://schemas.android.com/apk/res/android"
6     xmlns:tools="http://schemas.android.com/tools"
7     android:layout_width="match_parent"
8     android:layout_height="match_parent"
9     tools:context=".MainActivity">
10
11     <fragment
12         android:id="@+id/fragment1"
13         android:name="imd.android.MyFragment"
14         android:layout_width="0dp"
15         android:layout_height="match_parent"
16         android:layout_weight="1" />
17
18     <fragment
19         android:id="@+id/fragment2"
20         android:name="imd.android.MyFragment2"
21         android:layout_width="0dp"
22         android:layout_height="match_parent"
23         android:layout_weight="2" />
24
25 </LinearLayout>
```

FRAGMENTS - ESTÁTICOS

fragment1.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/
    android:layout_height="match_parent"
    android:layout_width="match_parent">

    <TextView android:layout_width="match_parent"
        android:text="Fragment 1"
        android:background="@color/colorAccent"
        android:gravity="center"
        android:textSize="26sp"
        android:layout_height="match_parent"/>

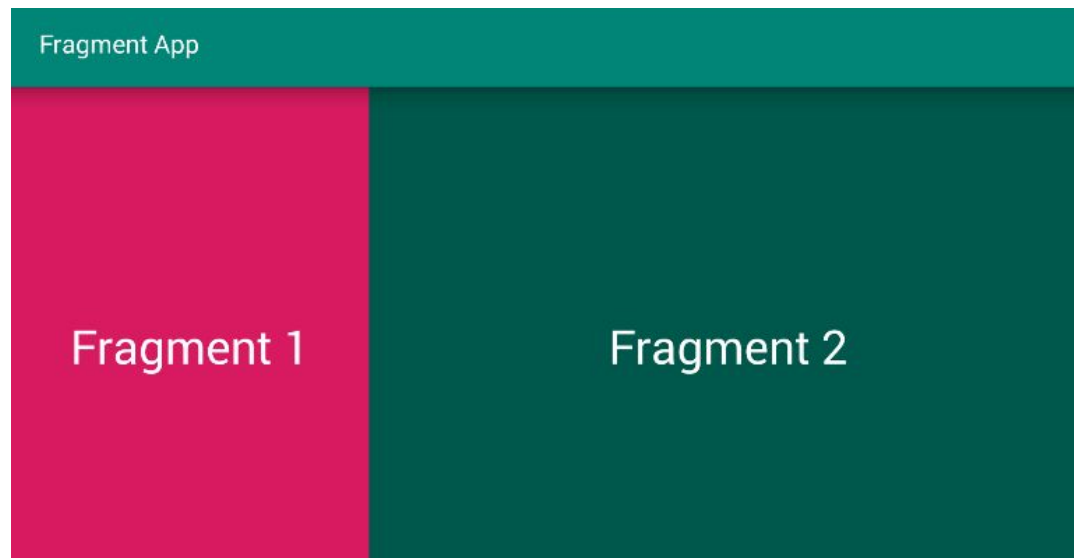
</LinearLayout>
```

fragment2.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/
    android:layout_height="match_parent"
    android:layout_width="match_parent">

    <TextView android:layout_width="match_parent"
        android:text="Fragment 2"
        android:background="@color/colorPrimary"
        android:gravity="center"
        android:textSize="26sp"
        android:layout_height="match_parent"/>

</LinearLayout>
```



FRAGMENTS - CLASSE FRAGMENT MANAGER

- ▶ Um objeto `FragmentManager` pode ser obtido a partir da `activity`

```
15  
16  
17  
  
val fm : FragmentManager = supportFragmentManager
```

- ▶ Pode ser usado para buscar um fragment associado à `activity`

```
19  
20  
21  
22  
  
val fragmentById : Fragment? = fm.findFragmentById(R.id.fragment1)
```

```
23  
24  
25  
  
val fragmentByTag : Fragment? = fm.findFragmentByTag("tag")
```

FRAGMENTS - DINÂMICOS

- ▶ Para gerenciar fragments via programação, é preciso usar **fragment transactions**
 - Uma fragment transaction agrupa um conjunto de alterações em fragments
- ▶ Inicia com **beginTransaction()**
- ▶ Chamadas aos métodos de gerenciamento de fragments
 - **add()**
 - Adiciona um fragment
 - **remove()**
 - Remove um fragment
 - **replace()**
 - Substitui um fragment
- ▶ Termina com **commit()**

FRAGMENTS - DINÂMICOS

activity_main.xml

```
3
4  <LinearLayout
5      xmlns:android="http://schemas.android.com/apk/res/android"
6      xmlns:tools="http://schemas.android.com/tools"
7      android:layout_width="match_parent"
8      android:layout_height="match_parent"
9      tools:context=".MainActivity"
10     android:baselineAligned="false">
11
12     <FrameLayout
13         android:id="@+id/lay_left"
14         android:layout_width="0dp"
15         android:layout_weight="1"
16         android:layout_height="match_parent">
17
18     </FrameLayout>
19
20     <FrameLayout
21         android:id="@+id/lay_right"
22         android:layout_width="0dp"
23         android:layout_weight="2"
24         android:layout_height="match_parent">
25
26     </FrameLayout>
27
28 </LinearLayout>
```

FRAGMENTS - DINÂMICOS

MainActivity.kt

```
9
10 class MainActivity : AppCompatActivity() {
11
12     override fun onCreate(savedInstanceState: Bundle?) {
13         super.onCreate(savedInstanceState)
14         setContentView(R.layout.activity_main)
15
16         val fm : FragmentManager = supportFragmentManager
17
18         val f1 : Fragment = MyFragment()
19         val f2 : Fragment = MyFragment2()
20
21         val ft : FragmentTransaction = fm.beginTransaction()
22         ft.add(R.id.lay_left, f1)
23         ft.add(R.id.lay_right, f2)
24         ft.commit()
25     }
26 }
27 }
```


FRAGMENTS - BACK STACK

- ▶ Uma transação pode ser adicionada a uma back stack de fragments, gerenciada pela activity
- ▶ Quando isso é feito, o botão Back (Voltar) do dispositivo faz com que a transação seja desfeita
- ▶ O método `addToBackStack()` é utilizado

```
20  
21  
22  
23  
24  
25  
26  
  
val ft : FragmentTransaction = fm.beginTransaction()  
ft.add(R.id.lay_left, f1)  
ft.add(R.id.lay_right, f2)  
ft.addToBackStack(null)  
ft.commit()
```

FRAGMENTS - SALVANDO ESTADO

- ▶ Assim como activities, fragments têm o método `onSaveInstanceState()`
- ▶ Permite armazenar o estado do fragment

```
8
9  <> class MyFragment : Fragment(){
10
11  ⬆  override fun onSaveInstanceState(outState: Bundle) {
12      super.onSaveInstanceState(outState)
13  }
14
15  }
16
```



FRAGMENTS - RESTAURANDO ESTADO

- Para restaurar o estado, é possível usar o bundle que contém os dados

```
14
15  override fun onCreate(savedInstanceState: Bundle?) {
16      super.onCreate(savedInstanceState)
17  }
18
19  override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?,
20                          savedInstanceState: Bundle?): View? {
21      return inflater.inflate(R.layout.fragment_layout, container, attachToRoot: false)
22  }
23
24  override fun onActivityCreated(savedInstanceState: Bundle?) {
25      super.onActivityCreated(savedInstanceState)
26  }
```

FRAGMENTS - EVITANDO DESTRUIÇÃO

- ▶ Mudanças de configuração provocam a destruição e recriação da activity
 - Mudar orientação, alterar idioma, abrir ou fechar teclado físico, etc.
- ▶ Quando a activity é destruída, seus fragments também são
- ▶ O uso de `retainInstance= true` evita isto



```
14  
15  override fun onCreate(savedInstanceState: Bundle?) {  
16      super.onCreate(savedInstanceState)  
17  
18      retainInstance= true  
19  }
```

Obrigado!
Dúvidas?

Professor Emerson Alencar
emerson@imd.ufrn.br