

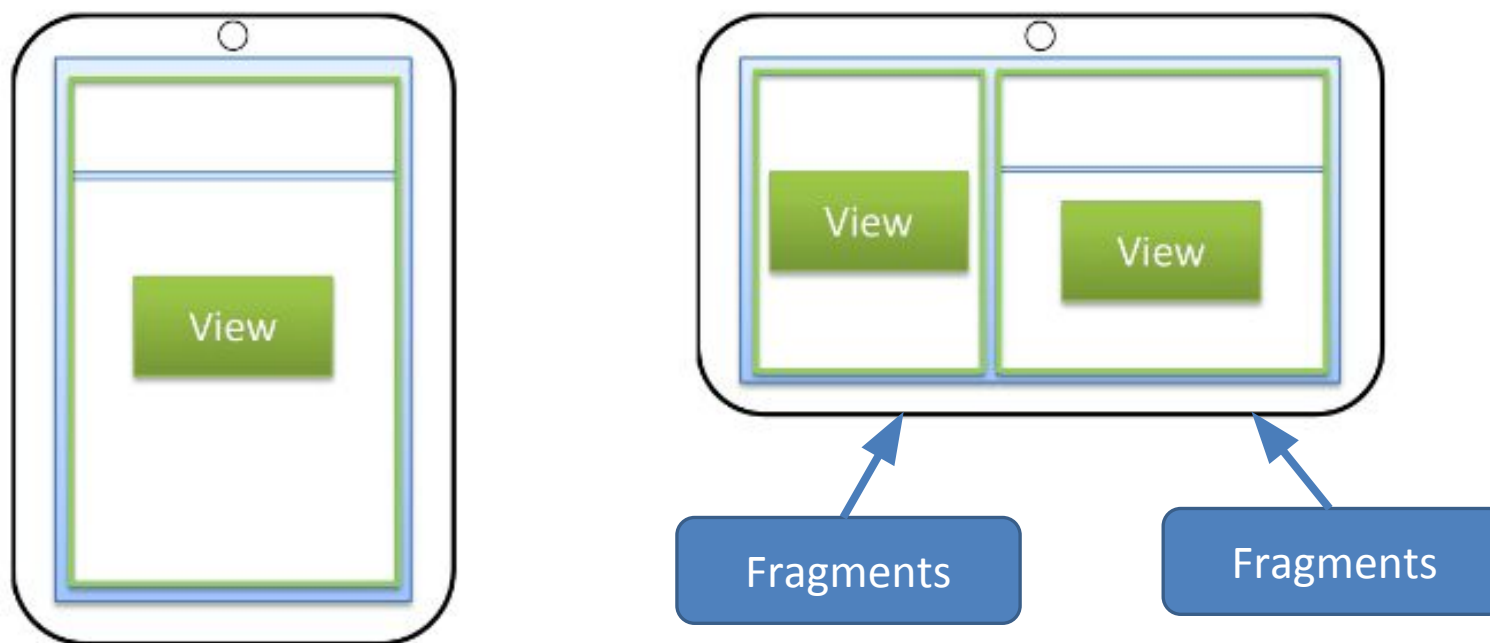
Desenvolvimento de aplicativos Android com Fragments

Prof. Emerson Moura de Alencar

Fragment

Surgimento dos Fragments

- 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



Fragment

- 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

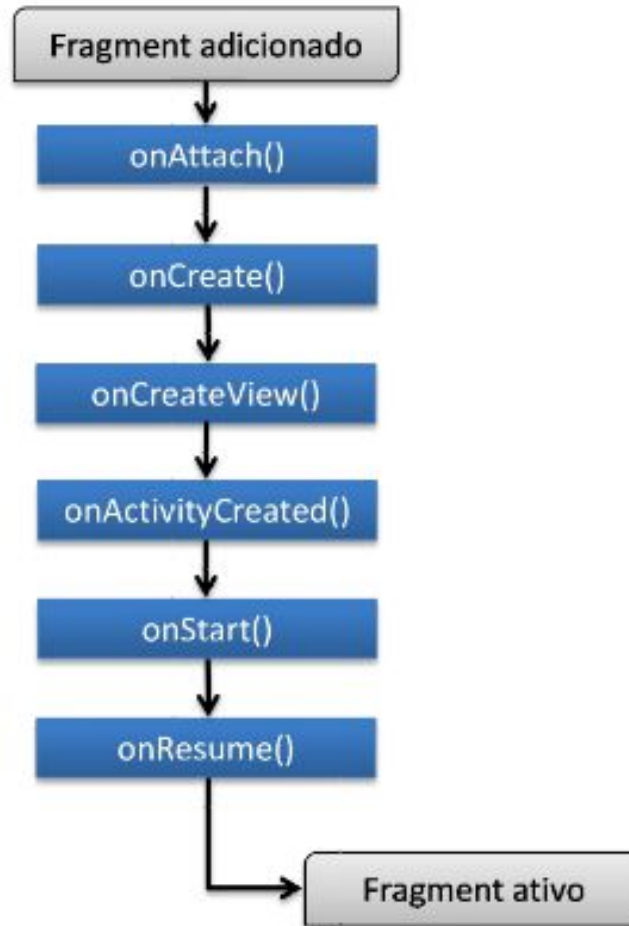
Criando Fragments

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

```
public class MyFragment extends Fragment {  
  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        //...  
    }  
  
    public void onPause() {  
        super.onPause();  
        //...  
    }  
  
    public View onCreateView(LayoutInflater inflater, ViewGroup container,  
        Bundle savedInstanceState) {  
        return inflater.inflate(R.layout.fragment_layout, container, false);  
    }  
}
```

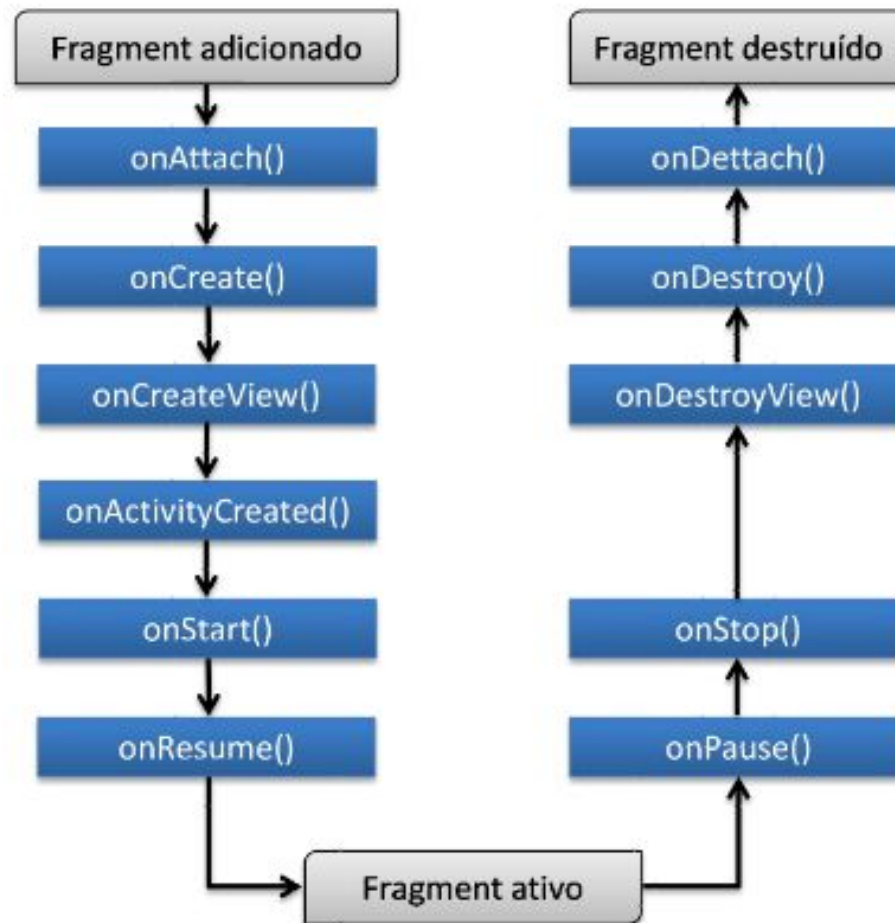
Ciclo de Vida de um Fragment

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



Ciclo de Vida de um Fragment

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



Adicionando Fragments

- 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.xml

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">

    <fragment
        android:id="@+id/fragment1"
        android:name="imd.android.MyFragment1"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1" />

    <fragment
        android:id="@+id/fragment2"
        android:name="imd.android.MyFragment2"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="2" />

</LinearLayout>
```

Fragments Estáticos

fragment1.xml

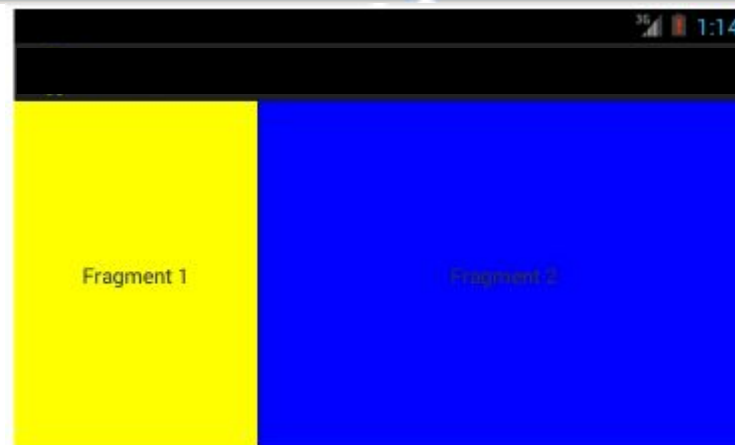
```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#FFFF00"
        android:gravity="center"
        android:text="@string/txt_frag1" />
</LinearLayout>
```

fragment2.xml

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#0000FF"
        android:gravity="center"
        android:text="@string/txt_frag2" />
</LinearLayout>
```



A Classe FragmentManager

- Um objeto **FragmentManager** pode ser obtido a partir da activity

```
FragmentManager fm = getFragmentManager();
```

- Pode ser usado para buscar um fragment associado à activity

```
Fragment f = fm.findFragmentById(R.id.fragment1);
```

```
Fragment f = 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.xml

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:baselineAligned="false">

    <FrameLayout
        android:id="@+id/lay_left"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1" />

    <FrameLayout
        android:id="@+id/lay_right"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="2" />

</LinearLayout>
```

Placeholder



Fragments Dinâmicos

MyActivity.java

```
public class MainActivity extends Activity {  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity);  
  
        FragmentManager fm = getFragmentManager();  
  
        MyFragment1 f1 = new MyFragment1();  
        MyFragment2 f2 = new MyFragment2();  
  
        FragmentTransaction ft = fm.beginTransaction();  
        ft.add(R.id.lay_left, f1);  
        ft.add(R.id.lay_right, f2);  
        ft.commit();  
    }  
}
```


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

```
FragmentTransaction ft = fm.beginTransaction();  
ft.add(R.id.lay_left, f1);  
ft.add(R.id.lay_right, f2);  
ft.addToBackStack(null);  
ft.commit();
```

Salvando o Estado de um Fragment

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

```
public class MyFragment extends Fragment {  
  
    protected void onSaveInstanceState(Bundle outState) {  
        //...  
        super.onSaveInstanceState(outState);  
    }  
}
```

Restaurando o Estado de um Fragment

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

```
public void onCreate(Bundle savedInstanceState) {  
    //...  
}  
  
public View onCreateView(LayoutInflater inflater,  
    ViewGroup container, Bundle savedInstanceState) {  
    //...  
}  
  
public void onActivityCreated(Bundle savedInstanceState) {  
    //...  
}
```

Evitando a Destruição do Fragment

- 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 método **setRetainInstance()** evita isto

```
public void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setRetainInstance(true);  
}
```

Só pode ser usado por fragments
que não estão na back stack

Usando a API de Compatibilidade

- Fragments é um recurso adicionado a partir do Android Honeycomb (API Level 11)
- Para que versões anteriores tenham acesso ao uso de fragments, a API de compatibilidade deve ser utilizada
- É preciso fazer algumas adaptações no código para que tudo funcione

Usando a API de Compatibilidade

- A activity que usa fragments deve herdar de **FragmentActivity**

```
public class MainActivity extends FragmentActivity { }
```

- Um objeto **FragmentManager** deve ser obtido através de **getSupportFragmentManager()**

```
FragmentManager fm = getSupportFragmentManager() ;
```

- Algumas classes estão em outros pacotes

Classes na API padrão	Classes na API de compatibilidade
android.app.FragmentManager	android.support.v4.app.FragmentManager
android.app.FragmentTransaction	android.support.v4.app.FragmentTransaction
android.app.Fragment	android.support.v4.app.Fragment

Alguma dúvida?

