



**INSTITUTO FED. DE EDUCAÇÃO, CIÊNC. E TEC. DE PERNAMBUCO**  
**CURSO:** TEC. EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS  
**DISCIPLINA:** PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS  
**PROFESSOR:** RAMIDE DANTAS  
**ASSUNTO:** TRABALHANDO COM FRAGMENTOS

Aluno (a):			
Matrícula:		Data:	

## Aula Prática 02

Adaptado de: <http://www.vogella.com/tutorials/AndroidFragments/article.html>

### Parte 1: Criando fragmentos

Passo 1: Crie uma novo projeto contendo um atividade em branco com as seguintes propriedades:

**Application Name:** Pratica02  
**Package name:** br.edu.ifpe.tads.pdm.pratica02  
**Template:** Empty Activity  
**Activity:** MainActivity  
**Layout:** activity\_main

Passo 2: Crie um novo fragmento (*app > new > Fragment > Fragment (Blank)*):

**Fragment Name:** DetailFragment  
**Fragment Layout Name:** fragment\_detail

Passo 3: Modifique **DetailFragment.java**, adicionando o método `setText()` e modificando o método `onAttach()` como mostrado abaixo:

```
public class DetailFragment extends Fragment {  
  
    ...  
  
    public void setText(String item) {  
        TextView view = (TextView)  
            getView().findViewById(R.id.detailsText);  
        view.setText(item);  
    }  
  
    @Override  
    public void onAttach(Context context) {  
        super.onAttach(context);  
        if (context instanceof OnFragmentInteractionListener) {  
            mListener = (OnFragmentInteractionListener) context;  
        } else {  
            mListener = null;  
        }  
    }  
}
```

Passo 4: Modifique **fragment\_detail.xml**, colocando o código abaixo:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/detailsText"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="center horizontal|center vertical"
        android:layout_marginTop="20dip"
        android:text="Default Text"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textSize="30sp" />
</LinearLayout>
```

Passo 5: Crie um novo fragmento (*app > new > Fragment > Fragment (Blank)*):

**Fragment Name:** OverviewFragment

**Fragment Layout Name:** fragment\_overview

Passo 6: Modifique a classe em **OverviewFragment.java** para refletir o código abaixo:

```
public class OverviewFragment extends Fragment {
    ...
    @Override // Substituir método
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
        Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_overview, container, false);

        Button button = (Button) view.findViewById(R.id.button1);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                updateDetail();
            }
        });
        return view;
    }
    ...

    @Override // Substituir método
    public void onAttach(Context context) {
        super.onAttach(context);
        if (context instanceof OnFragmentInteractionListener) {
            mListener = (OnFragmentInteractionListener) context;
        } else {
            throw new ClassCastException(context.toString()
                + " must implement OverviewFragment.OnItemSelectedListener");
        }
    }
    ...

    // Mudar o parâmetro de Uri para para String msg
    public interface OnFragmentInteractionListener {
        void onFragmentInteraction(String msg);
    }
    ...

    public void updateDetail() {
        String newTime = String.valueOf(System.currentTimeMillis());
        mListener.onFragmentInteraction(newTime);
    }
}
```

O código de `OverviewFragment` define uma interface `OnFragmentInteractionListener`, que deve ser implementada pela atividade que contém esse fragmento (do contrário será lançada uma exceção). Essa interface é usada para notificar sempre que um item for selecionado na lista (nesse caso, quando o botão for acionado). O acionamento do botão apenas gera um string contendo o tempo em milissegundos, que é passado para o listener – no caso, a atividade – que decidirá o que fazer com esse valor.

Passo 7: Modifique **fragment\_overview.xml**, colocando o código abaixo:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Press to update" />

</LinearLayout>
```

## Parte 2: Usando os fragmentos na atividade principal

Passo 1: No layout da atividade, **activity\_main.xml**, colocar o código abaixo:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:baselineAligned="false"
    android:orientation="horizontal" >

    <fragment
        android:id="@+id/listFragment"
        android:layout_width="0dp"
        android:layout_weight="1"
        android:layout_height="match_parent"
        class="br.edu.ifpe.tads.pdm.pratica02.OverviewFragment" />

    <fragment
        android:id="@+id/detailFragment"
        android:layout_width="0dp"
        android:layout_weight="2"
        android:layout_height="match_parent"
        class="br.edu.ifpe.tads.pdm.pratica02.DetailFragment" />

</LinearLayout>
```

O código acima instancia os fragmentos criados na parte 1, associando-os a atividade principal da aplicação.

Passo 2: Modifique **MainActivity.java**, colocando o código abaixo:

```
import ...;

public class MainActivity extends AppCompatActivity
    implements OverviewFragment.OnFragmentInteractionListener {

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

    @Override
    public void onFragmentInteraction(String link) {
        DetailFragment fragment = (DetailFragment) getSupportFragmentManager()
            .findFragmentById(R.id.detailFragment);
        if (fragment != null && fragment.isInLayout()) {
            fragment.setText(link);
        }
    }
}
```

Veja que a atividade principal implementa a interface `OnFragmentInteractionListener` da classe `OverviewFragment`. Na implementação do método `onFragmentInteraction()`, o valor recebido é setado no fragmento de detalhe (`DetailFragment`) que faz parte da atividade.

Passo 3: Teste a aplicação. Veja que ambos os fragmentos são exibidos um ao lado do outro, não importando a orientação do dispositivo. (Se estiver usando o emulador, use `Ctrl+F12` para mudar a orientação.)



### Parte 3: Adequando os fragmentos a orientação do dispositivo

Passo 1: Crie um layout alternativo para a atividade principal quando o dispositivo estiver na posição retrato (portrait)

Vá em **res/layout/activity\_main.xml**, clique *new > Layout resource file*, dê o mesmo nome “activity\_main.xml”. Em “Available qualifiers” escolha “orientation”, clique >>, e “portrait” no lado direito.

Passo 2: No arquivo **res/layout/activity\_main.xml/activity\_main.xml (port)**, coloque o código abaixo:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal" >

    <fragment
        android:id="@+id/listFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="?android:attr/actionBarSize"
        class="br.edu.ifpe.tads.pdm.pratica02.OverviewFragment" />
</LinearLayout>
```

Esse novo arquivo de layout será carregado com o dispositivo na posição retrato (Portrait). Nele apenas um fragmento é usado: o que mostra o overview dos items.

Passo 3: Crie uma nova atividade chamada `DetailActivity`, que exibirá o conteúdo quando o dispositivo estiver na posição retrato.

**Activity:**            `DetailActivity`  
**Layout:**            `activity_detail`

Passo 4: Modifique **`DetailActivity.java`** para conter o código abaixo:

```
public class DetailActivity extends AppCompatActivity {
    public static final String EXTRA_URL = "url";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        if (getResources().getConfiguration().orientation ==
            Configuration.ORIENTATION_LANDSCAPE) {
            finish();
            return;
        }
        setContentView(R.layout.activity_detail);
        Bundle extras = getIntent().getExtras();
        if (extras != null) {
            String url = extras.getString(EXTRA_URL);
            DetailFragment detailFragment = (DetailFragment)
                getSupportFragmentManager().findFragmentById(R.id.detailFragment);
            detailFragment.setText(url);
        }
    }
}
```

Passo 5: Modifique o layout **activity\_detail.xml** para conter o código abaixo:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <fragment
        android:id="@+id/detailFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        class="br.edu.ifpe.tads.pdm.pratica02.DetailFragment" />

</LinearLayout>
```

A atividade `DetailActivity` é acionada quando o usuário clica no botão do fragmento `OverviewFragment` e o dispositivo está na posição retrato. Ela contém o fragmento `DetailFragment` que havíamos criado antes e que é usado na atividade principal quando o dispositivo está na posição paisagem.

Passo 6: Modifique o método `onFragmentInteraction()` da classe `MainActivity`, colocando o seguinte código:

```
@Override
public void onFragmentInteraction(String link) {
    DetailFragment fragment = (DetailFragment) getSupportFragmentManager()
        .findFragmentById(R.id.detailFragment);
    if (fragment != null && fragment.isVisible()) {
        fragment.setText(link);
    } else {
        Intent intent = new Intent(getApplicationContext(),
            DetailActivity.class);
        intent.putExtra(DetailActivity.EXTRA_URL, link);
        startActivity(intent);
    }
}
```

O novo tratador verifica se existe um fragmento do tipo `DetailFragment` na atividade (o que indica que o dispositivo está orientado horizontalmente [paisagem]), e sendo o caso procede como anteriormente. Caso contrário (dispositivo na vertical/retrato), lança um `Intent` que dispara a atividade `DetailActivity` criada nos passos anteriores.

Passo 7: Teste a aplicação. Veja como ela se comporta quando colocamos o dispositivo na posição retrato e paisagem.

## Parte 4 (Desafio): Trabalhando com FragmentManager

**Passo 1:** Retire os fragmentos (`<fragment>`) dos arquivos de layout (.xml) da atividade principal (o default e o criado para orientação retrato [portrait]).

Substitua as tags `<fragment>` por `<LinearLayout>` com nomes apropriados em ambos os arquivos. Esses layouts serão usados como pontos de inserção dos fragmentos no próximo passo.

**Passo 2:** No `onCreate()` de `MainActivity`, adicione código usando `FragmentManager` e `FragmentTransaction` para criar e adicionar os fragmentos `DetailFragment` e `OverviewFragment` nos lugares apropriados.

Para obter o `FragmentManager`, use

```
getSupportFragmentManager()
```

Para criar um `FragmentTransaction`, use

```
manager.beginTransaction()
```

Para criar e inserir um fragmento, use

```
transac.add([layout],[frag_class].newInstance(),"tag_frag")
```

`[layout]` é o Id do layout onde o fragmento será inserido (passo 1);  
`[frag_class].newInstance()` é um método estático para instanciar o fragmento, deve existir na classe do fragmento;  
`"tag_frag"` é um nome usado para identificar o fragmento.

Para comitar a transação, use

```
transac.commit()
```

**Passo 3:** Ao modificar o `onCreate()`, certifique-se de que o dispositivo está na orientação certa. Para isso veja se o layout onde você deve inserir o fragmento está presente ou não antes de inserir o fragmento (use `findViewById()`).

Certifique-se também de não estar recriando o fragmento desnecessariamente. O Android preserva fragmentos entre recriações da atividade. Use `manager.getFragmentByTag()` para verificar isso.

**Passo 4:** Modifique o método `onFragmentInteraction()` na `MainActivity` de forma a obter o fragmento usando `manager.getFragmentByTag()`, ao invés de usar `findFragmentById()`.