Lecture 5

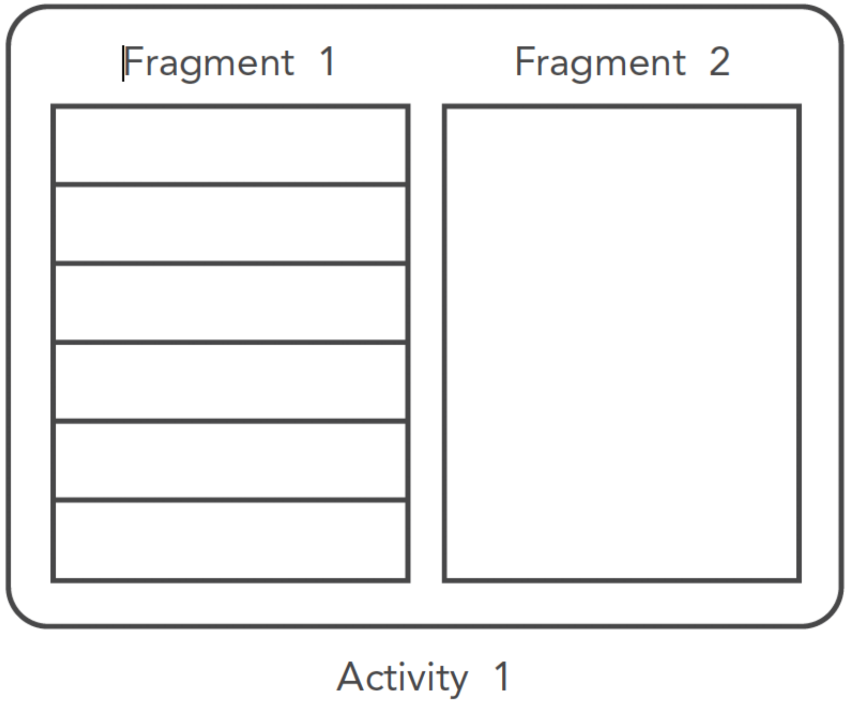
Fragments

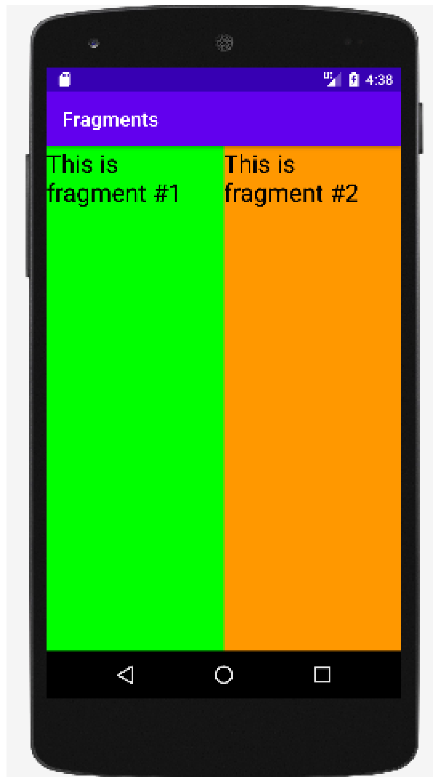
Course: **Mobile App Development**

By: Tho C. Mai
Nha Trang University

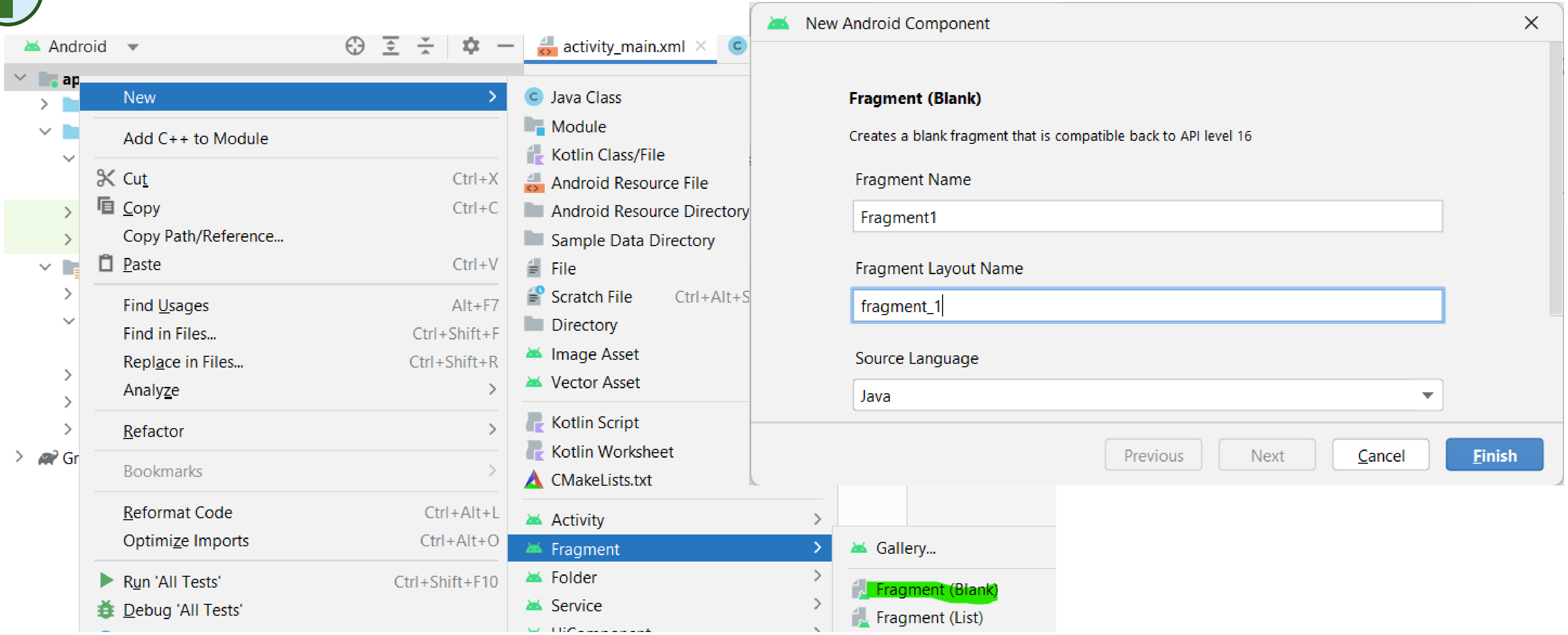
Fragments

- Activity is a container for views => typically fills the entire screen
- Fragments are introduced for large screen devices
  - One activity contains several mini-activities (fragments)





Using Fragments



Fragment (Blank)

Creates a blank fragment that is compatible back to API level 16

Fragment Name: Fragment1

Fragment Layout Name: fragment_1

Source Language: Java

Using Fragments

- Create a new **Fragment,** and name it **Fragment1**
- In the res/layout folder, we have **fragment_1.xml**. Populate it with the following code

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:orientation="vertical"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:background="#00FF00"
   tools:context=".Fragment1">
   <TextView
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:text="This is fragment #1"
      android:textColor="#000000"
      android:textSize="25sp" />
</LinearLayout>
```

Using Fragments

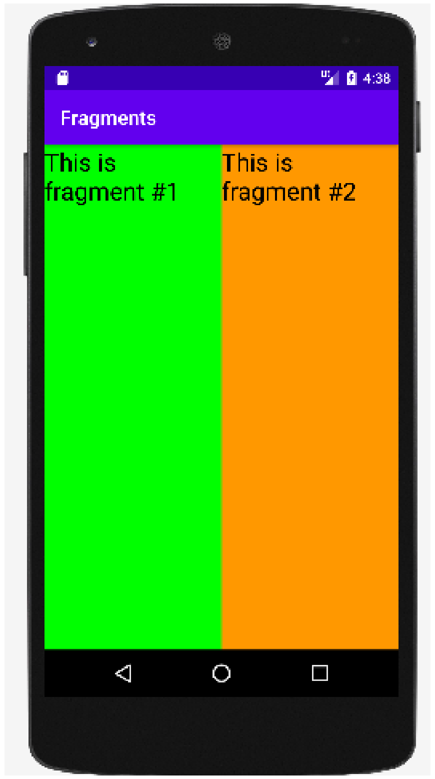
- Also Create a new **Fragment,** and name it **Fragment2**
- in the res/layout folder, we have **fragment_2.xml.** Populate it as follows

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:orientation="vertical"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:background="#00FF00"
   tools:context=".Fragment1">
   <TextView
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:text="This is fragment #2"
      android:textColor="#000000"
      android:textSize="25sp" />
</LinearLayout>
```

Using Fragments

- In activity_main.xml, replace all with in the following code:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent“
   android:layout_height="match_parent“      tools:context=".MainActivity">
   <fragment
      android:name="com.maicuongtho.fragments.Fragment1"
      android:id="@+id/fragment1"
      android:layout_weight="1"
      android:layout_width="fill_parent"
      android:layout_height="match_parent" />
   <fragment
      android:name="com.maicuongtho.fragments.Fragment2"
      android:id="@+id/fragment2"
      android:layout_weight="1"
      android:layout_width="fill_parent"
      android:layout_height="match_parent" />
</LinearLayout>
```



This is fragment #1

This is fragment #2

Adding Fragments Dynamically

- In the same project, modify the activity_main.xml file by commenting out the two <fragment> elements

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent“
   android:layout_height="match_parent“
   tools:context=".MainActivity“
    android:orientation="vertical“  >
<!-- <fragment-->
…
<!--    <fragment-->
…
</LinearLayout>
```

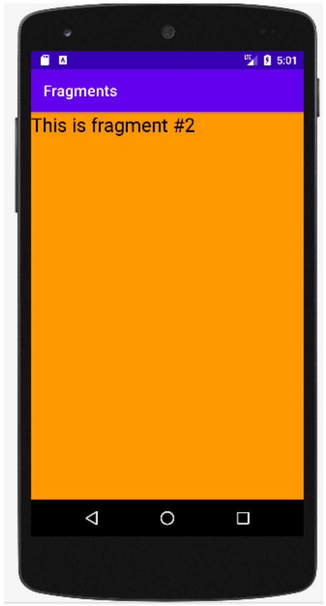
Adding Fragments Dynamics

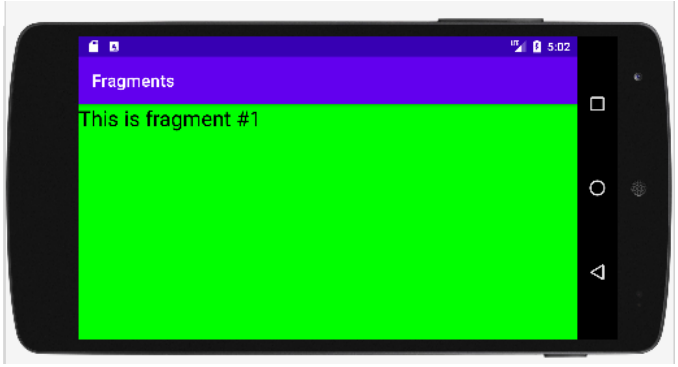
- Add lines in the following code to the MainActivity.java file

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
//      setContentView(R.layout.activity_main);
    FragmentManager fragmentManager = getSupportFragmentManager();
    FragmentTransaction fragmentTransaction =  fragmentManager.beginTransaction();
    //---get the current display info---
    DisplayMetrics display = this.getResources().getDisplayMetrics();
    int width = display.widthPixels;        int height = display.heightPixels;
    if (width> height) {   //---landscape mode---
        Fragment1 fragment1 = new Fragment1();
        // android.R.id.content refers to the content view of the activity
        fragmentTransaction.replace(android.R.id.content, fragment1);
    }
    else  {          //---portrait mode---
        Fragment2 fragment2 = new Fragment2();
        fragmentTransaction.replace(android.R.id.content, fragment2);
    }
    fragmentTransaction.commit();
  }
}
```

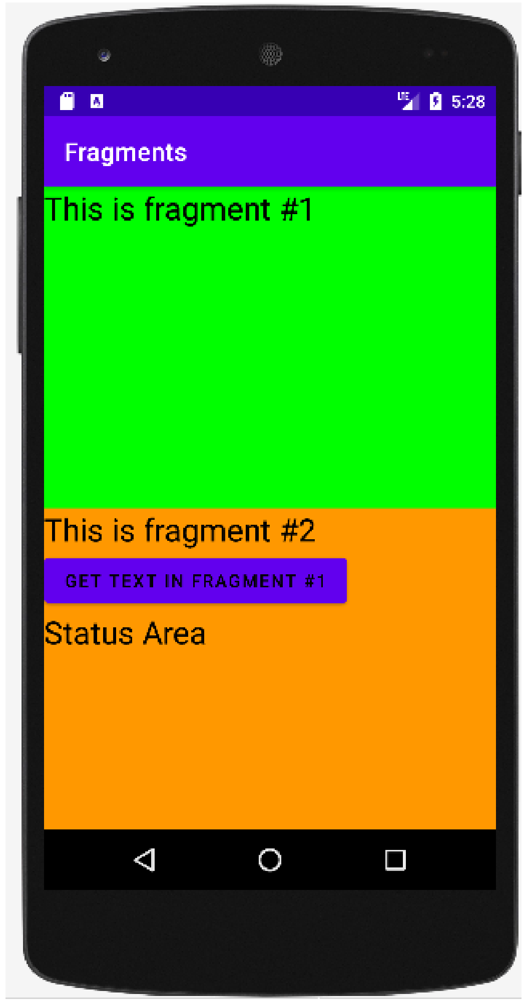
Remove: setContentView(R.layout.*activity_main*);

FragmentTransaction class to perform fragment transactions (such as add, remove, or replace) in your activity





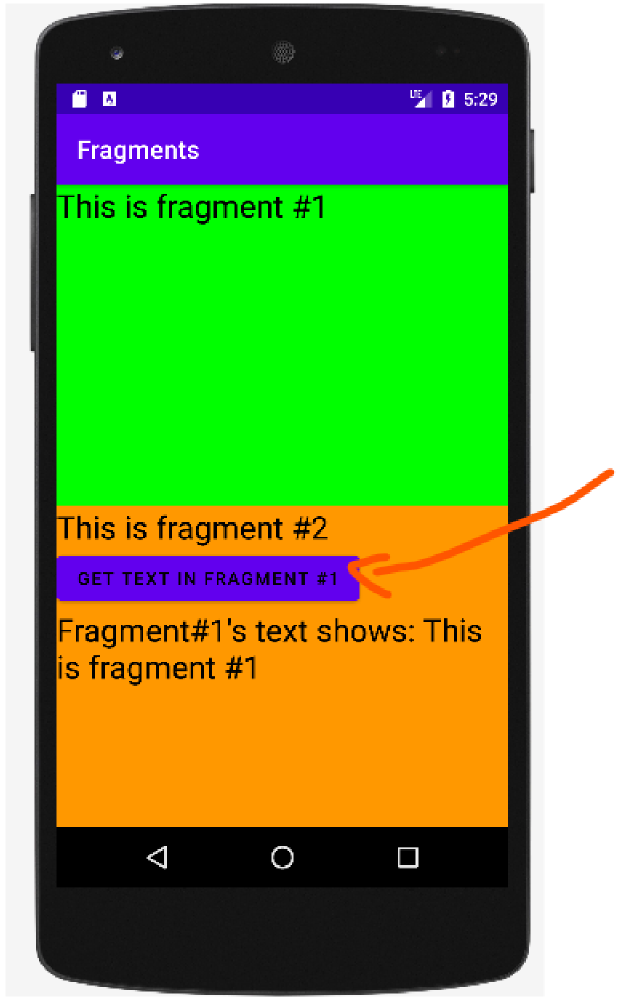
Interactions between Fragments



This is fragment #1

This is fragment #2

Status Area



This is fragment #1

This is fragment #2

Fragment#1's text shows: This is fragment #1

Interactions between Fragments

- An activity might contain two or more fragments working together to present a coherent UI to the user
  - E.g.: the user taps on an item in that fragment, details about the selected item might be displayed in another fragment
- Continue in the same project, add bolded statements to **fragment_1.xml**

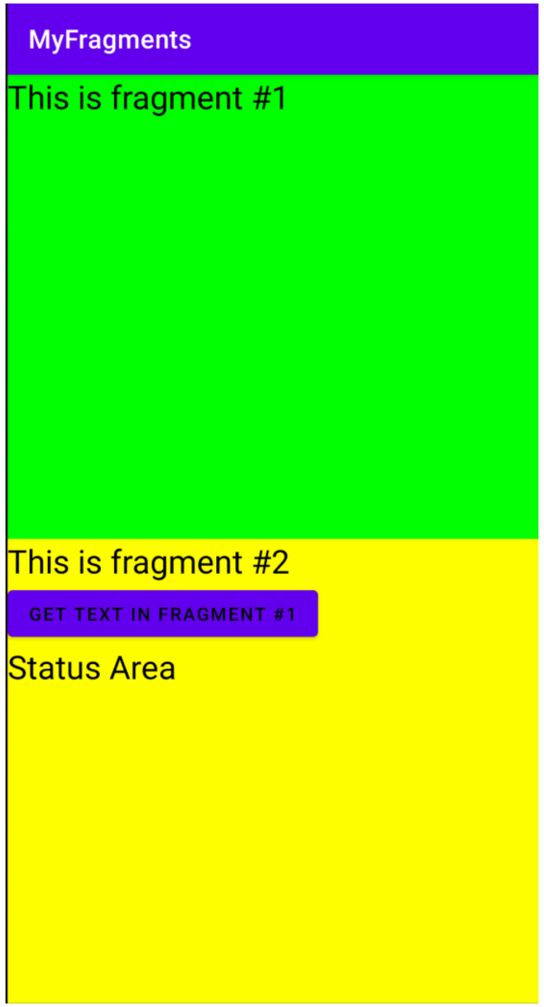
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:orientation="vertical"    android:layout_width="fill_parent"
   android:layout_height="fill_parent“      android:background="#00FF00"
   tools:context=".Fragment1">
   <TextView
      android:id="@+id/lblFragment1"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:text="This is fragment #1"
      android:textColor="#000000"
      android:textSize="25sp" />
</LinearLayout>
```

Interactions between Fragments

- Add the following bolded lines to fragment_2.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout

   ........
   <Button
      android:id="@+id/btnGetText"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Get text in Fragment #1"
      android:textColor="#000000"
      android:onClick="onClick" />
   <TextView
      android:id="@+id/lblStatus"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:text="Status Area"
      android:textColor="#000000"
      android:textSize="25sp" />
</LinearLayout>
```



Interactions between Fragments

- In activity_main.xml, uncomment the two fragments:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent“       android:layout_height="match_parent"
   tools:context=".MainActivity“             android:orientation="vertical">
   <fragment
      android:name="com.maicuongtho.fragments.Fragment1"
      android:id="@+id/fragment1"
      android:layout_weight="1"
      android:layout_width="fill_parent"
      android:layout_height="match_parent" />
   <fragment
      android:name="com.maicuongtho.fragments.Fragment2"
      android:id="@+id/fragment2"
      android:layout_weight="1"
      android:layout_width="fill_parent"
      android:layout_height="match_parent" />
</LinearLayout>
```

Interactions between Fragments

- Modify the MainActivity.java file by commenting out the code added in the previous step, and **add setContentView() back**

```
package com.maicuongtho.fragments;
……
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
//        FragmentManager fragmentManager = getSupportFragmentManager();
//………
//        fragmentTransaction.commit();
    }
 }
```

Interactions between Fragments

- Add the bolded statements to the Fragment2.java

Fagments2.java

```
…
public class Fragment2 extends Fragment {
…
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup containe
         return inflater.inflate(R.layout.fragment_2, container, false);
   }
   @Override
   public void onStart() {
      super.onStart();
       Button btnGetText = (Button)getActivity().findViewById(R.id.btnGetText); //---Button view---
       btnGetText.setOnClickListener(new View.OnClickListener() {        //---Button click--
         public void onClick(View v) {
            TextView lbl = (TextView) getActivity().findViewById(R.id.lblFragment1);
            TextView statusarea = (TextView) getActivity().findViewById(R.id.lblStatus);
            statusarea.setText("Fragment#1's text shows: "+lbl.getText());
         }
      });
   }
}
```

use a LayoutInflater object to inflate the UI from the specified XML file

Your turn

I fear not the man who has practiced 10,000 kicks once,
but I fear the man who has practiced one kick 10,000 times.
Bruce Lee

- Practice 7,8,9/ Excercie 7,8,9
  1) Repeat the examples by yourself (new project for each case)
  2) Push it to your github repository
     - ✓ With a report with screenshots of the final app in action, data structures used/class design, and the implementation logic.
- Practice 10/ Excercie 10 (Homework #4)

Homework #4

- Notepad app with custom keypad
  - Top fragment: display the note entered
  - Bottom fragment: display several rows of letters, numbers, and symbols (each as a button) for text input; when a user pushes a button, the corresponding input is appended in the top fragment

What to submit:

- Push it to your github repository,
  - a report with screenshots of the final app in action, data structures used/class design, and the implementation logic.