Mobile Applications for Business

Master SIA/SDBIS

Octavian Dospinescu

2021

General topics

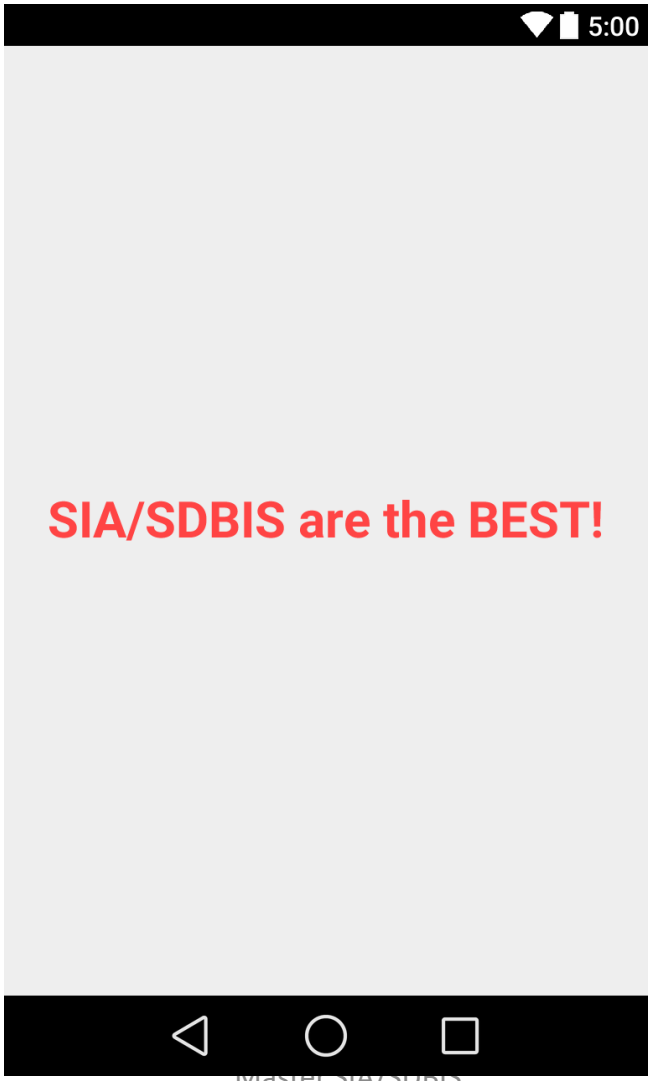
- Activity
- Activity's lifecycle
- Activity's Instance State and screen rotation
- Starting new activities
- Finishing an activity
- Fragments
- Fragment's lifecycle
- Setting listeners in a programmatic way

General topics

- Activity
- Activity's lifecycle
- Activity's Instance State and screen rotation
- Starting new activities
- Finishing an activity
- Fragments
- Fragment's lifecycle
- Setting listeners in a programmatic way

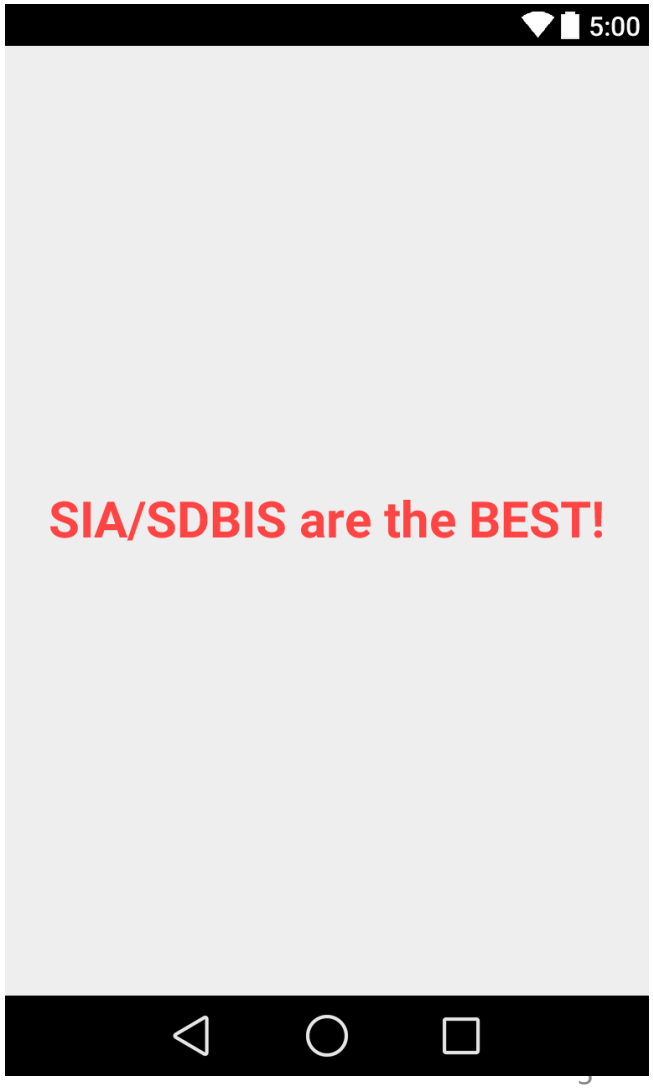
Activity – the visual component of an Android application

(http://developer.android.com/reference/android/app/Activity.html)



Activity – how does it work? ☺

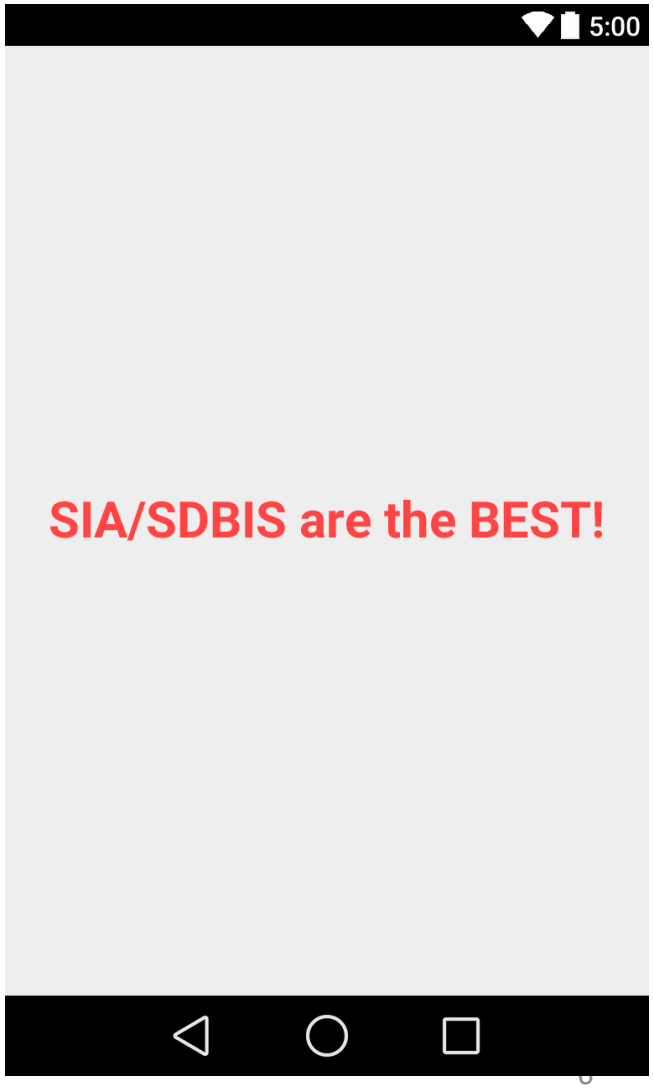
- Define a **layout** in xml
- Define an **Activity class**
- Implement the **onCreate**() method in the Activity and *setContentView* as being the layout previously defined.



Activity – how does it works? ☺

- Define a **layout** (activity_frm_principal.xml)

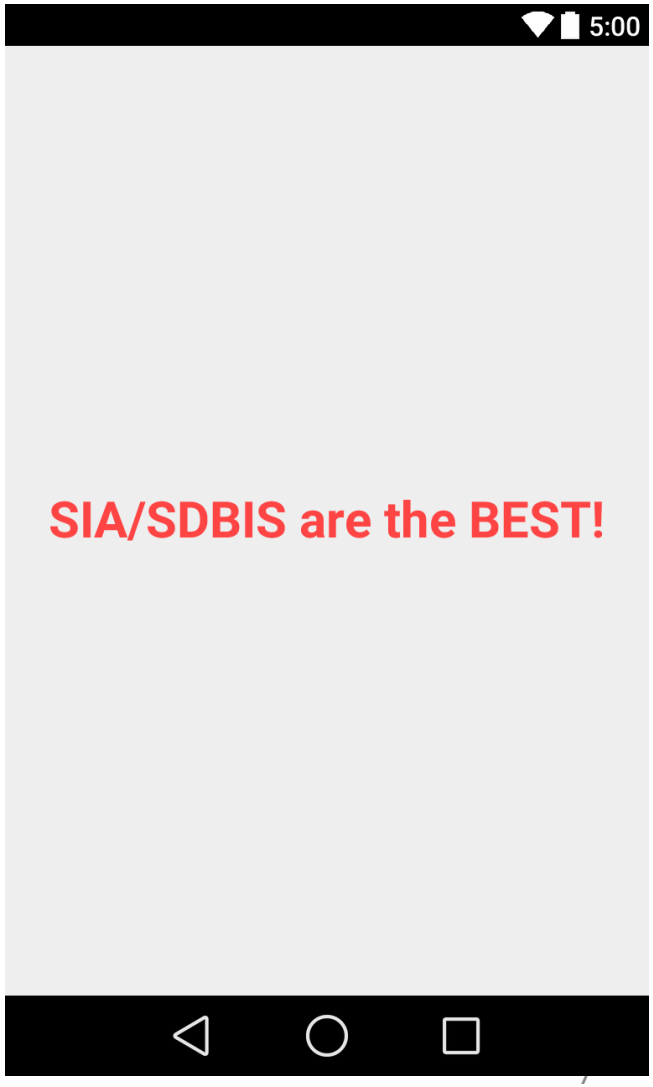
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".FrmPrincipal">

    <TextView android:text="SIA/SDBIS are the BEST!"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:textSize="30dp"
        android:textStyle="bold"
        android:textColor="@android:color/holo_red_light"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```



Activity – how does it works? ☺

- Define an **Activity class**

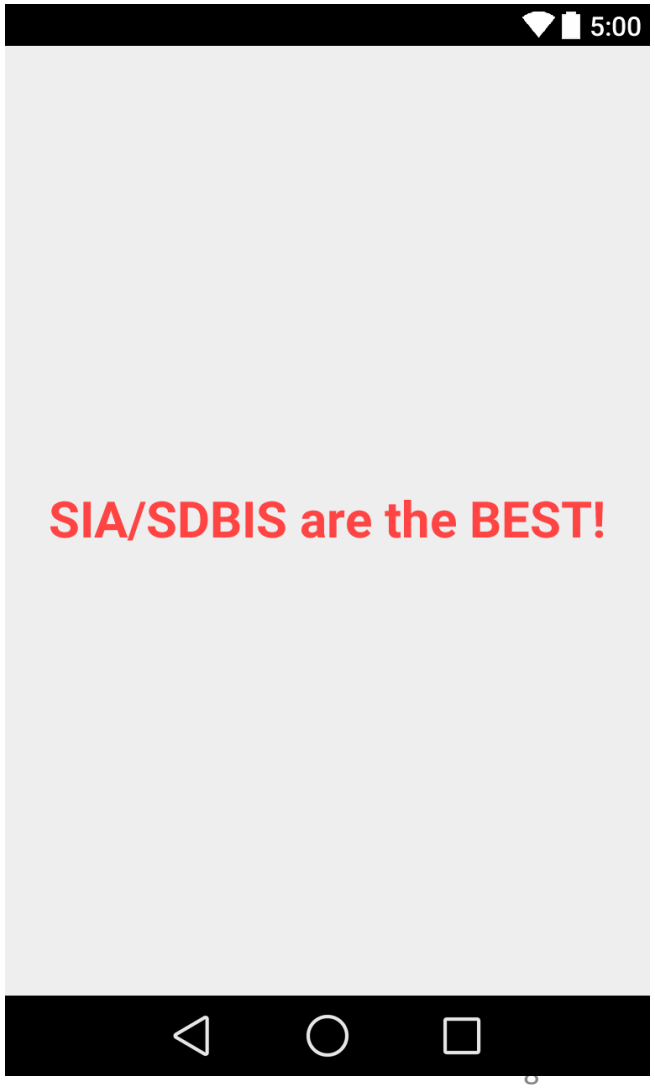
```
package com.example.adminlocal.appcurs04;

import android.app.Activity;
import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;


public class FrmPrincipal extends Activity {

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



Activity – how does it works? ☺

Implement the **onCreate**() method in the Activity and setContentView as being the layout previously defined.

```
public class FrmPrincipal extends Activity {

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

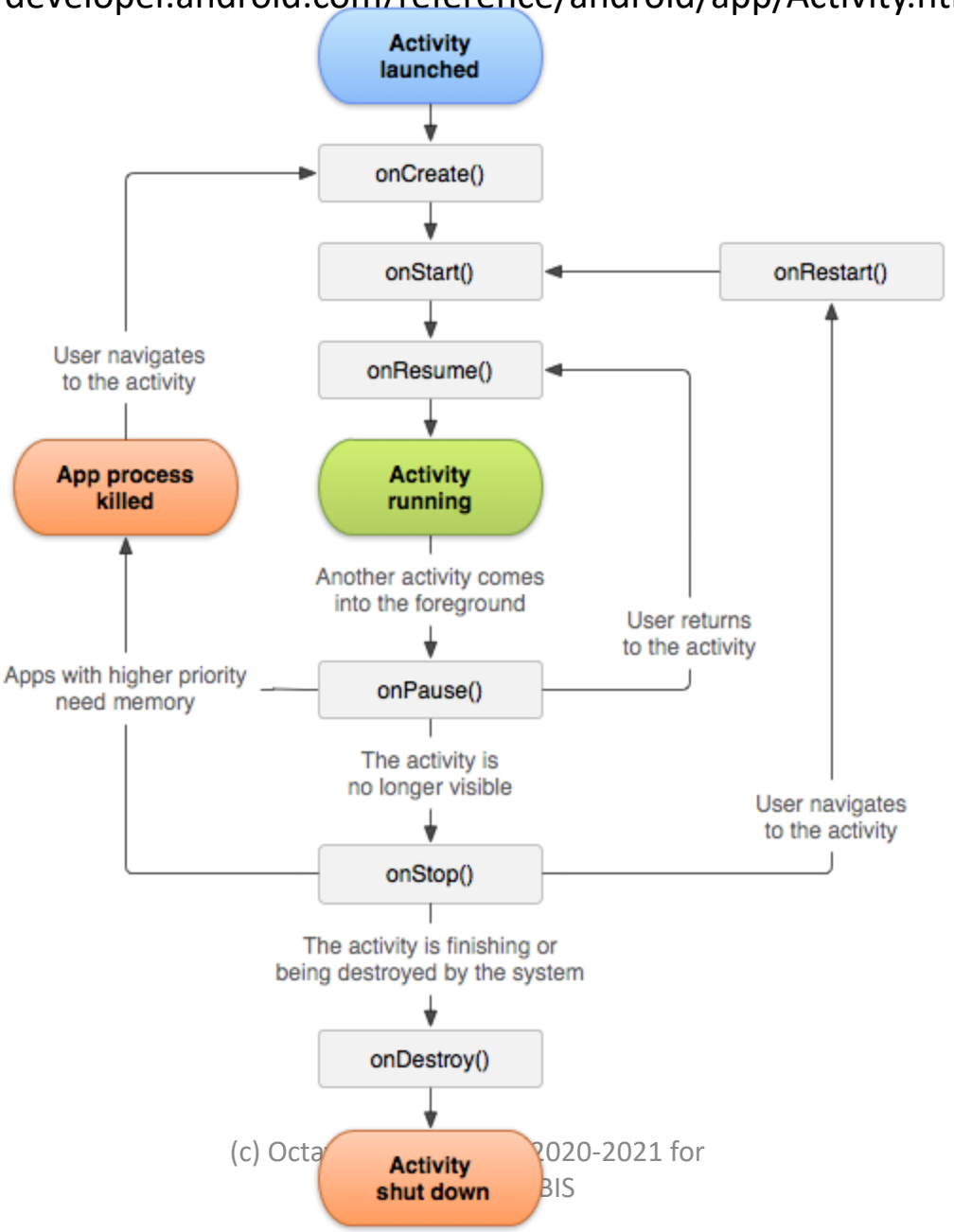


General topics

- Activity
- Activity's lifecycle
- Activity's Instance State and screen rotation
- Starting new activities
- Finishing an activity
- Fragments
- Fragment's lifecycle
- Setting listeners in a programmatic way

It’s very simple ☺

according to http://developer.android.com/reference/android/app/Activity.html



Activity's states

- **Running**: the activity is in the foreground of the screen;
- **Paused**: the activity is still visible, but it has lost focus
- **Stopped**: the activity is completely hidden by another application

Activity's callback methods

- **onCreate()**: the activity is created by the operating system;
- **onStart()**: the activity is becoming visible to the user, but the user cannot interact with it yet
- **onResume()**: the activity is visible and the user can interact with it

Activity's callback methods

- **onPause()** : the activity is still visible, but has lost focus
- **onStop()** : the application is no longer visible to the user
- **onRestart()** : the application is restarted after an onStop()
- **onDestroy()** : the last call before the application is really dead ☺

An implementation

```
public class FrmPrincipal extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_frm_principal);
        Log.i("TAVY","onCreate - the application was created");
    }

    @Override
    protected void onStart() {
        super.onStart();
        Log.i("TAVY","onStart - the application was started");
    }

    @Override
    protected void onResume() {
        super.onResume();
        Log.i("TAVY","onResume - the application was resumed (it is visible now)");
    }

    @Override
    protected void onPause() {
        super.onPause();
        Log.i("TAVY","onPause - the application is paused (still visible)");
    }

    @Override
    protected void onStop() {
        super.onStop();
        Log.i("TAVY","onStop - the application is stopped (invisible)");
    }
    @Override
    protected void onDestroy() {
        super.onDestroy();
        Log.i("TAVY","onDestroy - Bye, bye! Adios!");
    }
```

The result of the implementation

Launch the application and then kill it by using the Back Button

```
03-14 17:54:55.781  565-565/? I/TAVY :  onCreate - the application was created
03-14 17:54:55.781  565-565/? I/TAVY :  onStart - the application was started
03-14 17:54:55.791  565-565/? I/TAVY :  onResume - the application was resumed (it is visible now)
03-14 17:55:46.982  565-565/? I/TAVY :  onPause - the application is paused (still visible)
03-14 17:55:49.971  565-565/? I/TAVY :  onStop - the application is stopped (invisible)
03-14 17:55:49.980  565-565/? I/TAVY :  onDestroy - Bye, bye! Adios!
```

Launch the application and then press the Home Button

```
03-14 17:57:16.390  565-565/? I/TAVY :  onCreate - the application was created
03-14 17:57:16.450  565-565/? I/TAVY :  onStart - the application was started
03-14 17:57:16.450  565-565/? I/TAVY :  onResume - the application was resumed (it is visible now)
03-14 17:57:23.662  565-565/? I/TAVY :  onPause - the application is paused (still visible)
03-14 17:57:27.200  565-565/? I/TAVY :  onStop - the application is stopped (invisible)
```

Homework – part 1 ☺

- Define a new activity
- Implement all the lifecycle methods of the activity by putting specific Log(s) information
- Start the activity and look at the Log(s) messages
- Press Back button, then restart the application on the emulator, then press Home button, then restart again… and so on…

General topics

- Activity
- Activity's lifecycle
- Activity's Instance State and screen rotation
- Starting new activities
- Finishing an activity
- Fragments
- Fragment's lifecycle
- Setting listeners in a programmatic way

Configuration changes

On a mobile device there are some types of **configuration changes**:

- change of language;
- change of screen orientation;
- change of input devices.

Configuration changes

- Every time a configuration change happens, **the activity is destroyed and recreated**.
- The destroy process "contains": onPause(), onStop(), **onDestroy().**
- The "re-creation" process contains: **onCreate()**, onStart(), onResume().

Configuration changes – important specifications!!!

When the configuration of the device is changed, all the graphical controls displayed on the activity must be redrawn in order to match the new configuration.

As a result, the “old” graphical controls are “lost”.

Configuration changes – an example

Our application has:

- A layout (activity_frm_portrait_landscape.xml);
- An activity (FrmPortretLandscape.java).

The layout has a TextView and a Button.

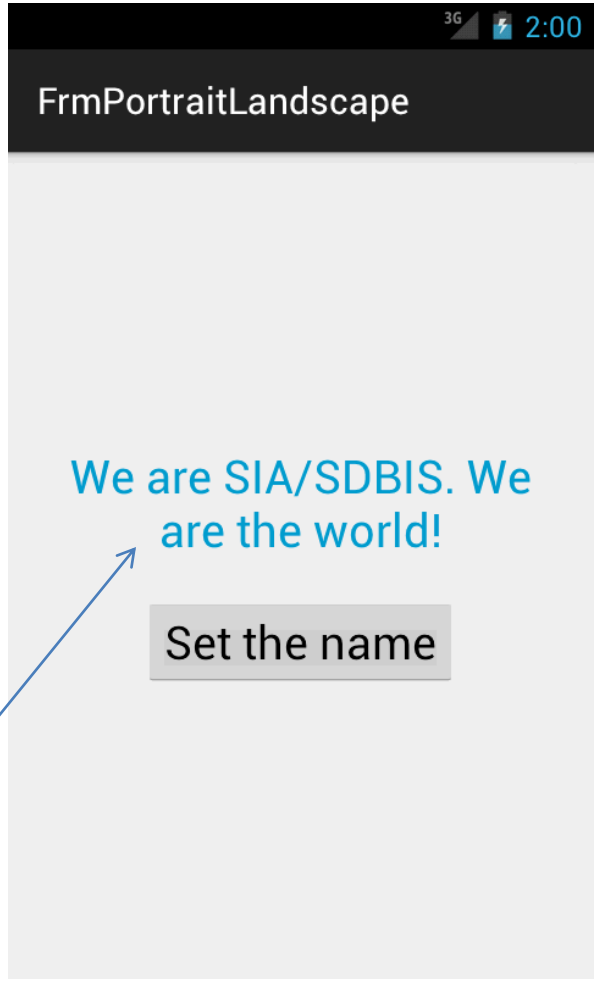
Inside the activity, we implement the OnClickListener for the button. It displays the name on the TextView.

Configuration changes – the layout

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:orientation="vertical"
    android:gravity="center"
    tools:context="com.example.adminlocal.appcurs04.FrmPortraitLandscape">

    <TextView
        android:id="@+id/txtMyName"
        android:text="Your name?"
        android:textSize="25dp"
        android:gravity="center"
        android:textColor="@android:color/holo_blue_dark"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />

    <Button
        android:layout_marginTop="20dp"
        android:id="@+id/btnSetName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Set the name"
        android:textSize="25dp"
        />
</LinearLayout>
```

The activity class

```
public class FrmPortraitLandscape extends Activity {

    Button btnSetTheName;
    TextView txtName;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //set the layout of the activity
        setContentView(R.layout.activity_frm_portrait_landscape);
        //get the graphical controls
        btnSetTheName = (Button) findViewById(R.id.btnSetName);
        txtName = (TextView) findViewById(R.id.txtMyName);
        //set the listener for the button
        btnSetTheName.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                txtName.setText("We are SIA/SDBIS. We are the world!");
            }
        });
    }
…
…
}
```

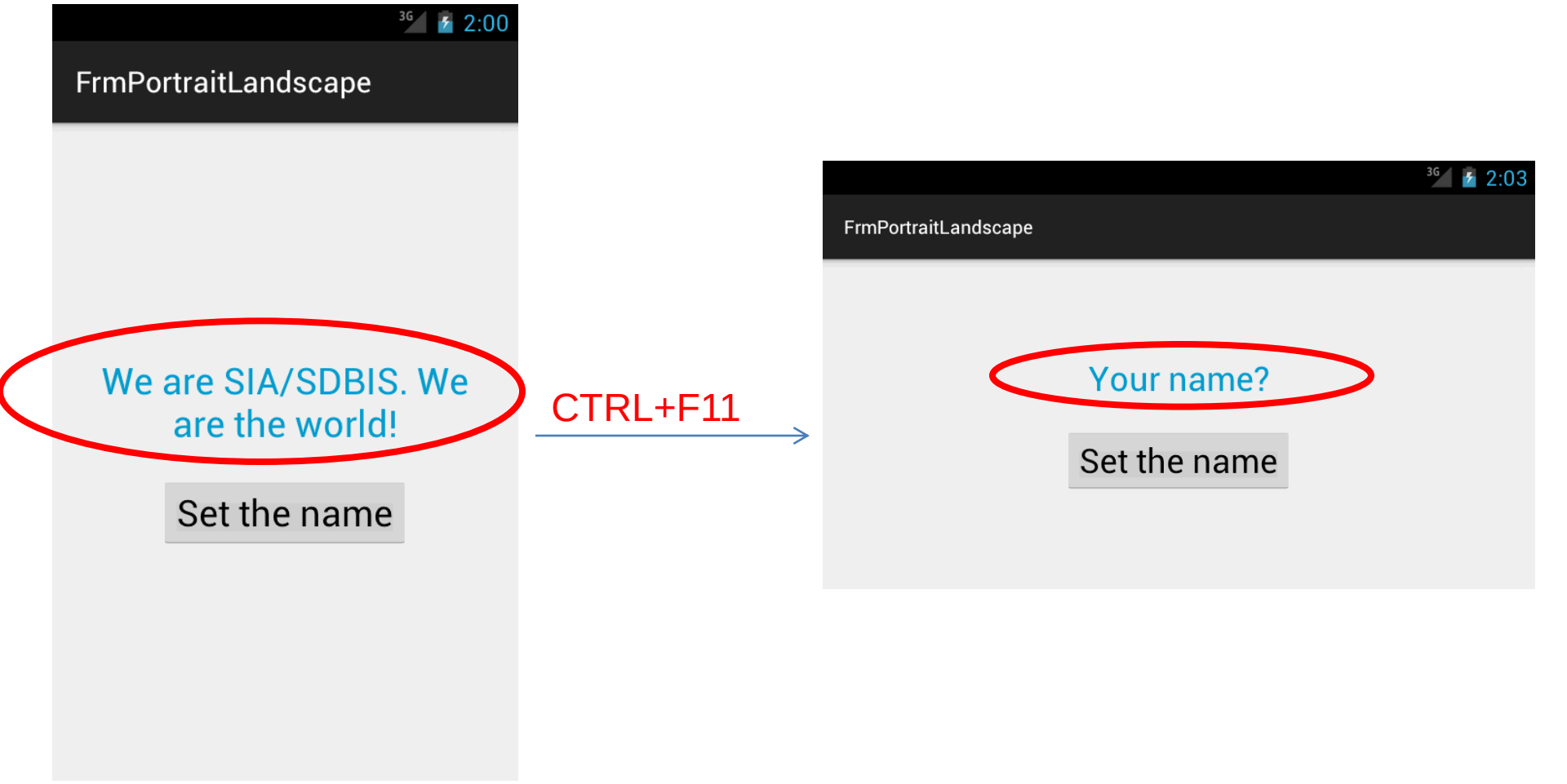


And now… let's change the configuration! ☺

- From the portrait view, we'll change the activity in the landscape mode.

- On the emulator it is used **CRTL+F11** to change from portrait to landscape.

Portrait2Landscape (CRTL+F11)



Portrait2Landscape = configuration change

- When the configuration of the device is changed, all the graphical controls displayed on the activity must be redrawn in order to match the new configuration.

- Every time a configuration change happens, the activity is destroyed and recreated.
- The destroy process "contains": onPause(), onStop(), onDestroy().
- The "re-creation" process contains: **onCreate()**, onStart(), onResume().

Portrait2Landscape = configuration change

- The “re-creation” process contains: **onCreate()**, onStart(), onResume().
- The **onCreate()** contains:

```
public class FrmPortraitLandscape extends Activity {

    Button btnSetTheName;
    TextView txtName;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //set the layout of the activity
        setContentView(R.layout.activity_frm_portrait_landscape);
        ……..
        …. ….
    }
…
…
}
```

So, the activity is re-created using the “default” layout.

How to save the state of the activity in order to preserve it after a configuration change?

- The event onSaveInstanceState(Bundle outState) occurs before the activity is killed by the operating system.
- Use outState to store the data which will be restored when the activity is re-created. The data is stored in a “bundle”.
- The restoration can be implemented in onCreate().

Implementation for saving and restoring the activity's state

```
@Override
  protected void onSaveInstanceState(Bundle outState) {  //when the activity is killed by the system
     super.onSaveInstanceState(outState);
     //we save the name from the activity, before it is being destroyed
     outState.putString("NAME", String.valueOf(txtName.getText()));
  }
```

```
@Override
  protected void onCreate(Bundle savedInstanceState) {   //when the activity is restored by the system
     super.onCreate(savedInstanceState);
     //set the layout of the activity
     setContentView(R.layout.activity_frm_portrait_landscape);
     //get the graphical controls
     btnSetTheName = (Button) findViewById(R.id.btnSetName);
     txtName = (TextView) findViewById(R.id.txtMyName);
     //check if a saved instance state exists
     if(savedInstanceState!=null)
     {
        //use the saved data during onSaveInstanceState
        txtName.setText(savedInstanceState.getString("NAME"));
     }
     //set the listener for the button
     btnSetTheName.setOnClickListener(new View.OnClickListener() {
… … … …
```

Homework ☺

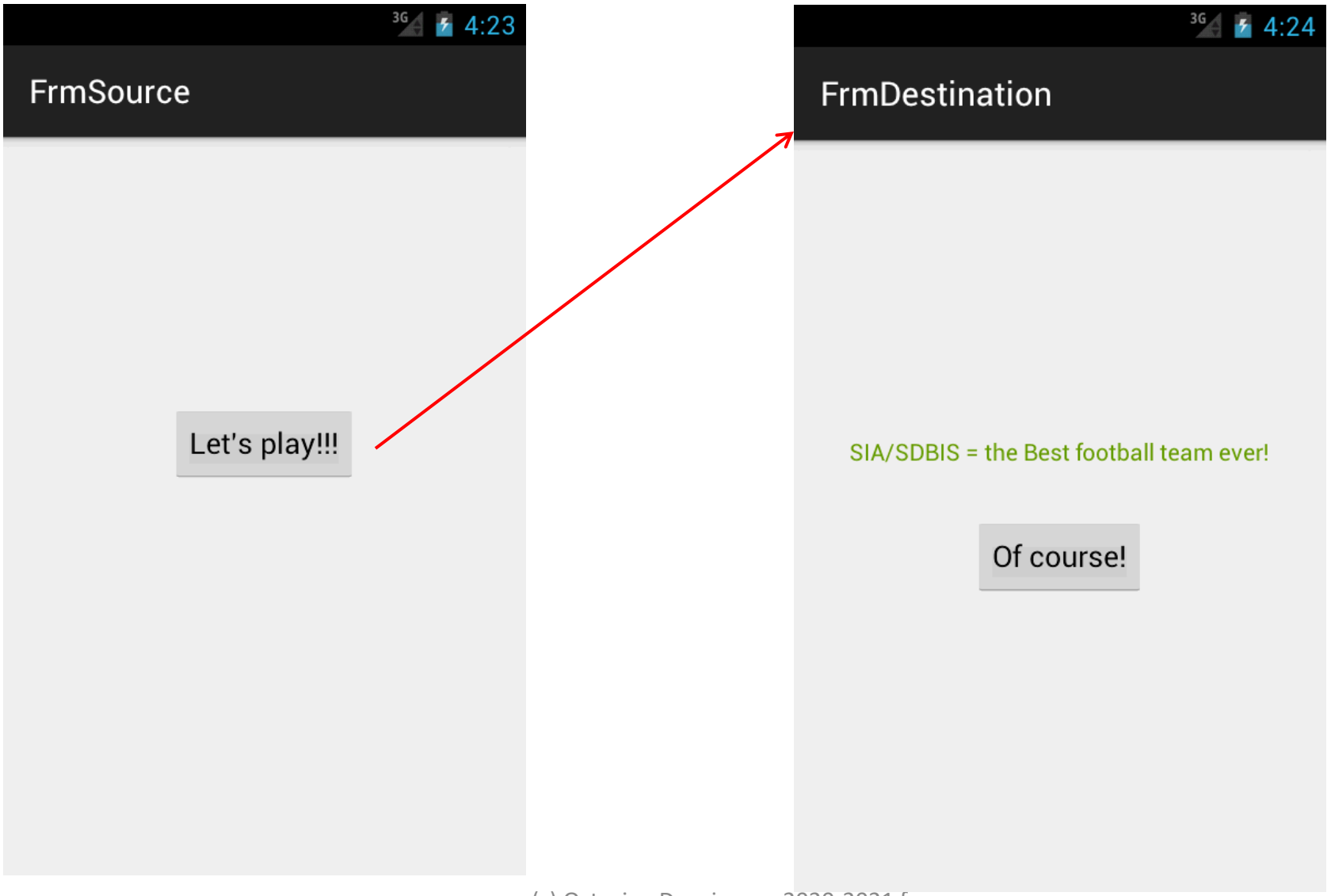
Read about:

- onSaveInstanceState() - http://developer.android.com/reference/android/app/Activity.html#onSaveInstanceState(android.os.Bundle)
- onRestoreInstanceState() - http://developer.android.com/reference/android/app/Activity.html#onRestoreInstanceState(android.os.Bundle)
- onCreate() - http://developer.android.com/reference/android/app/Activity.html#onCreate(android.os.Bundle)

General topics

- Activity
- Activity's lifecycle
- Activity's Instance State and screen rotation
- Starting new activities
- Finishing an activity
- Fragments
- Fragment's lifecycle
- Setting listeners in a programmatic way

The story ☺

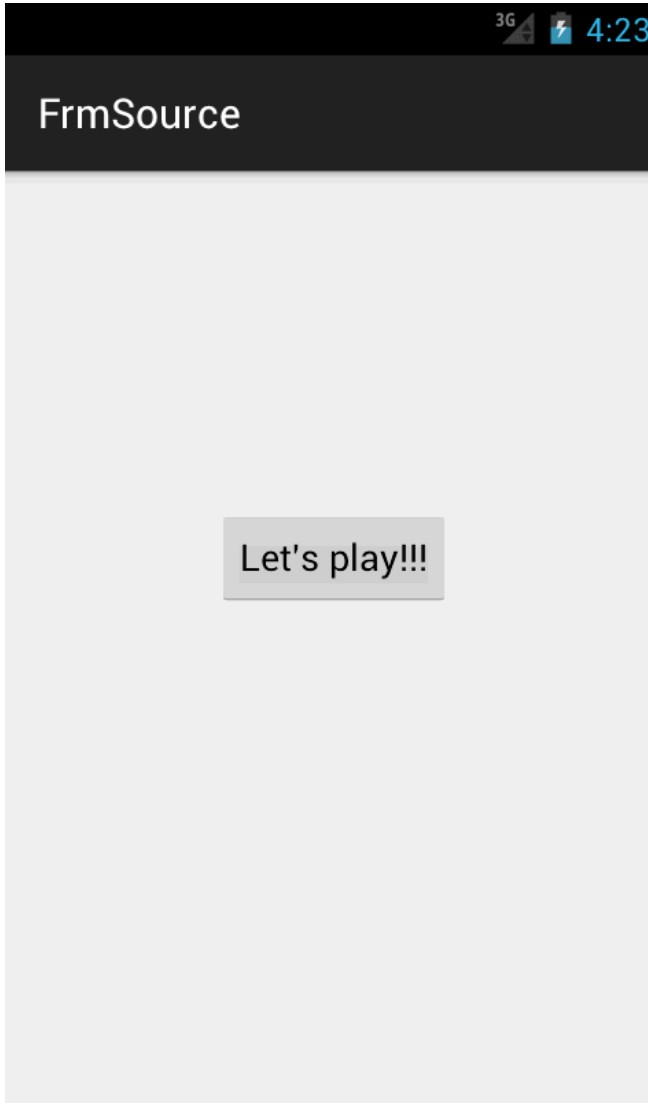

4:23
FrmSource
Let's play!!!
4:24
FrmDestination
SIA/SDBIS = the Best football team ever!
Of course!

The implementation

- 2 activities:
  - FrmSource
  - FrmDestination
- Layouts and buttons

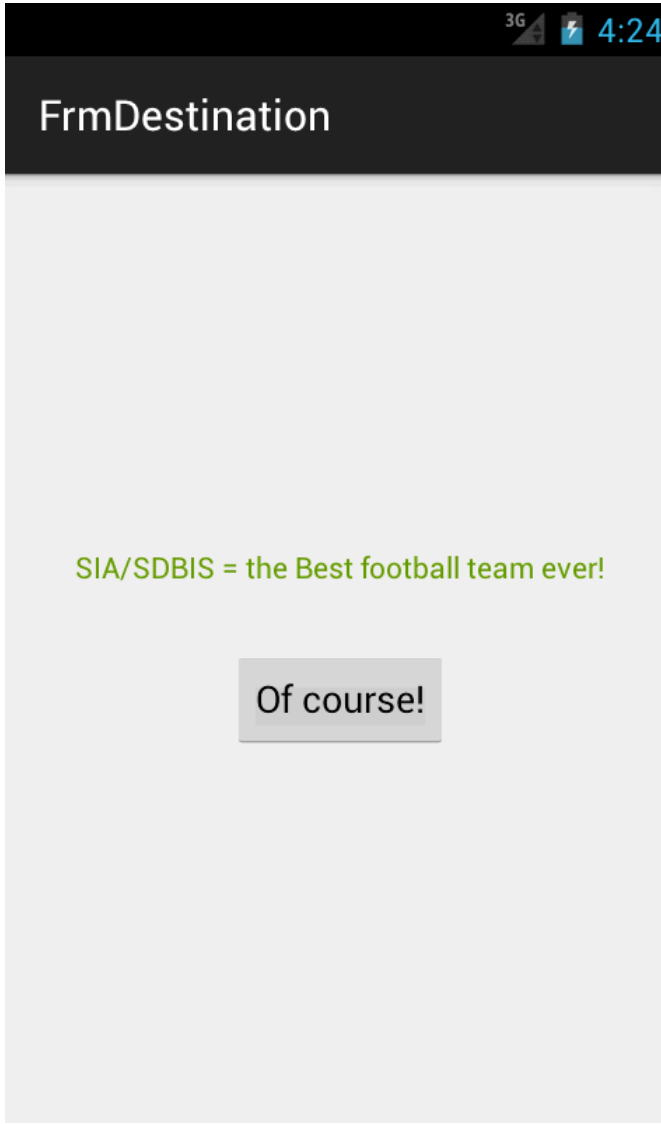
FrmSource - layout

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.example.adminlocal.appcurs04.FrmSource">

    <Button
        android:id="@+id/btnPlay"
        android:text="Let's play!!!"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="148dp" />

</RelativeLayout>
```



FrmDestination - layout

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent“
   android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:orientation="vertical"
    android:gravity="center"
    tools:context="com.example.adminlocal.appcurs04.FrmDestination">
    <TextView
        android:text="SIA/SDBIS = the Best football team ever!"
        android:textColor="@android:color/holo_green_dark"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
    <Button
        android:id="@+id/btnBack"
        android:text="Of course!"
        android:layout_marginTop="30dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</LinearLayout>
```



FrmSource – the implementation

```
public class FrmSource extends  Activity {

    Button btnStartNewActivity;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_frm_source);
      //get the button
      btnStartNewActivity = (Button) findViewById(R.id.btnPlay);
      //define the behavior of the button
      btnStartNewActivity.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            //define a new Intent
            Intent i = new Intent(FrmSource.this, FrmDestination.class);
            //starting the new Activity
            startActivity(i);
         }
      });
```

General topics

- Activity
- Activity's lifecycle
- Activity's Instance State and screen rotation
- Starting new activities
- Finishing an activity
- Fragments
- Fragment's lifecycle
- Setting listeners in a programmatic way

FrmDestination – the implementation

```
public class FrmDestination extends Activity {

    Button btnGoBack;

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

        //get the button
        btnGoBack = (Button) findViewById(R.id.btnBack);
        btnGoBack.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //just finish the current activity
                finish();
            }
        });
    }
```

Homework ☺

Read about:

- **startActivityForResult()** - http://developer.android.com/reference/android/app/Activity.html#StartingActivities, and here: http://developer.android.com/reference/android/app/Activity.html#startActivityForResult(android.content.Intent, int)
- **onActivityResult()** - http://developer.android.com/reference/android/app/Activity.html#onActivityResult(int, int, android.content.Intent)

Final homework (It's a joke. It's not final!) ☺

- Implement 3 activities:
  - FrmActivityJohn;
  - FrmActivityMary;
  - FrmActivityTom.
- Implement buttons and behaviours:
  - From John you can go to Mary;
  - From Mary you can go to Tom;
  - From Tom you can go to John or Mary.

Another homework (I told you it was a joke!) ☺

- Implement all the events onCreate(), onStart(), onResume(), onPause(), onStop(), onDestroy(), onSaveInstanceState() for the previous 3 activities (John, Mary, Tom).

- Check how these events are called when you start the application and then go from one activity to other. Also, change the screen orientation.

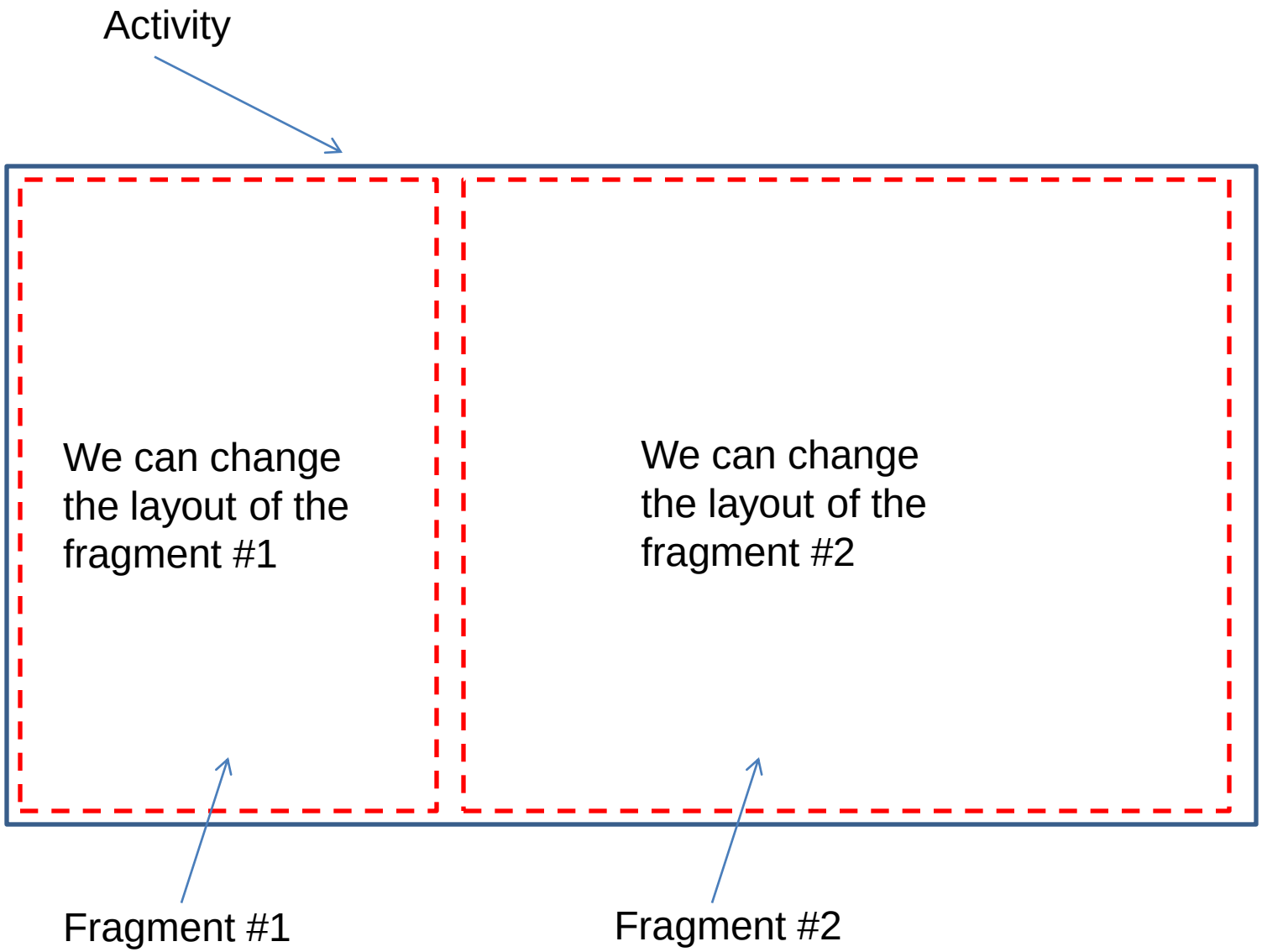
- Check if onSaveInstanceState() is called when you invoke finish(). Why?!?

General topics

- Activity
- Activity's lifecycle
- Activity's Instance State and screen rotation
- Starting new activities
- Finishing an activity
- Fragments
- Fragment's lifecycle
- Setting listeners in a programmatic way

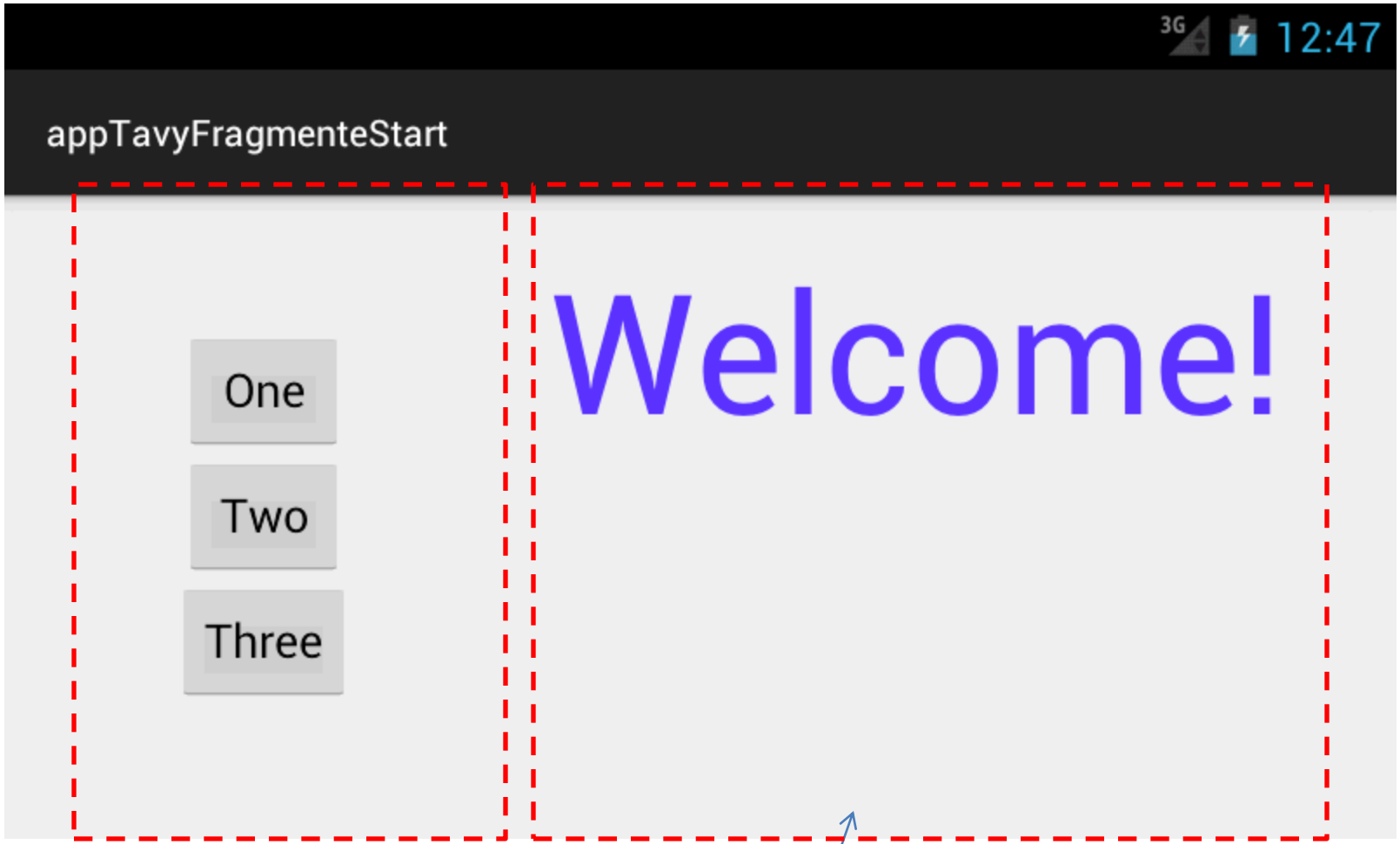
Fragments

- An activity layout can be organized with “fragments”.
- We obtain more flexibility.
- A fragment is an area that can be (re)defined whenever is necessary.

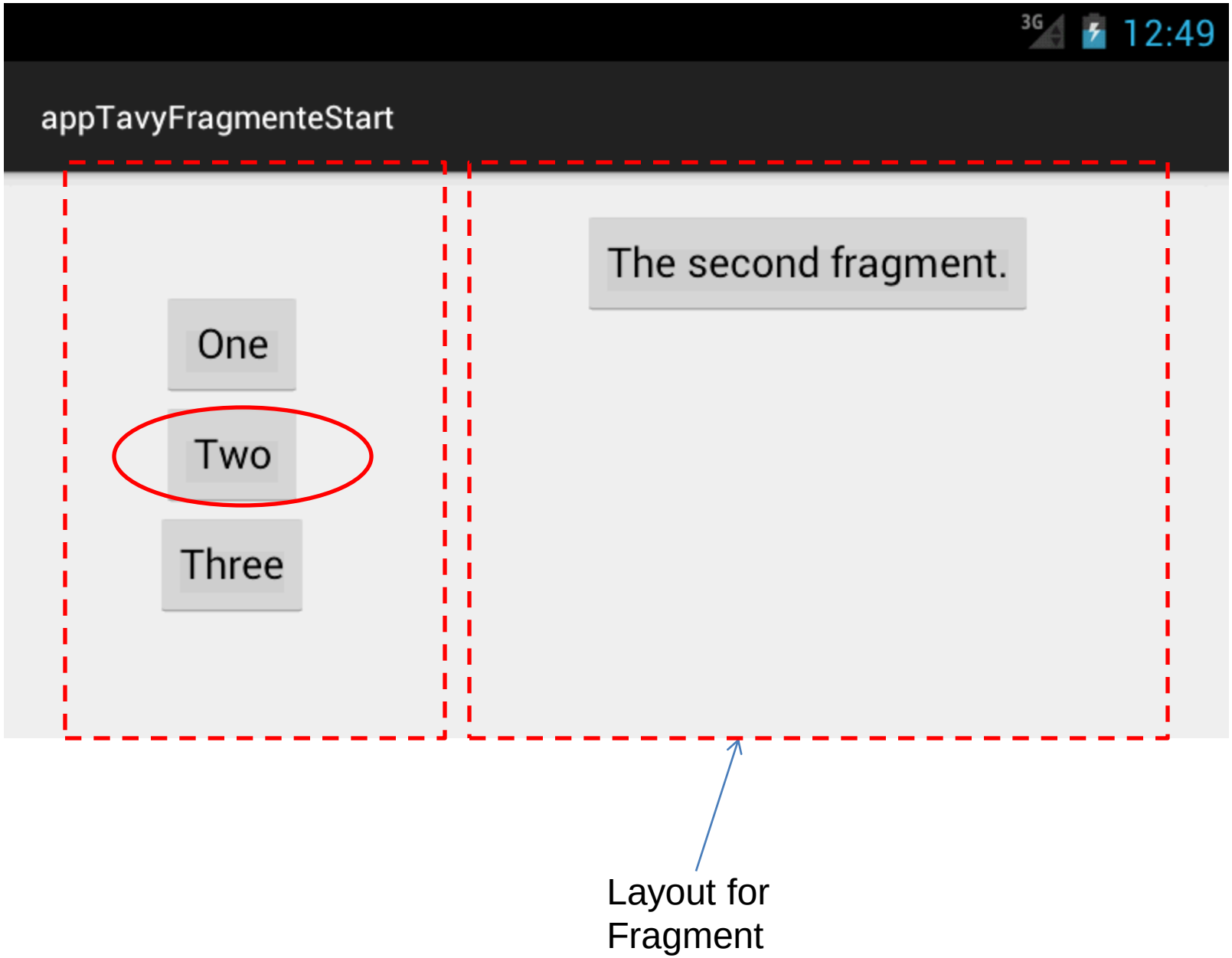

Activity
We can change the layout of the fragment #1
We can change the layout of the fragment #2
Fragment #1
Fragment #2

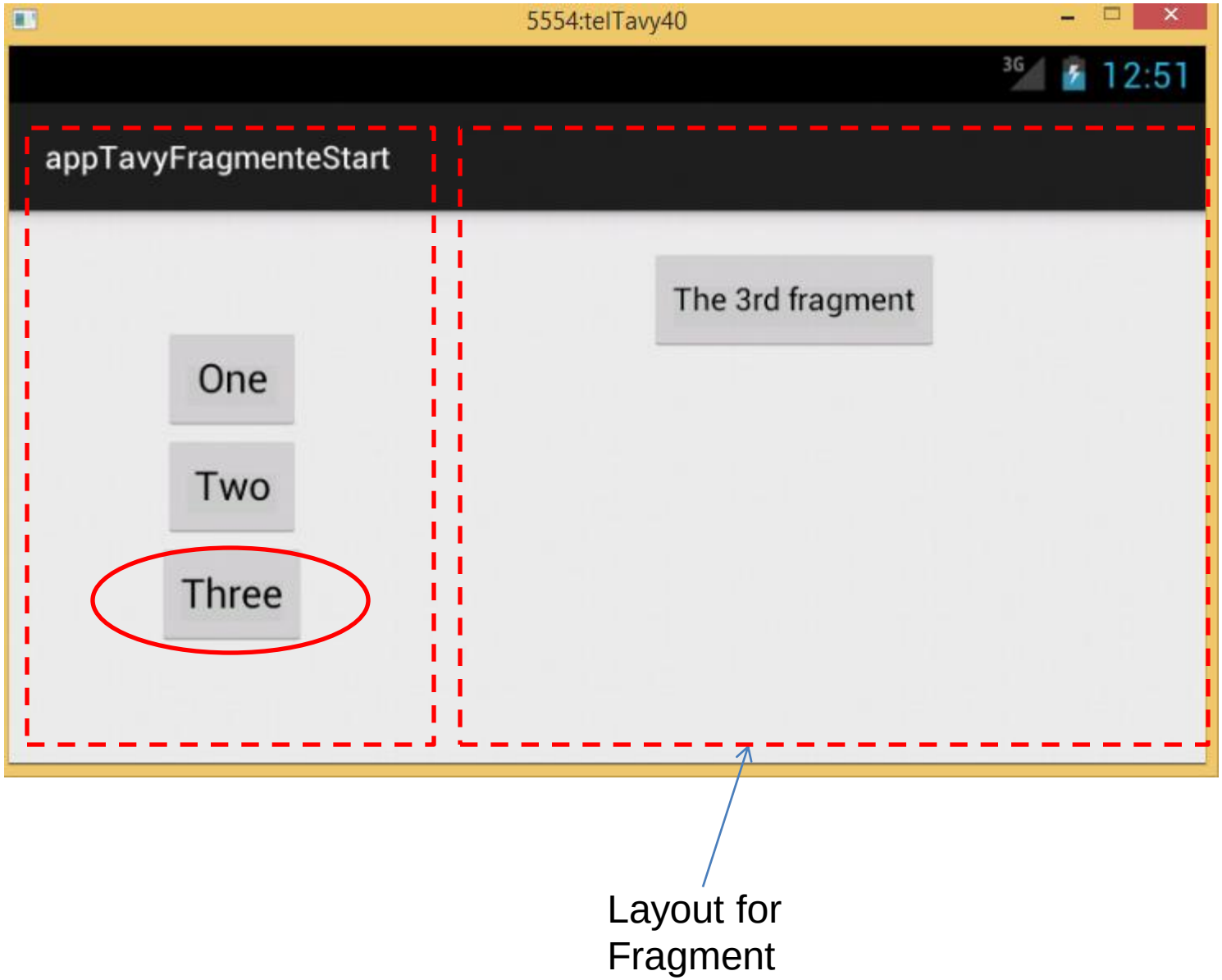
Activity and fragments

- Every activity is an instance of **Activity** class
- Every fragment is an instance of **Fragment** class
- Inside the activity, we use the **FragmentManager** to manage the fragments
- The fragment manager is able to replace the fragments/layouts on the activity, using **fragment transactions**



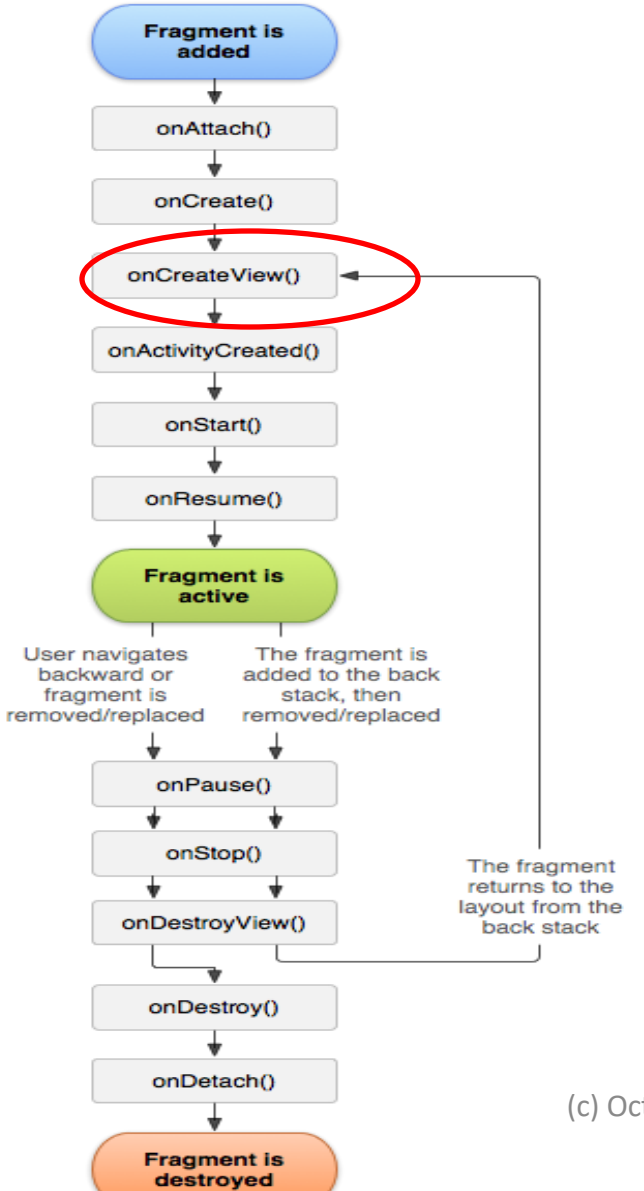
Layout for Fragment

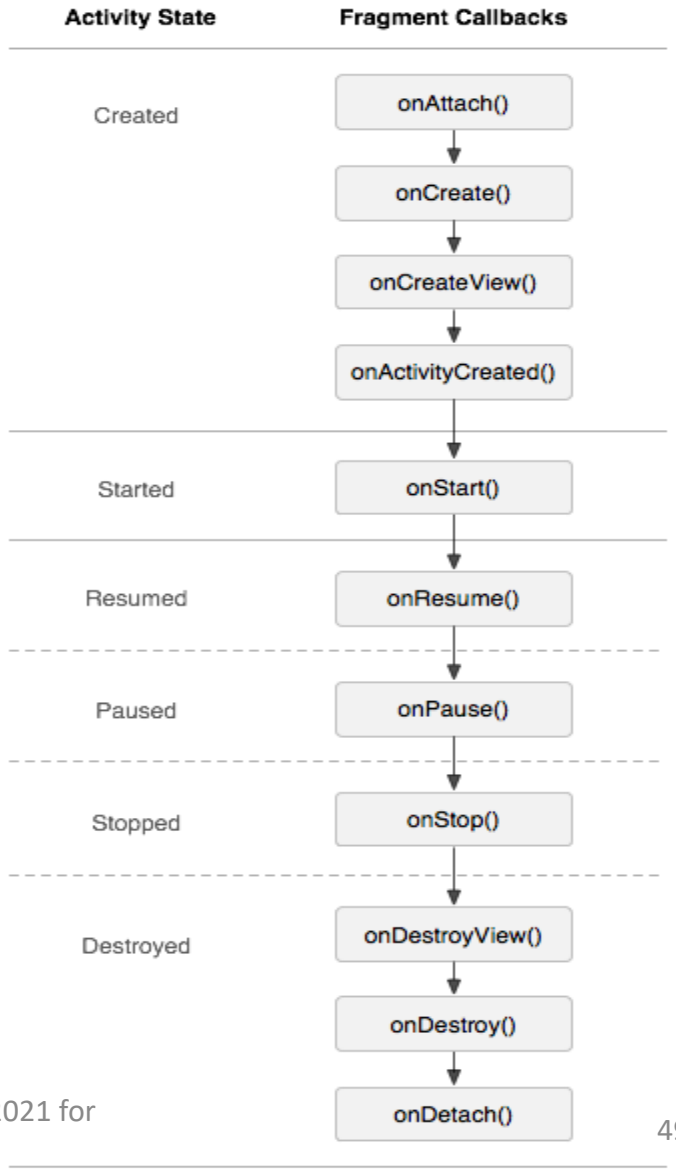

3G
12:49
appTavyFragmenteStart
The second fragment.
One
Two
Three
Layout for
Fragment


5554:telTavy40
3G
12:51
appTavyFragmenteStart
The 3rd fragment
One
Two
Three
Layout for
Fragment

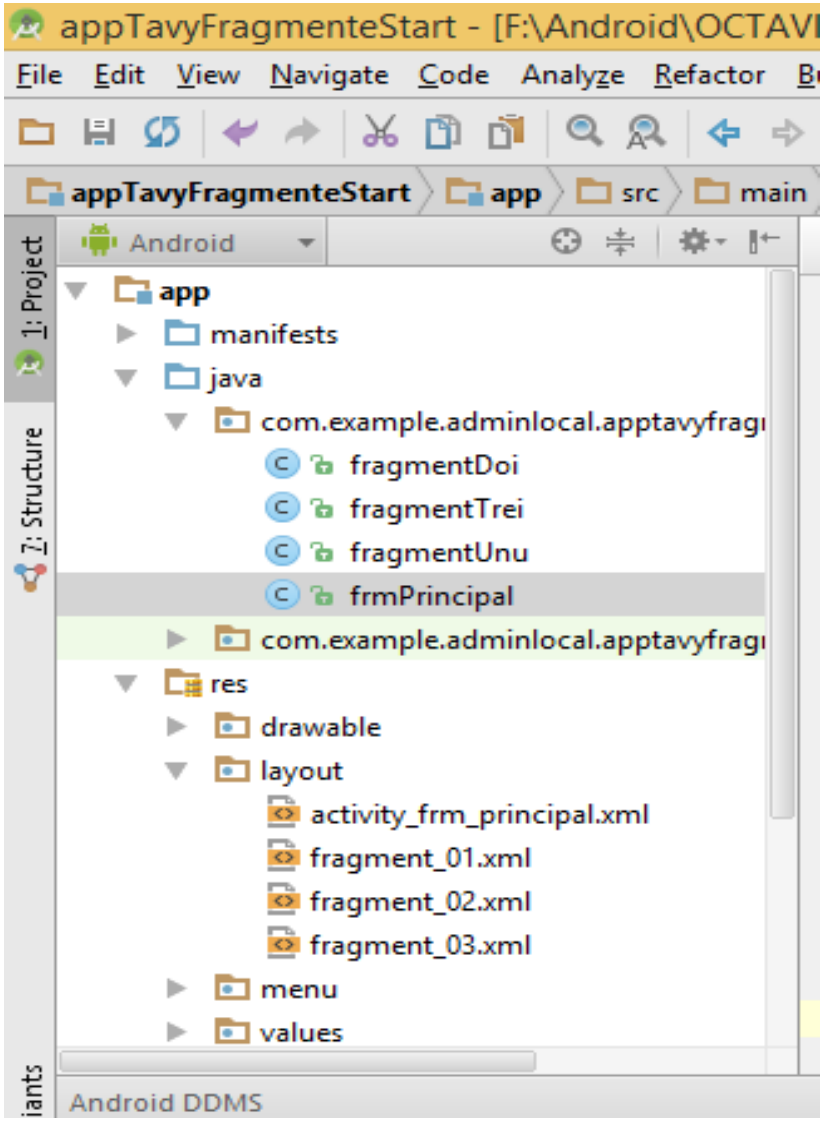
Fragment's lifecycle

(according to: http://developer.android.com/guide/components/fragments.html)





Example


appTavyFragmenteStart - [F:\Android\OCTAVI
File Edit View Navigate Code Analyze Refactor B
appTavyFragmenteStart app src main
Android
1: Project
7: Structure
app
manifests
java
com.example.adminlocal.apptavyfragı
fragmentDoi
fragmentTrei
fragmentUnu
frmPrincipal
com.example.adminlocal.apptavyfragı
res
drawable
layout
activity_frm_principal.xml
fragment_01.xml
fragment_02.xml
fragment_03.xml
menu
values
Android DDMS

Layout for the activity **frmPrincipal**

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:orientation="horizontal"
    android:layout_height="match_parent"
android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin" tools:context=".frmPrincipal">

    <LinearLayout
        android:id="@+id/layoutStanga"
        android:layout_weight="1"
        android:orientation="vertical"
        android:layout_width="0dp"
        android:gravity="center"
        android:layout_height="fill_parent">
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="One"
            android:id="@+id/btnUnu" />
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Two"
            android:id="@+id/btnDoi" />
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Three"
            android:id="@+id/btnTrei" />
    </LinearLayout>
    <LinearLayout
        android:id="@+id/layoutDreapta"
        android:layout_weight="2"
        android:orientation="vertical"
        android:layout_width="0dp"
        android:layout_height="fill_parent"></LinearLayout>
</LinearLayout>
```

Layout for **fragment_01**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:weightSum="1">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="Welcome!"
        android:id="@+id/textView"
        android:layout_gravity="center_horizontal"
        android:layout_weight="0.03"
        android:textColor="#ff5a33ff"
        android:textSize="@android:dimen/notification_large_icon_width" />
</LinearLayout>
```

Layout for **fragment_02**

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

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="The second fragment."
        android:id="@+id/buttonPropriu"
        android:layout_gravity="center_horizontal" />
</LinearLayout>
```

Layout for **fragment_03**

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

    <Button
        style="?android:attr/buttonStyleSmall"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="The 3rd fragment"
        android:id="@+id/button2"
        android:layout_gravity="center_horizontal" />
</LinearLayout>
```

Implementation for **fragmentUnu**

```
import android.os.Bundle;
import android.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;


public class fragmentUnu extends Fragment {

    public fragmentUnu() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
       //aici creez fragmentul pe baza propriului meu layout
        View fragmentulMeu;
        fragmentulMeu = inflater.inflate(R.layout.fragment_01,container,false);

        return fragmentulMeu;
    }
}
```

Implementation for **fragmentDoi**

```
public class fragmentDoi extends Fragment {


    public fragmentDoi() {
        // Required empty public constructor
    }


    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
       View fragmentulMeu = inflater.inflate(R.layout.fragment_02,container,false);
        //obtin butonul
        Button btnFragment = (Button) fragmentulMeu.findViewById(R.id.buttonPropriu);
        btnFragment.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                (Toast.makeText(getActivity(),"Mergeeee!!!",Toast.LENGTH_LONG)).show();


            }
        });
        //
        return fragmentulMeu;
    }
```

Implementation for **fragmentTrei**

```
public class fragmentTrei extends Fragment {


    public fragmentTrei() {
        // Required empty public constructor
    }


    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
       View fragmentulMeu = inflater.inflate(R.layout.fragment_03,container,false);
        return fragmentulMeu;
    }


}
```

Implementation for **frmPrincipal**

```
public class frmPrincipal extends  Activity {

    Button btnUnu, btnDoi, btnTrei;
    FragmentManager fm;
    FragmentTransaction ft;

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

        //obtin butoanele
        btnUnu = (Button)findViewById(R.id.btnUnu);
        btnDoi=(Button)findViewById(R.id.btnDoi);
        btnTrei=(Button)findViewById(R.id.btnTrei);

        fm = getFragmentManager();
        fm.beginTransaction().add(R.id.layoutDreapta,new fragmentUnu()).commit();
```

Implementation for **frmPrincipal**

```
//definesc actiunile pentru fiecare buton in parte
    btnUnu.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            fragmentUnu frg=new fragmentUnu();

            ft=fm.beginTransaction();
            ft.replace(R.id.layoutDreapta,frg);
            ft.commit();
        }
    });

    btnDoi.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            fragmentDoi frg= new fragmentDoi();
            ft=fm.beginTransaction();
            ft.replace(R.id.layoutDreapta,frg);
            ft.commit();
        }
    });

    btnTrei.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            fragmentTrei frg=new fragmentTrei();
            ft=fm.beginTransaction();
            ft.replace(R.id.layoutDreapta,frg);
            ft.commit();
```

Communicating with the Activity

- The fragment is an independent object from an Activity.
- One fragment can be used in many activities.
- Once a fragment is used in an activity, we can access the “parent” activity of the fragment:

```
Button btnFromFragment = (Button) (getActivity().findViewById(R.id.btnOne));
```

Communicating with the Activity

- The activity can call methods in fragment by using the FragmentManager

```
FragmentOne frg = (FragmentOne)getFragmentManager().findFragmentById(R.id.fragment_01);
```

Homework ☺

- Read and implement about fragments at: : http://developer.android.com/guide/components/fragments.html

The complete code (at the lab☺)