КАЛУЖСКИЙ ФИЛИАЛ
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО
ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИМЕНИ Н.Э. БАУМАНА (национальный исследовательский университет)»

Факультет "Фундаментальные науки"

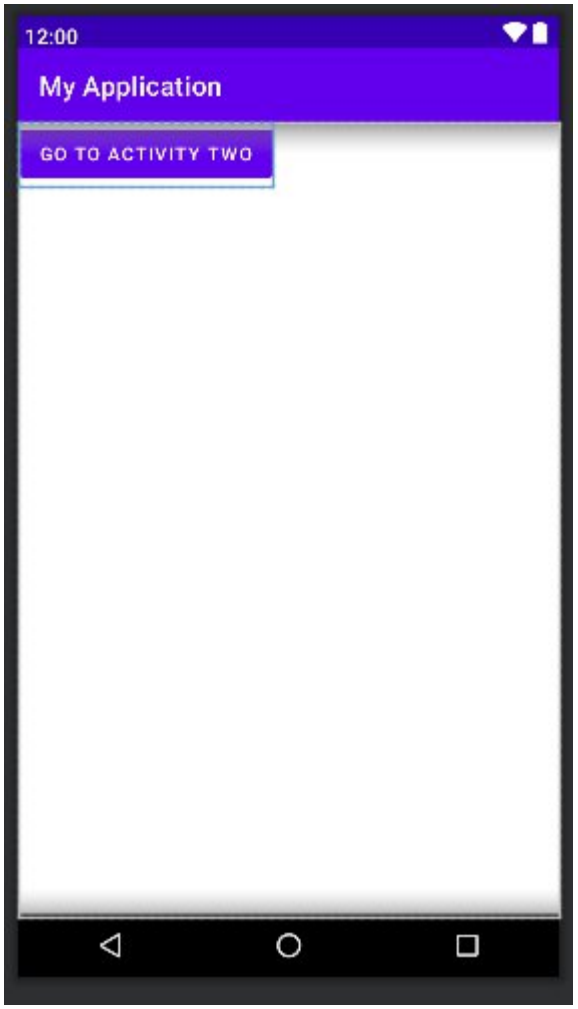
Кафедра "Программное обеспечение ЭВМ, информационные технологии и прикладная математика"

Многоэкранные приложения

Калуга

Создадим приложение, в котором вызывается две активности.
Ниже приведена разметка для первой активности

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

    <Button
        android:id="@+id/btnActTwo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Go to Activity Two" >
    </Button>

</LinearLayout>
```



На экране одна кнопка, по нажатию которой вызывается второй экран.

```java
//Java
public class MainActivity extends AppCompatActivity{
    Button btnActTwo;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        btnActTwo = (Button) findViewById(R.id.btnActTwo);
        btnActTwo.setOnClickListener((View.OnClickListener) this);

    }
    public void onClick(View v) {
        switch (v.getId()) {
            case R.id.btnActTwo:
                // TODO Call second activity
                break;
            default:
                break;
        }
    }
}
```

```kotlin
//Kotlin
class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?)
{
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        btnActTwo.setOnClickListener(this::OnClick)
    }

    fun OnClick(view: View){
        when(view.getId()){
            R.id.btnActTwo -> {
                //TODO Call second activity
            }
        }
    }

}
```

Определена кнопка btnActTwo и ей присвоена Activity в качестве обработчика. Реализация метода onClick для кнопки пока заполнена частично - определяем, какая кнопка была нажата. Чуть позже здесь будет вызываться второй экран. Но сначала второй экран надо создать.

```java
//Java
package com.example.les_21;

import android.app.Activity;

public class ActivityTwo extends AppCompatActivity
{
}
```

```kotlin
//Kotlin
package com.example.les_21

import
androidx.appcompat.app.AppCompatActivity

class ActivityTwo : AppCompatActivity() {
}
```

Класс ActivityTwo создан. Он абсолютно пустой. Необходимо реализовать метод onCreate, который вызывается при создании Activity:

```java
//Java
package com.example.les_21;

import android.app.Activity;
import android.os.Bundle;

public class ActivityTwo extends AppCompatActivity {

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

}
```

```kotlin
//Kotlin
package com.example.les_21;

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity

class ActivityTwo : AppCompatActivity() {
  override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
  }
}
```

```
<?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/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="This is Activity Two" >
    </TextView>

</LinearLayout>
```

Экран будет отображать TextView с текстом "This is Activity Two". Используем файл second.xml в методе setContentView в ActivityTwo.java

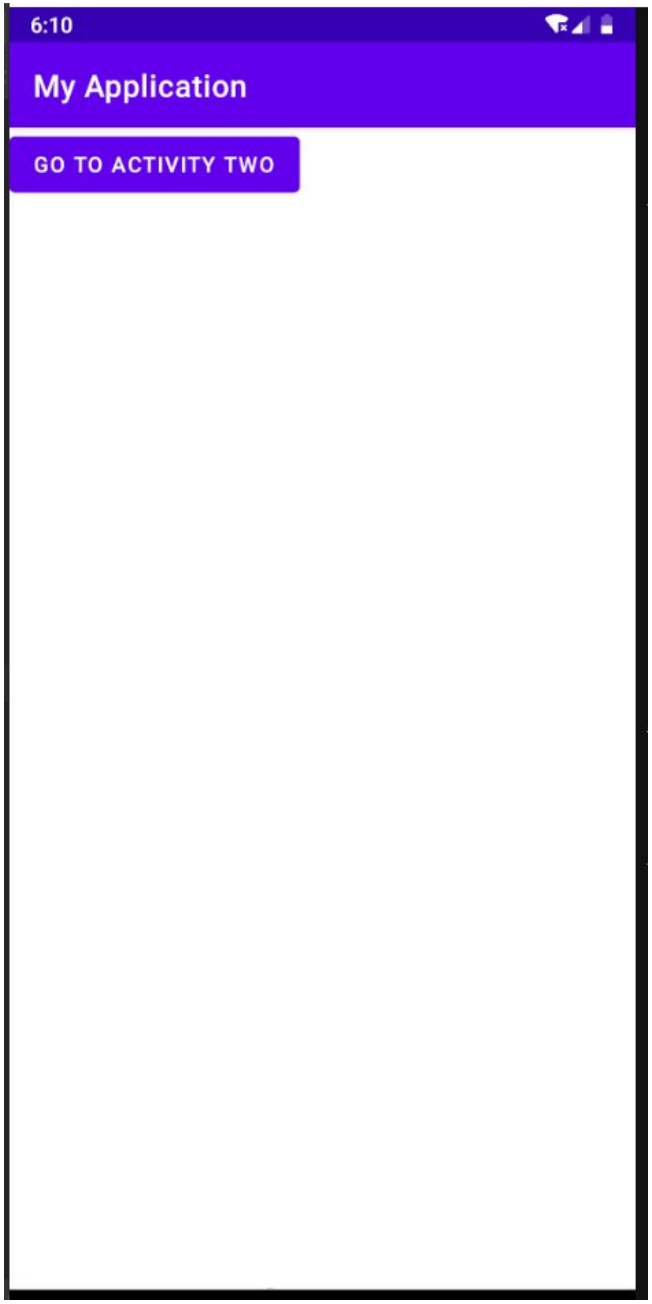
```
//Java
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.second);
  }
```

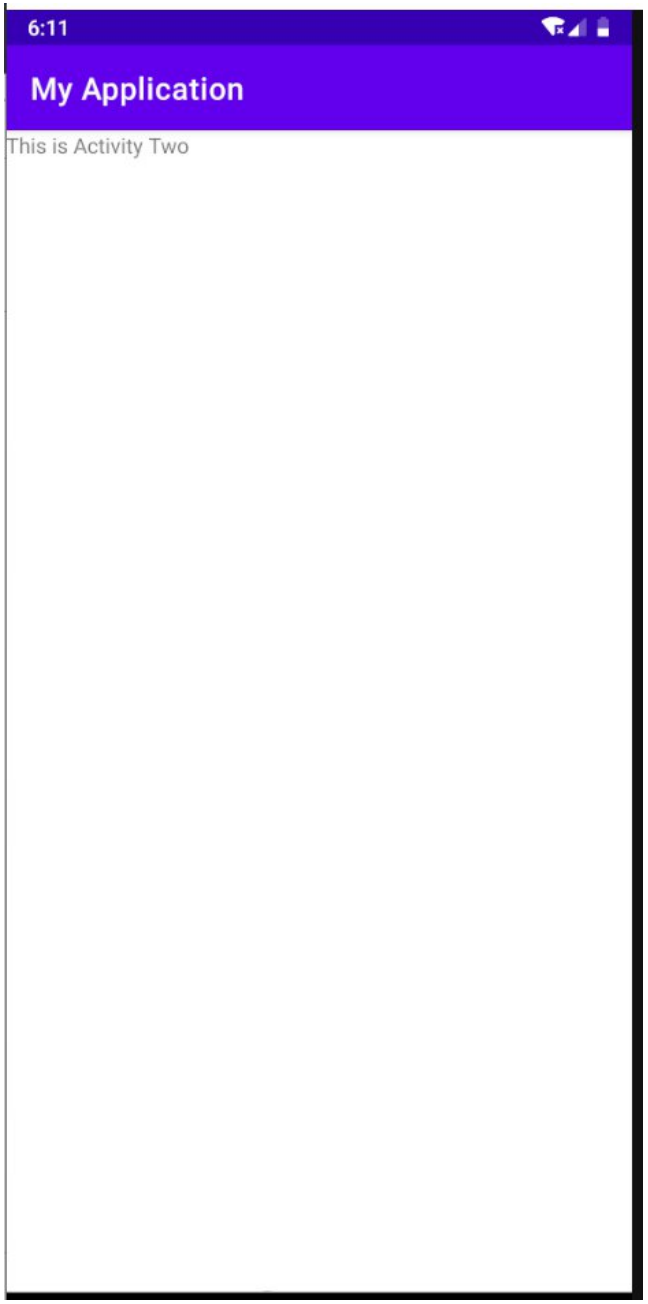
```
//Kotlin
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.second)
  }
```

Необходимо вернуться в MainActivity.java и завершить реализацию метода onClick (нажатие кнопки), а именно - прописать вызов ActivityTwo. Открываем MainActivity.java и добавляем строки:

```
//Java
package com.example.les_21;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {
Button btnActTwo;

@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
btnActTwo = (Button) findViewById(R.id.btnActTwo);
btnActTwo.setOnClickListener(this);
};

public void onClick(View v) {
switch (v.getId()) {
case R.id.btnActTwo:
Intent intent = new Intent(this, ActivityTwo.class);
startActivity(intent);
break;
default:
break;
}}}
```

```kotlin
//Kotlin
package com.example.les_21

import android.content.Intent
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        btnActTwo.setOnClickListener(this::OnClick)
    }

    fun OnClick(view: View){
        when(view.getId()){
            R.id.btnActTwo -> {
                var intent = Intent(this@MainActivity, ActivityTwo::class.java)
                startActivity(intent)
            }
        }
    }

}
```


6:10
My Application
GO TO ACTIVITY TWO


6:11
My Application
This is Activity Two

Рассмотрим код вызова Activity.

```
//Java
Intent intent = new Intent(this, ActivityTwo.class);
startActivity(intent);

//Kotlin
var intent = Intent(this@MainActivity, ActivityTwo::class.java)
startActivity(intent)
```

Использован объект Intent. Рассмотрим данный объект более подробно.

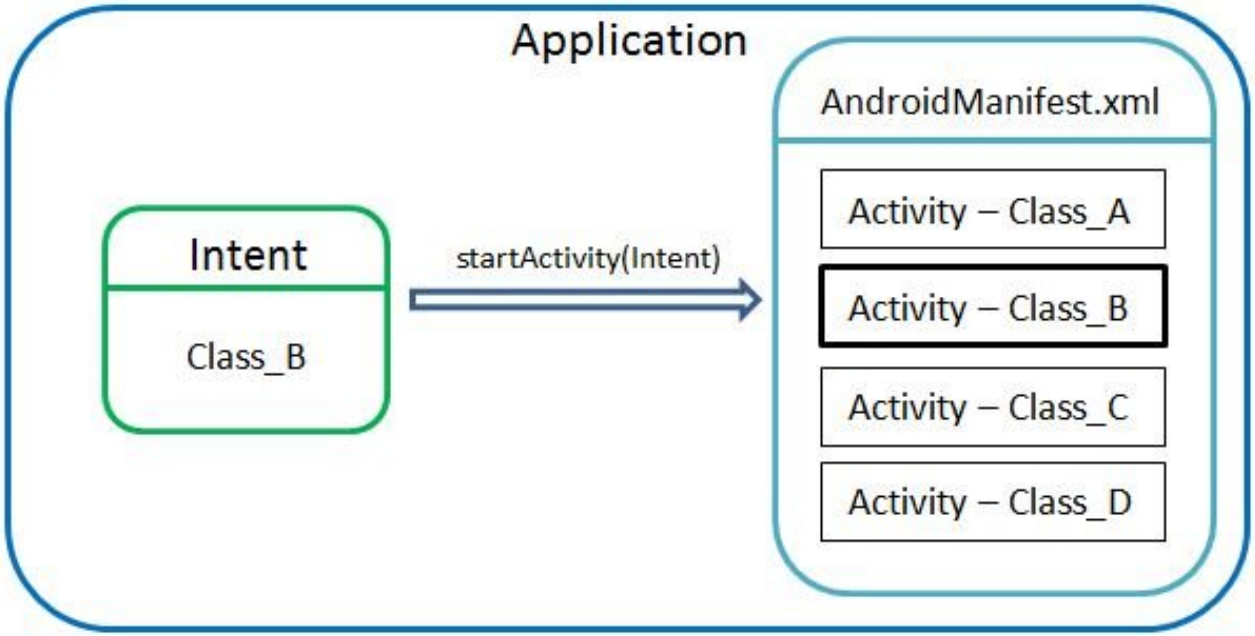
В нашем случае Intent – это объект, в котором указывается, какое Activity необходимо вызвать. После чего этот Intent-объект передается методу startActivity, который находит соответствующее Activity и показывает его. При создании Intent использовался конструктор

`Intent (Context packageContext, Class cls)` с двумя параметрами.

Первый параметр – это Context. При программном создании View в конструкторах используется объект Context. Activity является подклассом Context, поэтому можно использовать ее – this. Вообще, Context – это объект, который предоставляет доступ к базовым функциям приложения таким как: доступ к ресурсам, к файловой системе, вызов Activiy и т.д.

Второй параметр – имя класса. При создании записи Activity в манифест-файле указывается имя класса. Поэтому если теперь указать тот же класс в Intent – то система, просмотрев манифест-файл обнаружит соответствие и покажет соответствующий Activity.

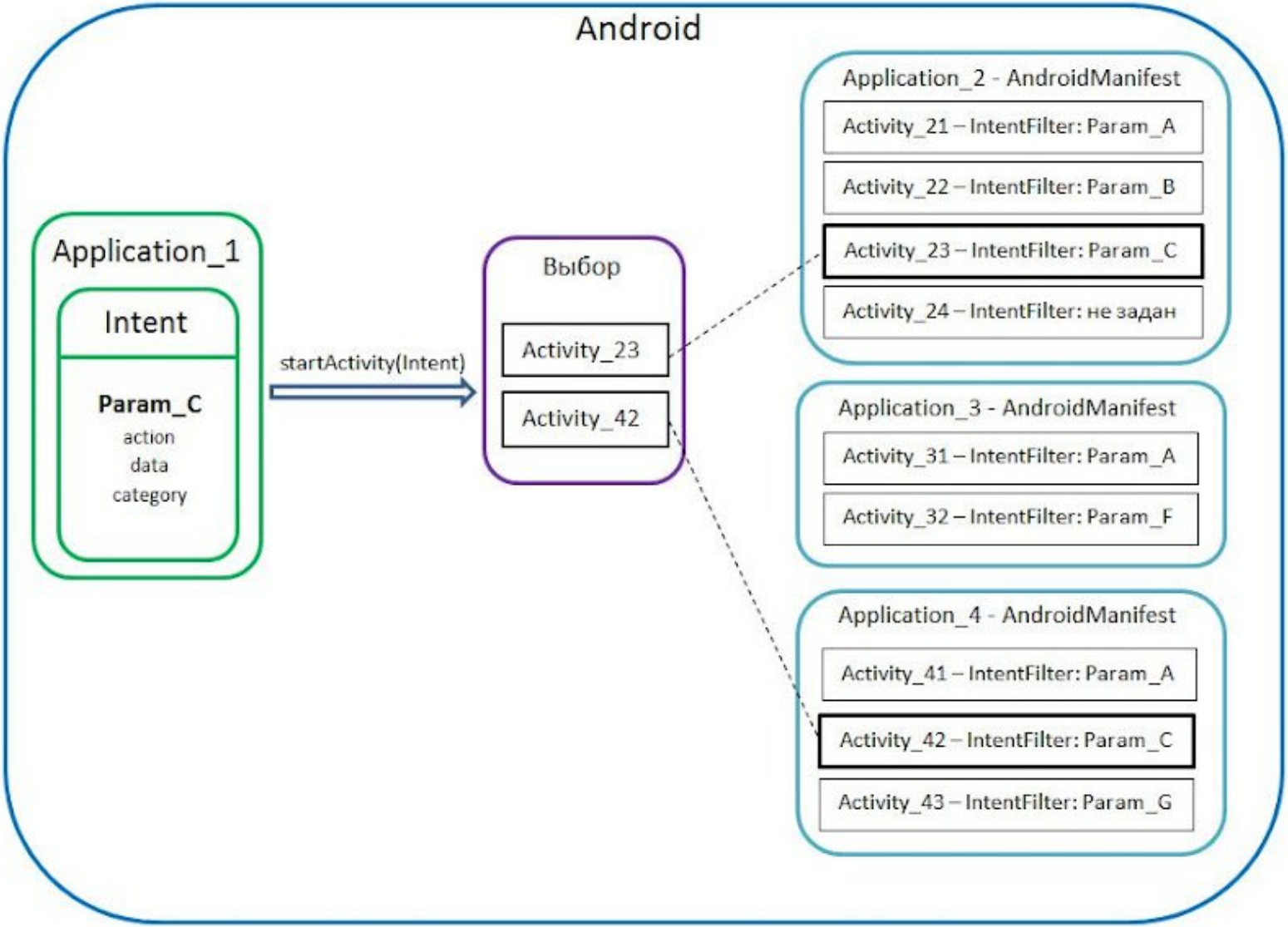
Вызов Activity с помощью такого Intent – это явный вызов. Т.е. с помощью класса явно указывается какое Activity необходимо отобразить. Это обычно используется внутри одного приложения. Схематично это можно изобразить так:



Здесь создается Intent и в качестве параметра ему передается класс Class_B. Далее вызывается метод startActivity с созданным Intent в качестве параметра. Метод проверяет AndroidManifest на наличие Activity связанной с классом Class_B и если находит, то отображает. Все это в пределах одного приложения.

Неявный вызов Activity

Существует также неявный вызов Activity. Он отличается тем, что при создании Intent **используется не класс, а заполняются параметры action, data, category** определенными значениями. Комбинация этих значений определяет цель, которую необходимо достичь. Например: отправка письма, открытие гиперссылки, редактирование текста, просмотр картинки, звонок по определенному номеру и т.д. В свою очередь для Activity прописывается **Intent Filter** - это набор тех же параметров: action, data, category (но значения уже свои - зависят от того, что умеет делать Activity). И если параметры нашего Intent совпадают с условиями этого фильтра, то Activity вызывается. Но при этом поиск уже идет по всем Activity всех приложений в системе. Если находится несколько, то система предоставляет вам выбор, какой именно программой вы хотите воспользоваться. Схематично это можно изобразить так:


Android
Application_1
Intent
Param_C
action
data
category
startActivity(Intent)
Выбор
Activity_23
Activity_42
Application_2 - AndroidManifest
Activity_21 – IntentFilter: Param_A
Activity_22 – IntentFilter: Param_B
Activity_23 – IntentFilter: Param_C
Activity_24 – IntentFilter: не задан
Application_3 - AndroidManifest
Activity_31 – IntentFilter: Param_A
Activity_32 – IntentFilter: Param_F
Application_4 - AndroidManifest
Activity_41 – IntentFilter: Param_A
Activity_42 – IntentFilter: Param_C
Activity_43 – IntentFilter: Param_G

В Application_1 создается Intent, заполняются параметры action, data, category. Для удобства, получившийся набор параметров назовем Param_C. С помощью startActivity этот Intent отправляется на поиски подходящей Activity, которая сможет выполнить то, что нам нужно (т.е. то, что определено с помощью Param_C). В системе есть разные приложения, и в каждом из них несколько Activity. Для некоторых Activity определен Intent Filter (наборы Param_A, Param_B и т.д.), для некоторых нет. Метод startActivity сверяет набор параметров Intent и наборы параметров Intent Filter для каждой Activity. Если наборы совпадают (Param_C для обоих), то Activity считается подходящей.
Если в итоге нашлась только одна Activity – она и отображается. Если же нашлось несколько подходящих Activity, то пользователю выводится список, где он может сам выбрать какое приложение ему использовать.

Например, если в системе установлено несколько музыкальных плееров, и вы запускаете mp3, то система выведет вам список Activity, которые умеют играть музыку и попросит выбрать, какое из них использовать. А те Activity, которые умеют редактировать текст, показывать картинки, звонить и т.п. будут проигнорированы.
Если для Activity не задан Intent Filter (Activity_24 на схеме), то Intent с параметрами ему никак не подойдет, и оно тоже будет проигнорировано.

Состояние Activity. Activity Lifecycle.

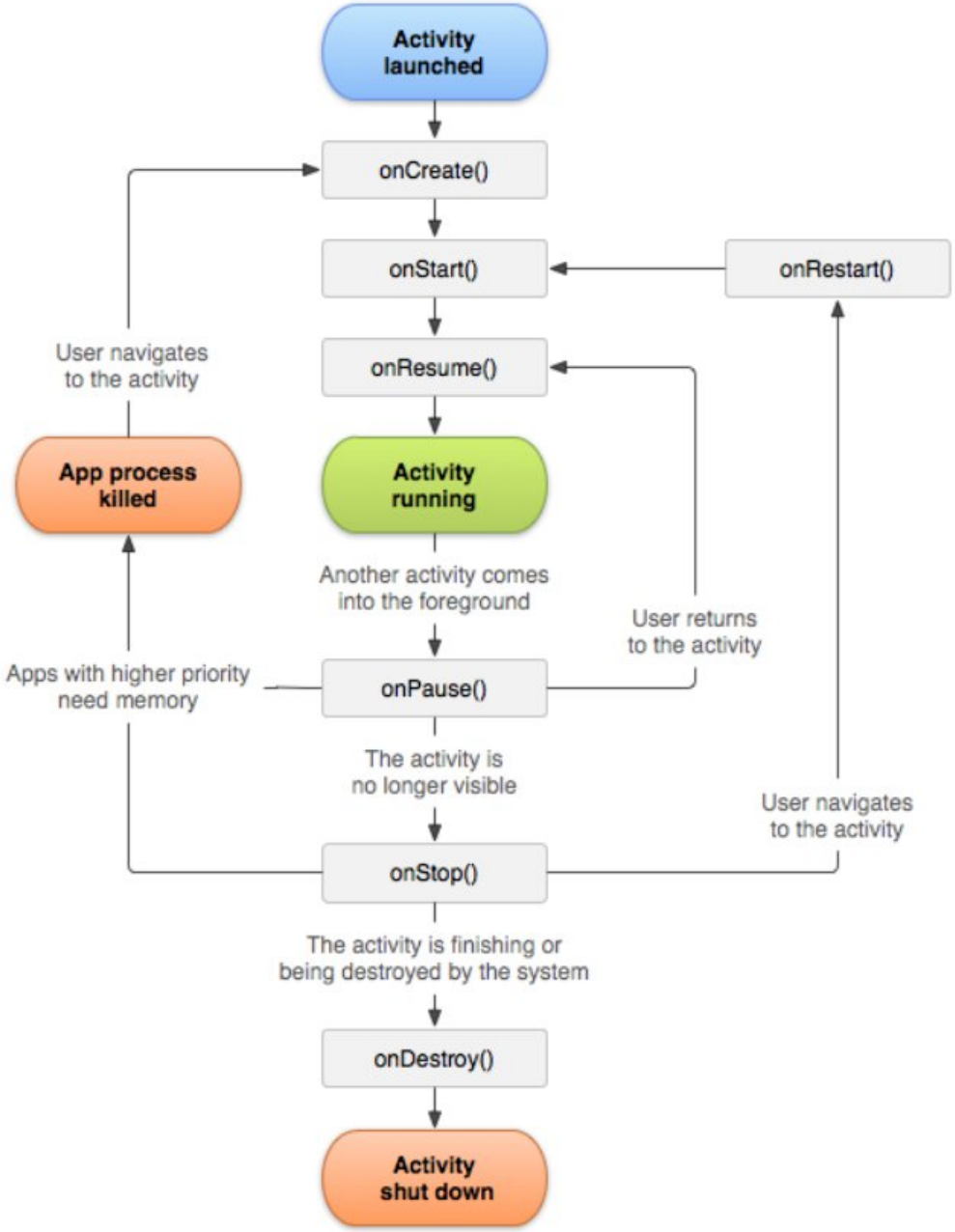
При работе приложения, пользователь создает новые Activity и закрывает старые, сворачивает приложение, снова открывает и т.д. Activity умеет обрабатывать все эти действия. Это необходимо, например, для освобождения ресурсов или сохранения данных. Созданное при работе приложения Activity может быть в одном из трех состояний:

Resumed - Activity видно на экране, оно находится в фокусе, пользователь может с ним взаимодействовать. Это состояние также иногда называют Running.

Paused - Activity не в фокусе, пользователь не может с ним взаимодействовать, но его видно (оно перекрыто другим Activity, которое занимает не весь экран или полупрозрачно).

Stopped - Activity не видно (полностью перекрывается другим Activity), соответственно оно не в фокусе и пользователь не может с ним взаимодействовать.

Когда Activity переходит из одного состояния в другое, система вызывает различные методы, которые мы можем заполнять своим кодом. Схематично это можно изобразить так:


Activity launched
onCreate()
onStart()
onRestart()
onResume()
User navigates to the activity
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

Создадим проект

```java
//Java
package com.example.les_23;

import android.os.Bundle;
import android.app.Activity;

public class MainActivity extends Activity {

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

```kotlin
//Kotlin
package com.example.les_23

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?)
{
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

Как видно уже реализован метод onCreate. Здесь важно понимать, что этот метод НЕ создает Activity. Создание – это дело системы. Т.е. система сама создает Activity, а программисту дает возможность спроектировать и выполнить свой код в методе onCreate(). В данном случае указывается, что Activity должна отобразить экран из **`R.layout.activity_main`**

Добавим все остальные методы из схемы, и в каждый добавим запись в лог.

//Java

```java
import android.util.Log;

public class MainActivity extends Activity {
final String TAG = "States";
@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
Log.d(TAG, "MainActivity: onCreate()");
}
@Override
protected void onStart() {
super.onStart();
Log.d(TAG, "MainActivity: onStart()");
}
@Override
protected void onResume() {
super.onResume();
Log.d(TAG, "MainActivity: onResume()");
}
@Override
protected void onPause() {
super.onPause();
Log.d(TAG, "MainActivity: onPause()");
}
@Override
protected void onStop() {
super.onStop();
Log.d(TAG, "MainActivity: onStop()");
}
@Override
protected void onDestroy() {
super.onDestroy();
Log.d(TAG, "MainActivity: onDestroy()");
}
}
```

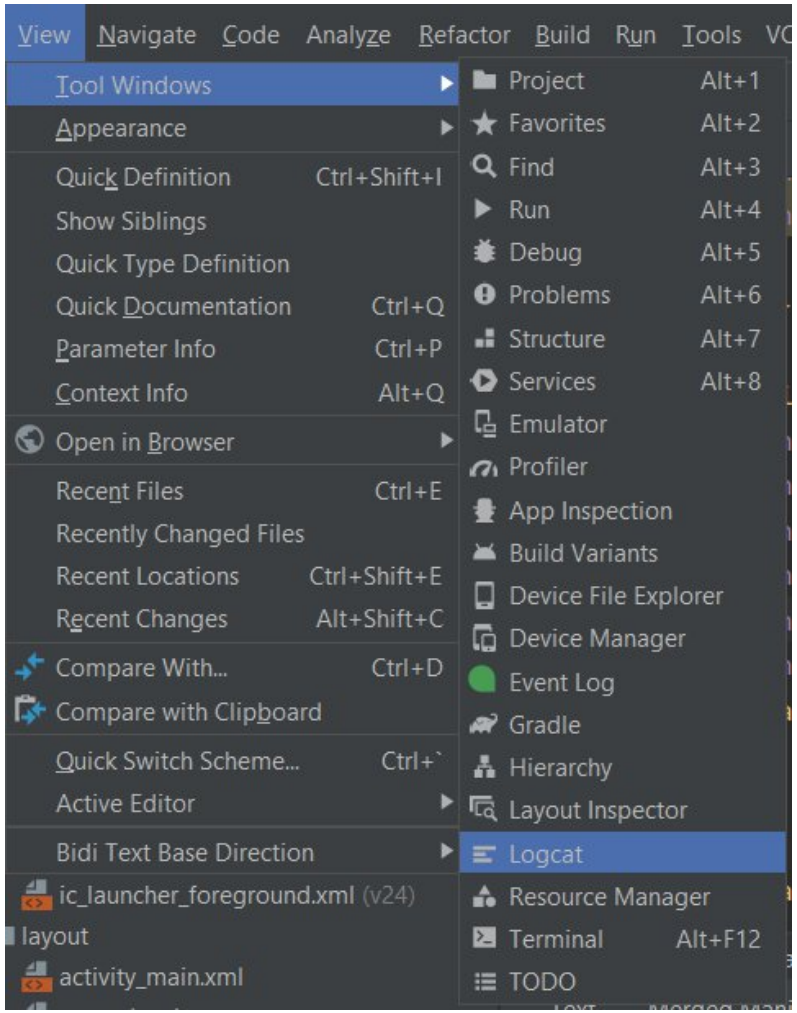
//Kotlin

```kotlin
import android.util.Log

class MainActivity : AppCompatActivity() {
    val TAG = "States"

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        Log.d(TAG, "MainActivity: onCreate()")
    }

    override fun onStart() {
        super.onStart()
        Log.d(TAG, "MainActivity: onStart()")
    }

    override fun onResume() {
        super.onResume()
        Log.d(TAG, "MainActivity: onResume()")
    }

    override fun onPause() {
        super.onPause()
        Log.d(TAG, "MainActivity: onPause()")
    }

    override fun onStop() {
        super.onStop()
        Log.d(TAG, "MainActivity: onStop()")
    }

    override fun onDestroy() {
        super.onDestroy()
        Log.d(TAG, "MainActivity: onDestroy()")
    }
}
```

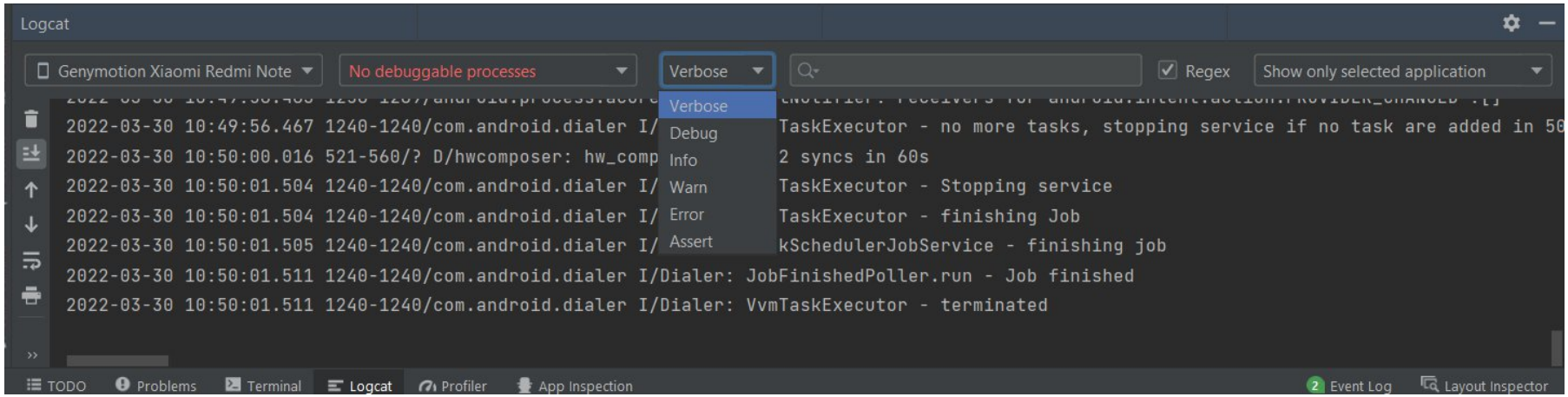
При реализации этих методов обязательно нужно вызвать соответствующие методы супер-класса и обязательно перед кодом.
Теперь, когда методы будут вызываться, это будет видно в логах. Настроим фильтр на тег «States», чтобы не искать свои сообщения в общем списке логов.

Логи приложения

Когда вы тестируете работу приложения, вы можете видеть логи работы. Они отображаются в окне Logcat. Чтобы отобразить окно откройте меню View > Tool Windows > Logcat



Должна появится вкладка LogCat



Рассмотрим эту вкладку подробней. Логи имеют разные уровни важности: ASSERT, ERROR, WARN, INFO, DEBUG, VERBOSE (по убыванию). Фильтры логов указаны в выпадающем списке. Обратите внимание, что фильтр показывает логи не только своего уровня, но и уровней более высокой важности. Также вы можете создавать, редактировать и удалять свои фильтры.

Рассмотрим создание пользовательских логов. Это можно выполнить с помощью класса Log и его методов Log.v() Log.d() Log.i() Log.w() and Log.e(). Названия методов соответствуют уровню логов, которые они запишут.

После того, как приложение запустилось, смотрим лог:

```
MainActivity: onCreate()
MainActivity: onStart()
MainActivity: onResume()
```

Activity создалось, прошло два состояния (Stopped, Paused) и теперь находится в третьем состоянии - Resumed. Т.е. оно создалось (onCreate), отобразилось (onStart) и получило возможность взаимодействовать с пользователем (onResume).

Теперь нажмем кнопку Back на эмуляторе. Activity закрылось. Смотрим лог:

```
MainActivity: onPause()
MainActivity: onStop()
MainActivity: onDestroy()
```

Activity проделывает путь, обратный созданию. Сначала теряет фокус (onPaused), затем исчезает с экрана (onStop), затем полностью уничтожается (onDestroy).

```
2022-03-30 10:54:00.056 2204-2204/com.example.myapplication D/States: MainActivity: onCreate()
2022-03-30 10:54:00.126 2204-2204/com.example.myapplication D/States: MainActivity: onStart()
2022-03-30 10:54:00.131 2204-2204/com.example.myapplication D/States: MainActivity: onResume()
2022-03-30 10:55:24.850 2204-2204/com.example.myapplication D/States: MainActivity: onPause()
2022-03-30 10:55:25.750 2204-2204/com.example.myapplication D/States: MainActivity: onStop()
2022-03-30 10:55:25.759 2204-2204/com.example.myapplication D/States: MainActivity: onDestroy()
```

Смена ориентации экрана

Рассмотрим, как ведет себя Activity, когда происходит смена ориентации экрана. Запустите снова приложение. В логах снова отобразились три метода, вызванные при создании. Теперь в эмуляторе нажмите CTRL+F12, ориентация сменилась. Рассмотрим полученные логи:

```
MainActivity: onPause()
MainActivity: onStop()
MainActivity: onDestroy()
MainActivity: onCreate()
MainActivity: onStart()
MainActivity: onResume()
```

Activity полностью уничтожается и снова создается. При этом обычно выполняются процедуры сохранения и восстановления данных, чтобы не потерялись данные, и приложение сохранило свой вид.

Также есть еще метод onRestart. Он вызывается перед методом onStart, если Activity не создается с нуля, а восстанавливается из состояния Stoped.

Итак, было рассмотрено какие состояния проходит Activity за время своего существования и какие методы при этом вызываются. Activity находилось только в состоянии Resumed (т.е. его видно, и оно в фокусе). Теперь на примере двух Activity попробуем понять, в каком случае Activity может остаться в состоянии Stopped, т.е. не видно и не в фокусе, но существует в памяти. Создадаим первое Activity с таким экраном:

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

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello" >
    </TextView>

    <Button
        android:id="@+id/btnActTwo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Go to Activity Two" >
    </Button>

</LinearLayout>
```

Откроем MainActivity.java и пишем туда все методы, включая onRestart, и в методах прописываем запись в логи. Также описываем и находим кнопку, присваиваем ей обработчик. В методе onClick пока ничего не пишем.

```
//Java
public class MainActivity extends Activity
implements View.OnClickListener {

  final String TAG = "States";
  Button btnActTwo;

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

    btnActTwo = (Button)
findViewById(R.id.btnActTwo);
    btnActTwo.setOnClickListener(this);

    Log.d(TAG, "MainActivity: onCreate()");
  }

  @Override
  protected void onRestart() {
    super.onRestart();
    Log.d(TAG, "MainActivity: onRestart()");
  }

  @Override
  protected void onStart() {
    super.onStart();
    Log.d(TAG, "MainActivity: onStart()");
  }

  @Override
  protected void onResume() {
    super.onResume();
    Log.d(TAG, "MainActivity: onResume()");
  }

  @Override
  protected void onPause() {
    super.onPause();
    Log.d(TAG, "MainActivity: onPause()");
  }

  @Override
  protected void onStop() {
    super.onStop();
    Log.d(TAG, "MainActivity: onStop()");
  }

  @Override
  protected void onDestroy() {
    super.onDestroy();
    Log.d(TAG, "MainActivity: onDestroy()");
  }

 // @Override
  public void onClick(View v) {
  }
}
```

```kotlin
//Kotlin
class MainActivity : AppCompatActivity()
{

    val TAG = "States"

    override fun
onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity_main)

btnActTwo.setOnClickListener(this::onClic
k)
        Log.d(TAG, "MainActivity:
onCreate()")
    }

    override fun onStart() {
        super.onStart()
        Log.d(TAG, "MainActivity:
onStart()")
    }

    override fun onResume() {
        super.onResume()
        Log.d(TAG, "MainActivity:
onResume()")
    }

    override fun onPause() {
        super.onPause()
        Log.d(TAG, "MainActivity:
onPause()")
    }

    override fun onStop() {
        super.onStop()
        Log.d(TAG, "MainActivity:
onStop()")
    }

    override fun onDestroy() {
        super.onDestroy()
        Log.d(TAG, "MainActivity:
onDestroy()")
    }

    fun onClick(view : View){

    }
}
```

Какие методы и в каком порядке выполняются при работе одного Activity, известно. Рассмотрим поведение при двух Activity, поэтому создаем второе Activity. Назовем ее SecondActivityTwo. Необходимо выполнить следующую последовательность действий: создать класс с таким именем и с суперклассом android.app.Activity, и прописать новое Activity в манифест-файле. Также надо создать layout-файл, назовем его two.xml и заполним этим кодом:

```
<?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/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="This is Activity Two" >
    </TextView>

</LinearLayout>
```

Создаем класс. Код SecondActivity.java:

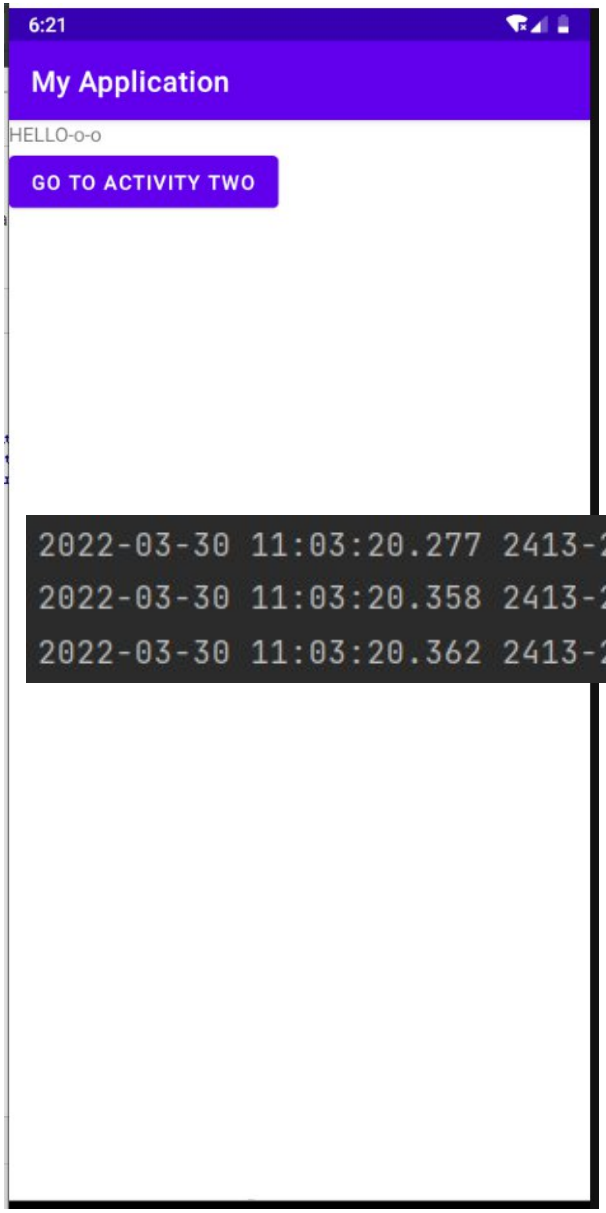
```
//Java
public class SecondActivity extends Activity {
final String TAG = "States";

  @Override
  public void onCreate(Bundle
savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_second);
    Log.d(TAG, "ActivityTwo: onCreate()");
  }

  @Override
  protected void onRestart() {
    super.onRestart();
    Log.d(TAG, "ActivityTwo: onRestart()");
  }

  @Override
  protected void onStart() {
    super.onStart();
    Log.d(TAG, "ActivityTwo: onStart()");
  }

  @Override
  protected void onResume() {
    super.onResume();
    Log.d(TAG, "ActivityTwo: onResume()");
  }

@Override
  protected void onPause() {
    super.onPause();
    Log.d(TAG, "ActivityTwo: onPause()");
  }

  @Override
  protected void onStop() {
    super.onStop();
    Log.d(TAG, "ActivityTwo: onStop()");
  }

  @Override
  protected void onDestroy() {
    super.onDestroy();
    Log.d(TAG, "ActivityTwo: onDestroy()");
  }
}
```

Меняем в MainActivity.java

```
  public void onClick(View v) {
  Intent intent = new Intent(this,
  SecondActivity.class);
  startActivity(intent);
    }
```

```
//Kotlin
class ActivityTwo : AppCompatActivity() {

    val TAG = "States"

    override fun onCreate(savedInstanceState:
Bundle?) {
        super.onCreate(savedInstanceState)

setContentView(R.layout.activity_second)

        Log.d(TAG, "ActivityTwo: onCreate()")
    }

    override fun onRestart() {
        super.onRestart()
        Log.d(TAG, "ActivityTwo:
onRestart()")
    }

    override fun onStart() {
        super.onStart()
        Log.d(TAG, "ActivityTwo: onStart()")
    }

    override fun onResume() {
        super.onResume()
        Log.d(TAG, "ActivityTwo: onResume()")
    }

    override fun onPause() {
        super.onPause()
        Log.d(TAG, "ActivityTwo: onPause()")
    }

    override fun onStop() {
        super.onStop()
        Log.d(TAG, "ActivityTwo: onStop()")
    }

    override fun onDestroy() {
        super.onDestroy()
        Log.d(TAG, "ActivityTwo:
onDestroy()")
    }

}
```

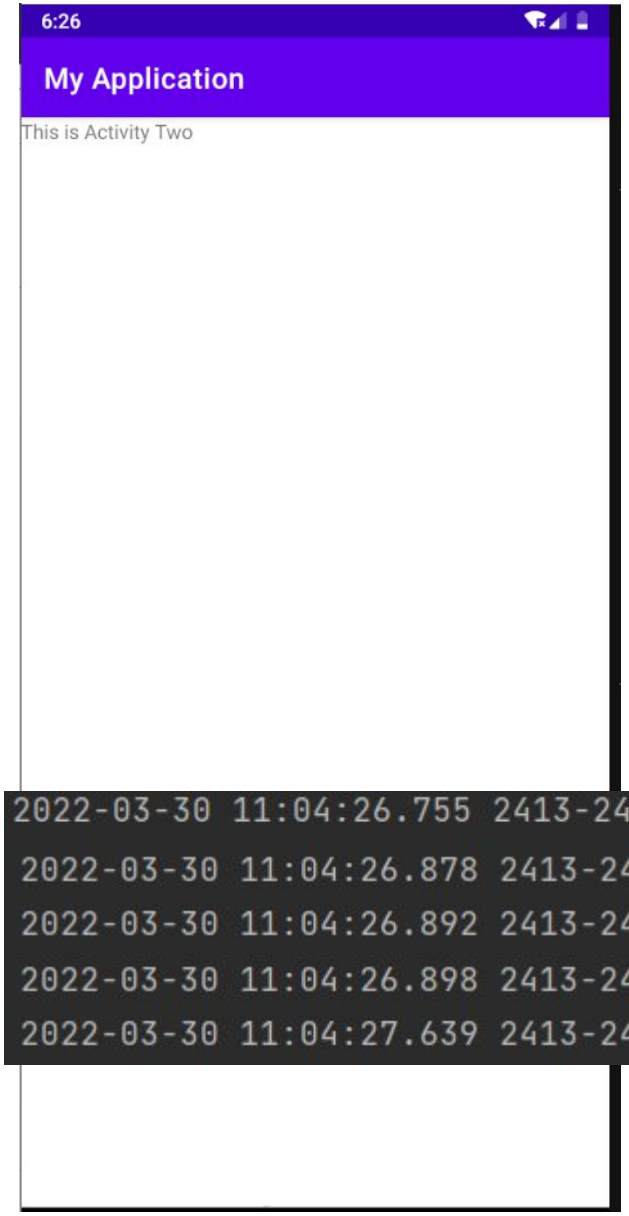
Шаг1. Запускаем приложение. Появилось MainActivity.



Вызываются три метода. Activity проходит через состояния Stopped, Paused и остается в состоянии Resumed.

```
2022-03-30 11:03:20.277 2413-2413/com.example.myapplication D/States: MainActivity: onCreate()
2022-03-30 11:03:20.358 2413-2413/com.example.myapplication D/States: MainActivity: onStart()
2022-03-30 11:03:20.362 2413-2413/com.example.myapplication D/States: MainActivity: onResume()
```

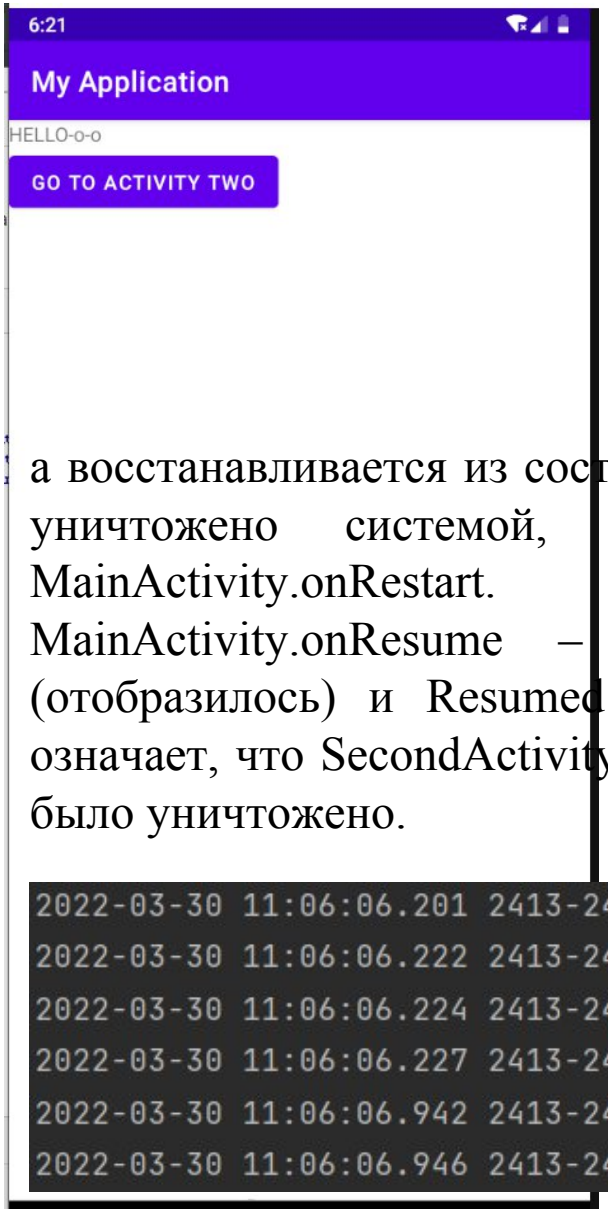
Шаг 2. Жмем кнопку «Go to Activity Two» на экране и появляется SecondActivity



Вызов MainActivity.onPause означает, что MainActivity теряет фокус и переходит в состояние Paused. Затем создается (onCreate), отображается (onStart) и получает фокус (onResume) ActivityTwo. Затем перестает быть видно (onStop) MainActivity. Обратите внимание, что не вызывается onDestroy для MainActivity, а значит, оно не уничтожается. MainActivity остается в памяти, в состоянии Stopped. А SecondActivity – находится в состоянии Resumed. Его видно и оно в фокусе, с ним можно взаимодействовать.

```
2022-03-30 11:04:26.755 2413-2413/com.example.myapplication D/States: MainActivity: onPause()
2022-03-30 11:04:26.878 2413-2413/com.example.myapplication D/States: ActivityTwo: onCreate()
2022-03-30 11:04:26.892 2413-2413/com.example.myapplication D/States: ActivityTwo: onStart()
2022-03-30 11:04:26.898 2413-2413/com.example.myapplication D/States: ActivityTwo: onResume()
2022-03-30 11:04:27.639 2413-2413/com.example.myapplication D/States: MainActivity: onStop()
```

Шаг 3. Жмем кнопку Назад (Back) на эмуляторе. Мы вернулись в MainActivity.

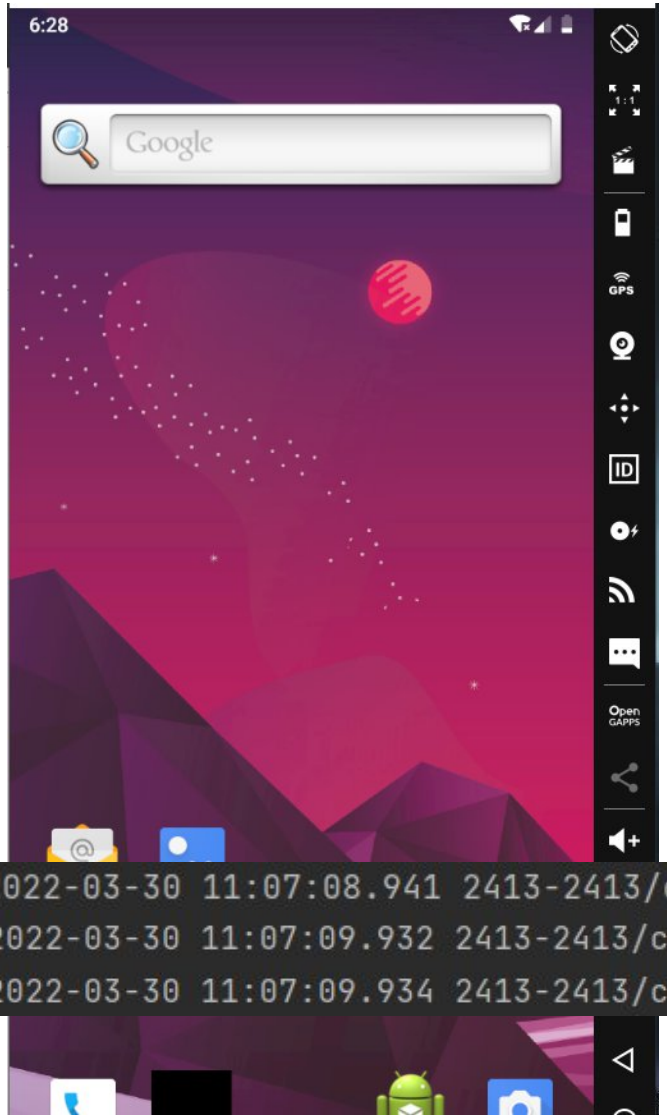


SecondActivity.onPause означает, что SecondActivity теряет фокус и переходит в состояние Paused. MainActivity теперь должна восстановиться из статуса Stopped. Метод onRestart вызывается перед методом onStart, если Activity не создается с нуля,

а восстанавливается из состояния Stoped – так происходит здесь, MainActivity не было уничтожено системой, оно находилось в памяти. Поэтому вызывается MainActivity.onRestart. Далее вызываются методы MainActivity.onStart и MainActivity.onResume – значит MainActivity перешло в состояние Paused (отобразилось) и Resumed (получило фокус). Вызов методов onStop и onDestroy означает, что SecondActivity было переведено в статус Stopped (потеряло видимость) и было уничтожено.

```
2022-03-30 11:06:06.201 2413-2413/com.example.myapplication D/States: ActivityTwo: onPause()
2022-03-30 11:06:06.222 2413-2413/com.example.myapplication D/States: MainActivity: onRestart()
2022-03-30 11:06:06.224 2413-2413/com.example.myapplication D/States: MainActivity: onStart()
2022-03-30 11:06:06.227 2413-2413/com.example.myapplication D/States: MainActivity: onResume()
2022-03-30 11:06:06.942 2413-2413/com.example.myapplication D/States: ActivityTwo: onStop()
2022-03-30 11:06:06.946 2413-2413/com.example.myapplication D/States: ActivityTwo: onDestroy()
```

Шаг 4. Жмем еще раз Назад и приложение закрылось.




```
2022-03-30 11:07:08.941 2413-2413/com.example.myapplication D/States: MainActivity: onPause()
2022-03-30 11:07:09.932 2413-2413/com.example.myapplication D/States: MainActivity: onStop()
2022-03-30 11:07:09.934 2413-2413/com.example.myapplication D/States: MainActivity: onDestroy()
```

Логи показывают, что MainActivity перешло в состояние Paused, Stopped и было уничтожено.

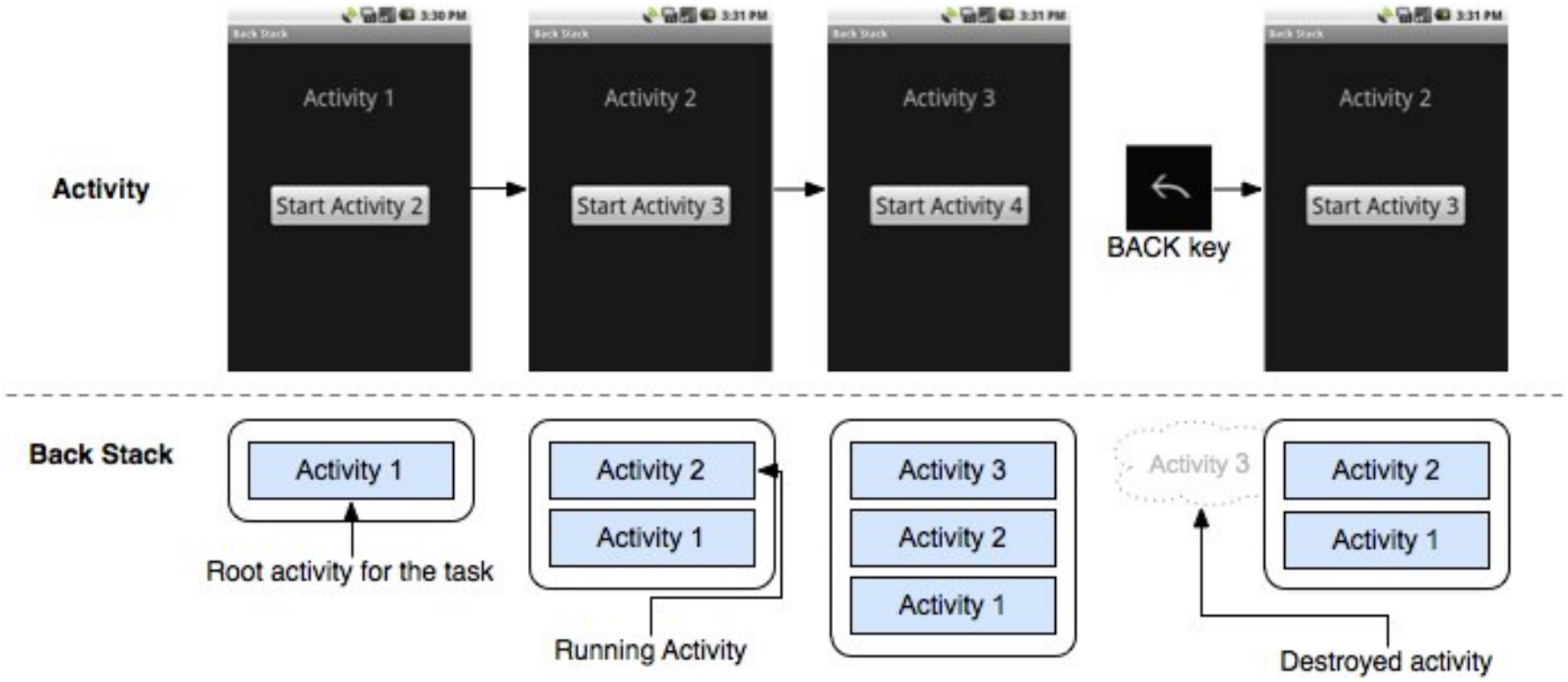
Task – группа из нескольких Activity, с помощью которых пользователь выполняет определенную операцию. Обычно стартовая позиция для создания Task – это экран Домой (Home).

Находясь в Home вы вызываете какое-либо приложение из списка приложений или через ярлык. Создается Task. И Activity приложения (которое отмечено как MAIN в манифест-файле) помещается в этот Task как корневое. Task выходит на передний фон. Если же при вызове приложения, система обнаружила, что в фоне уже существует Task, соответствующий этому приложению, то она выведет его на передний план и создавать ничего не будет.

Когда Activity_A вызывает Activity_B, то Activity_B помещается на верх Task и получает фокус. Activity_A остается в Task, но находится в состоянии Stopped (его не видно и оно не в фокусе). Далее, если пользователь жмет Back находясь в Activity_B, то Activity_B удаляется из Task и уничтожается. А Activity_A оказывается теперь на верху Task и получает фокус.

В каком порядке открывались (добавлялись в Task) Activity, в таком порядке они и содержатся в Task. Они никак специально не сортируются и не упорядочиваются внутри. Набор Activity в Task еще называют back stack.

Схема демонстрирует пример:



В верхней части то, что видит пользователь. В нижней – содержимое Task. Видно, как при вызове новых Activity они добавляются в верх стэка. А если нажата кнопка Назад, то верхнее Activity из стэка удаляется и отображается предыдущее Activity.

Предположим есть Task с несколькими Activity. Он на переднем фоне, мы с ним работаем сейчас.
Если нажать кнопку Home, то ничего не будет удалено, все Activity сохранятся в этом Task-е, а сам Task просто уйдет на задний фон и его всегда можно будет вызвать оттуда, снова вызвав приложение, Activity которого является корневым для Task-а. Либо можно удерживать кнопку Home и мы увидим как раз список Task-ов, которые расположены на заднем фоне.
Если же в активном Task-е несколько раз нажимать кнопку Назад, то в итоге в стэке не останется Activity, пустой Task будет удален и пользователь увидит экран Home.

Рассмотрим почему на шаге 2 MainActivity исчезло с экрана, но осталось висеть в памяти и не было уничтожено? Ведь на шаге 3 было уничтожено ActivityTwo после того, как оно пропало с экрана. А на шаге 4 было в итоге уничтожено и MainActivity. Почему шаг 2 стал исключением?
Ответ на этот вопрос следующий: на шаге 2 MainActivity осталось в стэке, а ActivityTwo вставилось на верх стэка и получило фокус. Ну а на шаге 3 и 4 были удалены Activity из верха стэка, в Task не осталось Activity, и мы увидели экран Home.

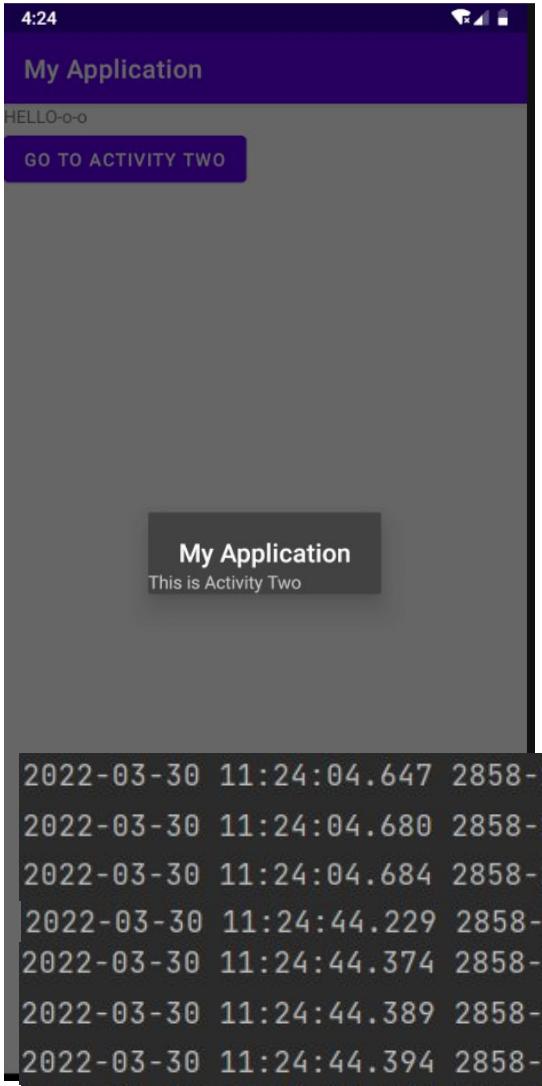
Если бы на шаге 3 нажали не Back, а Home, то Task с обоими Activity ушел бы на задний фон и ничего не было бы уничтожено.

Состояние Paused

Это состояние означает, что Activity не в фокусе, но оно видно, пусть и частично. Этого можно добиться, если присвоить диалоговый стиль для SecondActivity. Оно отобразится как всплывающее окно и под ним будет частично видно MainActivity – оно и будет в статусе Paused.

Для этого открываем AndroidManifest.xml, в пространстве второй активности указываем свойство android:theme="@style/Theme.AppCompat.Dialog"

```
<activity
    android:name=".ActivityTwo"
    android:exported="false"
    android:theme="@style/Theme.AppCompat.Dialog"/>
```



После запуска приложения появилось MainActivity
Логи:

```
MainActivity: onCreate()
MainActivity: onStart()
MainActivity: onResume()
```

Вызываем ActivityTwo.
Логи:

```
MainActivity: onPause()
ActivityTwo: onCreate()
ActivityTwo: onStart()
ActivityTwo: onResume()
```

Видим, что не был вызван метод onStop для MainActivity, а значит приложение не было переведено в состояние Stopped и находится в режиме Paused.

```
2022-03-30 11:24:04.647 2858-2858/com.example.myapplication D/States: MainActivity: onCreate()
2022-03-30 11:24:04.680 2858-2858/com.example.myapplication D/States: MainActivity: onStart()
2022-03-30 11:24:04.684 2858-2858/com.example.myapplication D/States: MainActivity: onResume()
2022-03-30 11:24:44.229 2858-2858/com.example.myapplication D/States: MainActivity: onPause()
2022-03-30 11:24:44.374 2858-2858/com.example.myapplication D/States: ActivityTwo: onCreate()
2022-03-30 11:24:44.389 2858-2858/com.example.myapplication D/States: ActivityTwo: onStart()
2022-03-30 11:24:44.394 2858-2858/com.example.myapplication D/States: ActivityTwo: onResume()
2022-03-30 11:26:39.349 2858-2858/com.example.myapplication D/States: ActivityTwo: onPause()
2022-03-30 11:26:39.427 2858-2858/com.example.myapplication D/States: MainActivity: onResume()
2022-03-30 11:26:39.471 2858-2858/com.example.myapplication D/States: ActivityTwo: onStop()
2022-03-30 11:26:39.480 2858-2858/com.example.myapplication D/States: ActivityTwo: onDestroy()
```

Нажмем Back.

Логи:

```
ActivityTwo: onPause()
MainActivity: onResume()
ActivityTwo: onStop()
ActivityTwo: onDestroy()
```

MainActivity восстановилось одним лишь вызовом onResume, а onStart не понадобился, т.к. оно было в состоянии Paused, а не Stopped.

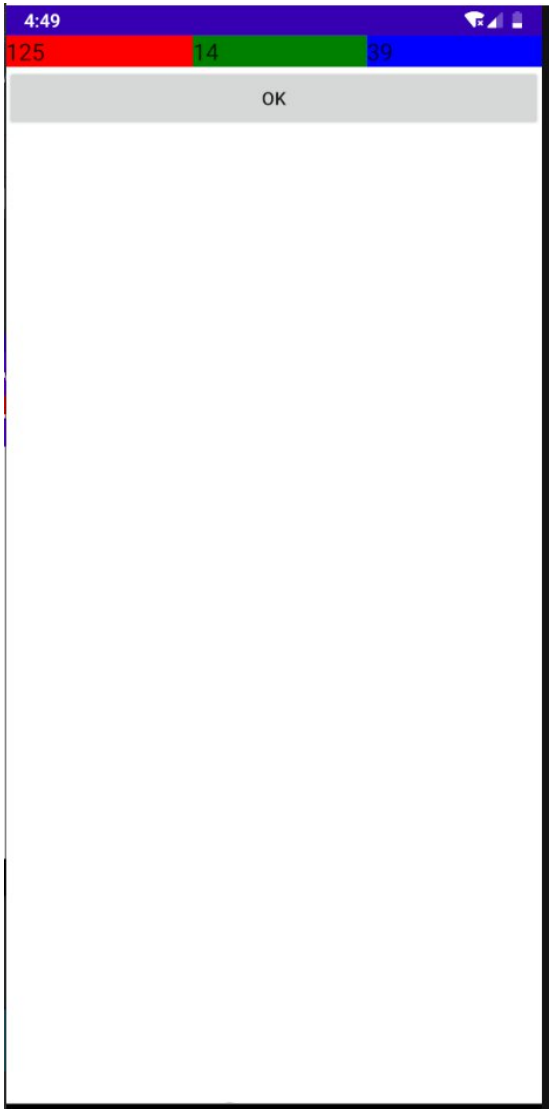
Далее можно нажать Back, а можно Home.

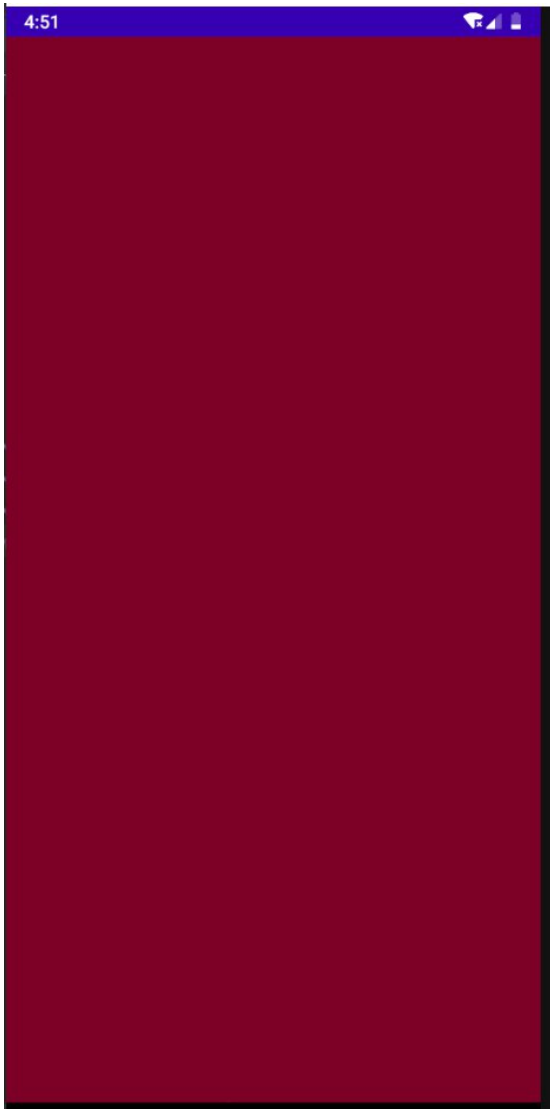
Чтобы вернуть ActivityTwo нормальный режим отображения, зайдите снова в манифест и удалите строку из поля Theme.

```
2022-03-30 11:29:19.062 2858-2858/com.example.myapplication D/States: MainActivity: onPause()
2022-03-30 11:29:19.982 2858-2858/com.example.myapplication D/States: MainActivity: onStop()
2022-03-30 11:29:19.985 2858-2858/com.example.myapplication D/States: MainActivity: onDestroy()
```

Рассмотрим пример простого приложения.

В первом проекте пользователю необходимо в EditText задать компоненты цвета RGB. При нажатии на кнопку меняется фон TextView на другой Activity.





Activity_main.xml

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/an
droid"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:id="@+id/lo" >


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

        <EditText
            android:id="@+id/editText1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:background="@color/red"
            android:hint="R" ></EditText>

        <EditText
            android:id="@+id/editText2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:background="@color/green"
            android:hint="G" ></EditText>
        <EditText
            android:id="@+id/editText3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:background="@color/blue"
            android:hint="B" ></EditText>
    </LinearLayout>

    <LinearLayout

        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal" >

        <Button
            android:id="@+id/button1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Ok" >

        </Button>
    </LinearLayout>

</LinearLayout>
```

Second.xml

```
<?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/textView1"
        android:layout_width="match_parent"
        android:layout_height="match_parent">
    </TextView>

</LinearLayout>
```

MainActivity.java

```
//Java
package com.example.laba4part1;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.graphics.Color;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class MainActivity extends Activity implements
OnClickListener {


 Button b1;  EditText et1;  EditText et2;  EditText et3;  TextView
tv;   Color c;

@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(com.example.laba4part1.R.layout.activity_main);
b1 = (Button) findViewById(com.example.laba4part1.R.id.button1);
b1.setOnClickListener(this);
et1 = (EditText)
findViewById(com.example.laba4part1.R.id.editText1);
et2 = (EditText)
findViewById(com.example.laba4part1.R.id.editText2);
et3 = (EditText)
findViewById(com.example.laba4part1.R.id.editText3);
tv = (TextView)
findViewById(com.example.laba4part1.R.id.textView10);}

public void onClick(View v)       {
Intent intent=new Intent(this, ActivityTwo.class);
intent.putExtra("R", et1.getText().toString());
intent.putExtra("G", et2.getText().toString());
intent.putExtra("B", et3.getText().toString());
 switch (v.getId())     {
case R.id.button1:
```

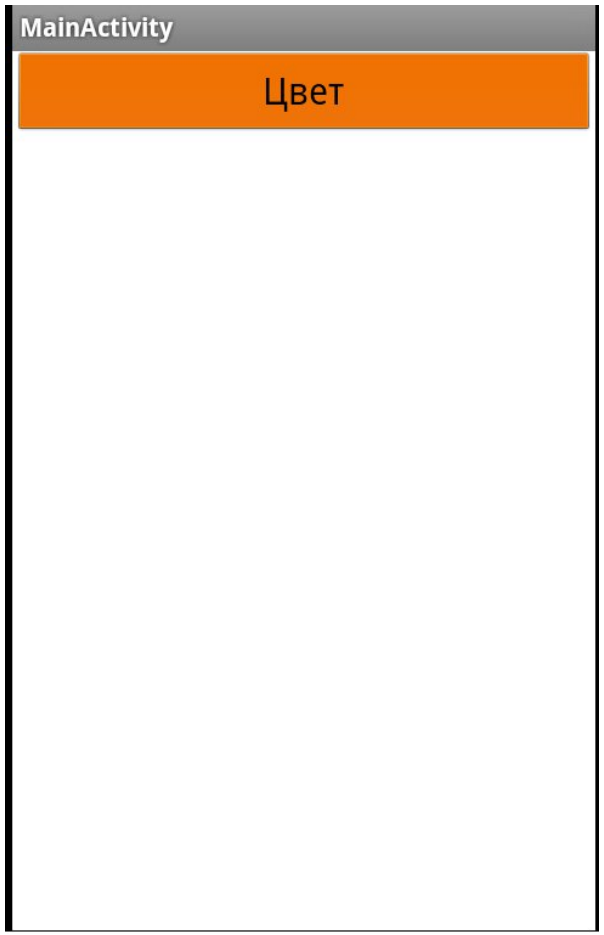
```kotlin
//Kotlin
package com.example.laba4part1

import android.content.Intent
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.EditText
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        button1.setOnClickListener(this::onClick)
    }

    fun onClick(view : View){
        var et1 = findViewById(R.id.editText1) as EditText
        var et2 = findViewById(R.id.editText2) as EditText
        var et3 = findViewById(R.id.editText3) as EditText
        var intent = Intent(this@MainActivity, ActivityTwo::class.java)
        intent.putExtra("R", et1.getText().toString())
        intent.putExtra("G", et2.getText().toString())
        intent.putExtra("B", et3.getText().toString())
        when (view.getId()){
            R.id.button1 -> {
                intent.putExtra("s", 40)
            }
        }
        startActivity(intent)
    }
}
```

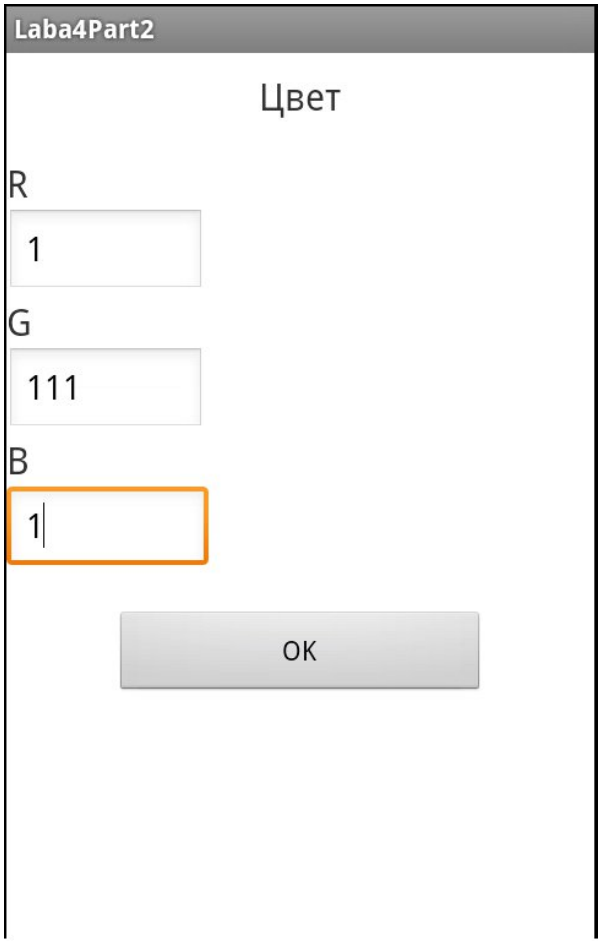
ActivityTwo.java

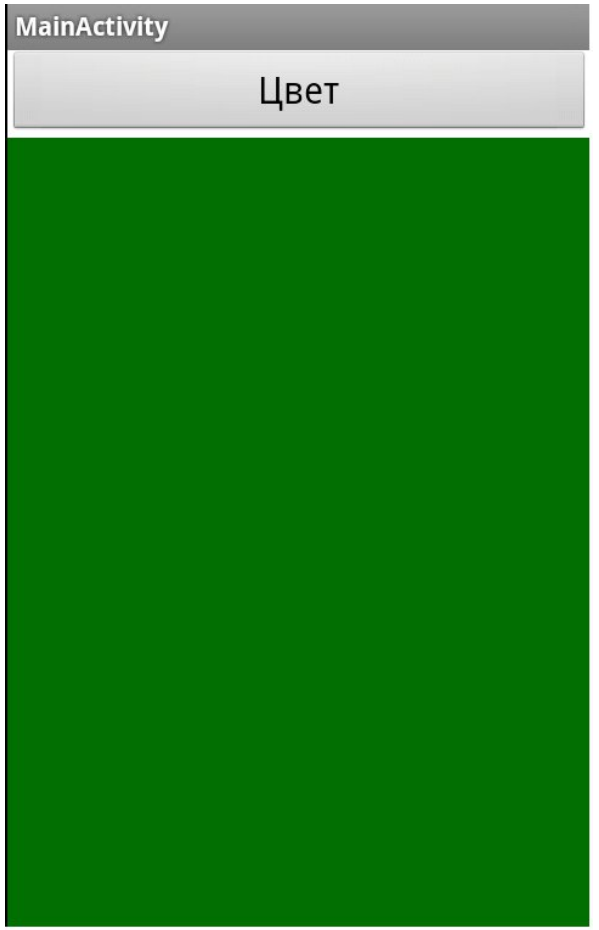
```
//Java
package com.example.laba4part1;
import android.app.Activity;
import android.content.Intent;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

public class ActivityTwo extends Activity {

     TextView tv;   Color c;


@Override
protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.second);
                Intent intent=getIntent();


                tv=(TextView)findViewById(R.id.textView10);


                String a = intent.getStringExtra("a");
                String col1 = intent.getStringExtra("R");
                String col2 = intent.getStringExtra("G");
                String col3 = intent.getStringExtra("B");
                String size = intent.getStringExtra("s");



                int co1=Integer.parseInt(col1);
                int co2=Integer.parseInt(col2);
                int co3=Integer.parseInt(col3);


                tv.setBackgroundColor(c.rgb(co1, co2, co3));
```

```kotlin
//Kotlin
package com.example.laba4part1

import android.graphics.Color
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_main.*

class ActivityTwo : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_second)

        var col1 = intent.getStringExtra("R")
        var col2 = intent.getStringExtra("G")
        var col3 = intent.getStringExtra("B")
        var size = intent.getStringExtra("s")

        var co1 = Integer.parseInt(col1)
        var co2 = Integer.parseInt(col2)
        var co3 = Integer.parseInt(col3)

        tv.setBackgroundColor(Color.rgb(co1, co2, co3))
        tv.setTextSize(size.toFloat())
    }
}
```

Во втором проекте пользователю необходимо нажать на кнопку «Цвет», чтобы задать в EditText компоненты цвета RGB. Затем, нажав на кнопку «ОК», в изначальной Activity меняется цвет TextView.







MainActivity.java

//Java

```
package com.example.laba4part2;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.graphics.Color;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends Activity implements OnClickListener {


Intent intent; TextView tv; Button b1; Button b2; Button b3;  Color c;

@Override
public void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 setContentView(R.layout.activity_main);
 tv = (TextView) findViewById(R.id.textView1500);
 b2 = (Button) findViewById(R.id.button2);
 b2.setOnClickListener(this);            }

public void onClick (View v)     {
switch (v.getId())         {
case (R.id.button2):
intent = new Intent ("col");
startActivityForResult(intent, 2);
break;                 }    }

@Override    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
if (data == null) {return;}
switch (requestCode)    {
case 2:
int n1=Integer.parseInt(data.getStringExtra("Col1"));
int n2=Integer.parseInt(data.getStringExtra("Col2"));
int n3=Integer.parseInt(data.getStringExtra("Col3"));
tv.setBackgroundColor(c.rgb(n1, n2, n3));
break;                 }                                    }}
```

```kotlin
//Kotlin
package com.example.laba4part2

import android.content.Intent
import android.graphics.Color
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.TextView
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        button2.setOnClickListener(this::onClick)
    }

    fun onClick(view : View){
        when (view.getId()){
            R.id.button2 -> {
                var intent = Intent(this@MainActivity, ActivityTwo::class.java)
                startActivityForResult(intent, 2)
            }
        }
    }

    override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
        if (data == null){
            return
        }
        when (requestCode){
            2 -> {
                var n1 = Integer.parseInt(data.getStringExtra("Col1"))
                var n2 = Integer.parseInt(data.getStringExtra("Col2"))
                var n3 = Integer.parseInt(data.getStringExtra("Col3"))
                var tv = findViewById(R.id.textView1500) as TextView
                tv.setBackgroundColor(Color.rgb(n1, n2, n3))
            }
        }
    }
}
```

ActivityTwo.java

//Java

```
package com.example.laba4part2;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;

public class ActivityThird extends Activity implements OnClickListener {


    Intent intent = new Intent();
    EditText et1;
    EditText et2;
    EditText et3;
    Button res2;


    @Override
  public void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.three);


  res2 = (Button) findViewById(R.id.button100);
  res2.setOnClickListener(this);


  et1 = (EditText) findViewById(R.id.editText100);
  et2 = (EditText) findViewById(R.id.editText200);
  et3 = (EditText) findViewById(R.id.editText300); }


    public void onClick (View v) {


    intent.putExtra("Col1", et1.getText().toString());
    intent.putExtra("Col2", et2.getText().toString());
    intent.putExtra("Col3", et3.getText().toString());
```

```kotlin
//Kotlin
package com.example.laba4part2

import android.os.Bundle
import android.view.View
import android.widget.EditText
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_main.*
import kotlinx.android.synthetic.main.second_activity.*

class ActivityTwo : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.second_activity)
        button100.setOnClickListener(this::onClick)
    }

    fun onClick(view : View){
        var et1 = findViewById(R.id.editText100) as EditText
        var et2 = findViewById(R.id.editText200) as EditText
        var et3 = findViewById(R.id.editText300) as EditText

        intent.putExtra("Col1", et1.getText().toString());
        intent.putExtra("Col2", et2.getText().toString());
        intent.putExtra("Col3", et3.getText().toString());

        setResult(RESULT_OK, intent);
        this.finish();

    }
}
```