Эргономика мобильных приложений

Преподаватель: **Меженная Марина Михайловна**

к.т.н., доцент,
доцент кафедры инженерной психологии и эргономики
а 609-2
mezhennaya@bsuir.by

Лекция 6:
Жизненный цикл Activity.
Передача данных с помощью Intent.

1. Жизненный цикл Activity.

2. Понятие стэка: Task.

3. Передача данных с помощью Intent. Методы putExtra, getExtra.

4. Метод startActivityForResult.

5. Хранение данных в виде пары: имя, значение с использованием класса SharedPreferences.

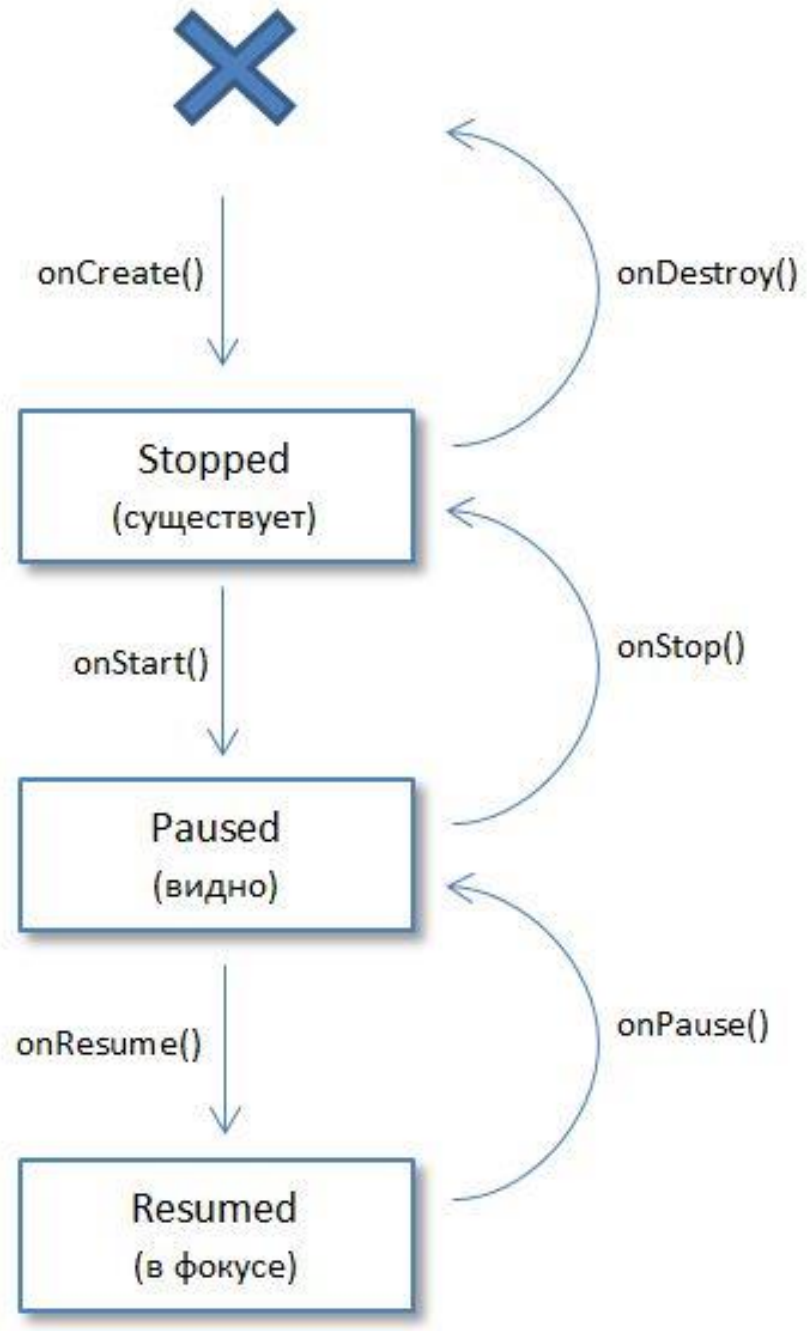
Жизненный цикл Activity

Три возможных состояния Activity:

Stopped - Activity не в фокусе и пользователь не может с ним взаимодействовать.

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).

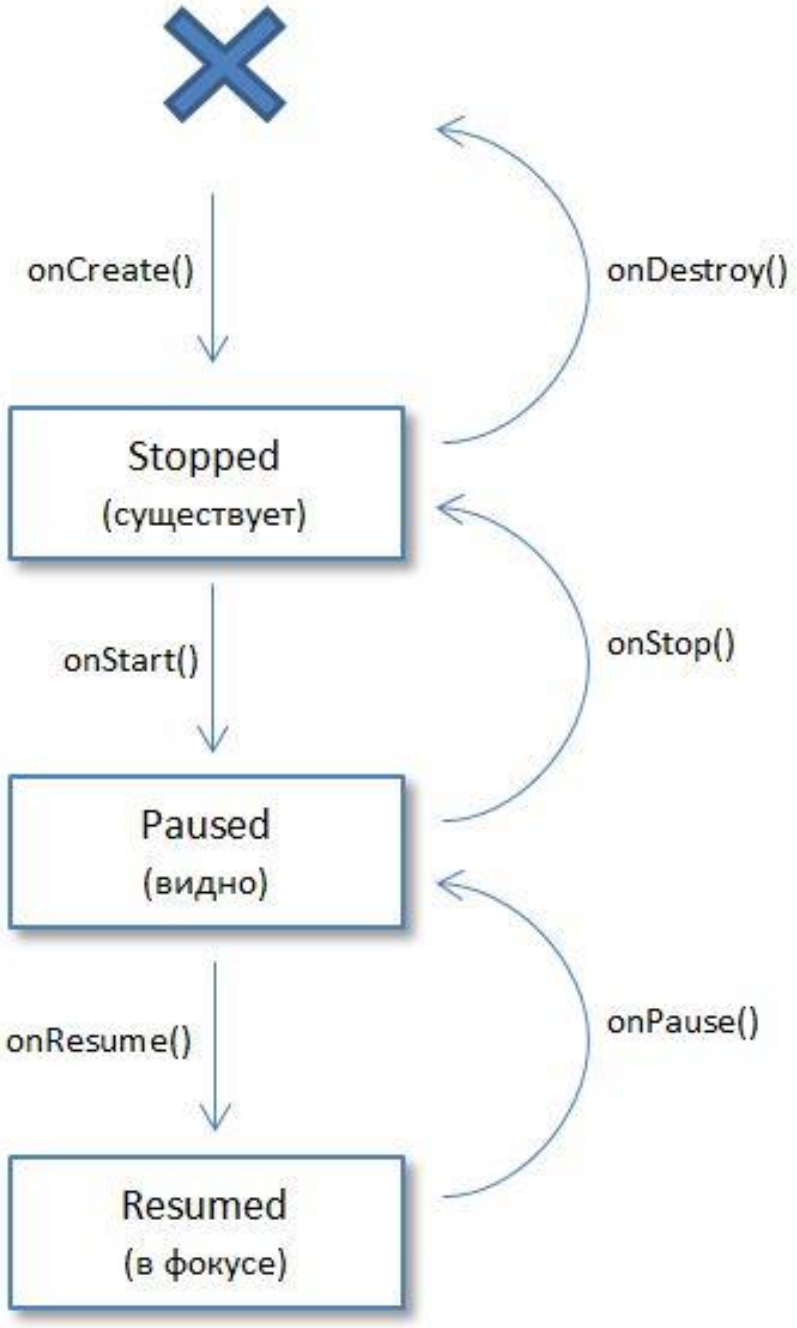
Resumed - Activity находится в фокусе, пользователь может с ним взаимодействовать.



Жизненный цикл Activity

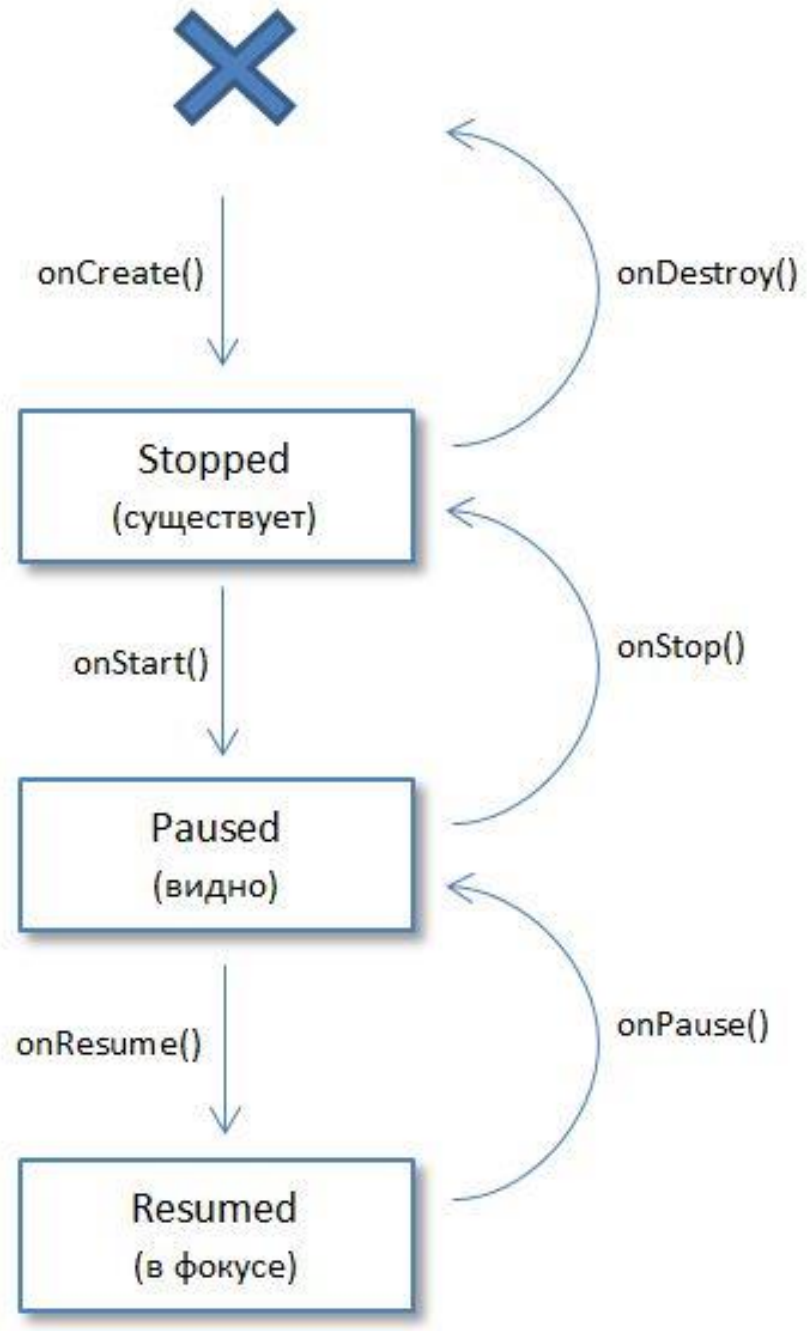
Когда Activity переходит из одного состояния в другое, система вызывает различные **методы**, которые можно заполнять своим кодом.

Эти методы **НЕ** вызывают смену состояния. Наоборот, смена состояния Activity является триггером, который вызывает эти методы.



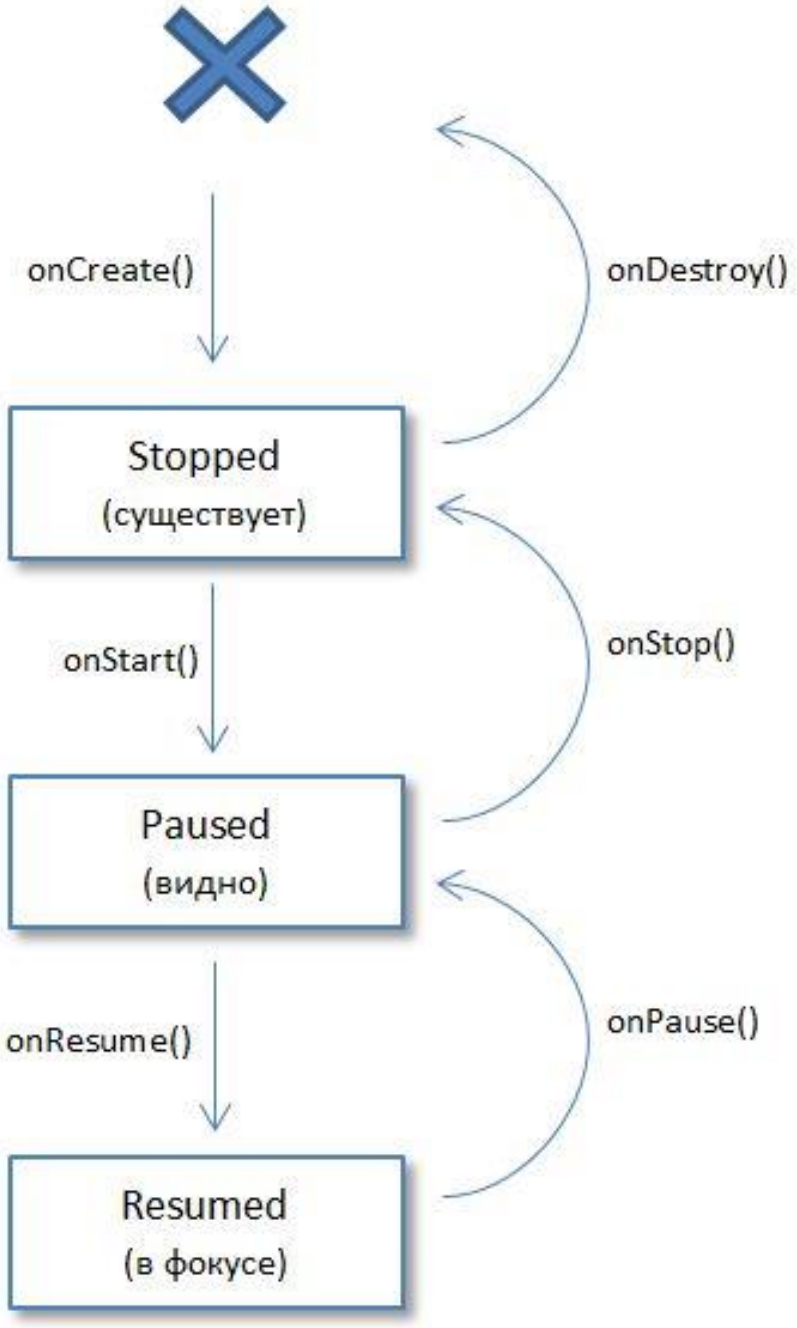
Жизненный цикл Activity

onCreate() – вызывается при первом создании Activity,
onRestart () *– вызывается перед методом onStart, если Activity не создается с нуля, а восстанавливается из состояния Stoped.*
onStart() – вызывается перед тем, как Activity будет видно пользователю,
onResume() – вызывается перед тем как будет доступно для активности пользователя.



Жизненный цикл Activity

onPause() – вызывается перед тем, как будет показано другое Activity,
onStop() – вызывается когда Activity, становится не видно пользователю
onDestroy() – вызывается перед тем, как Activity будет уничтожено.



Жизненный цикл Activity

Чтобы переопределить вышеописанные методы ЖЦ Activity:
Ctrl + O

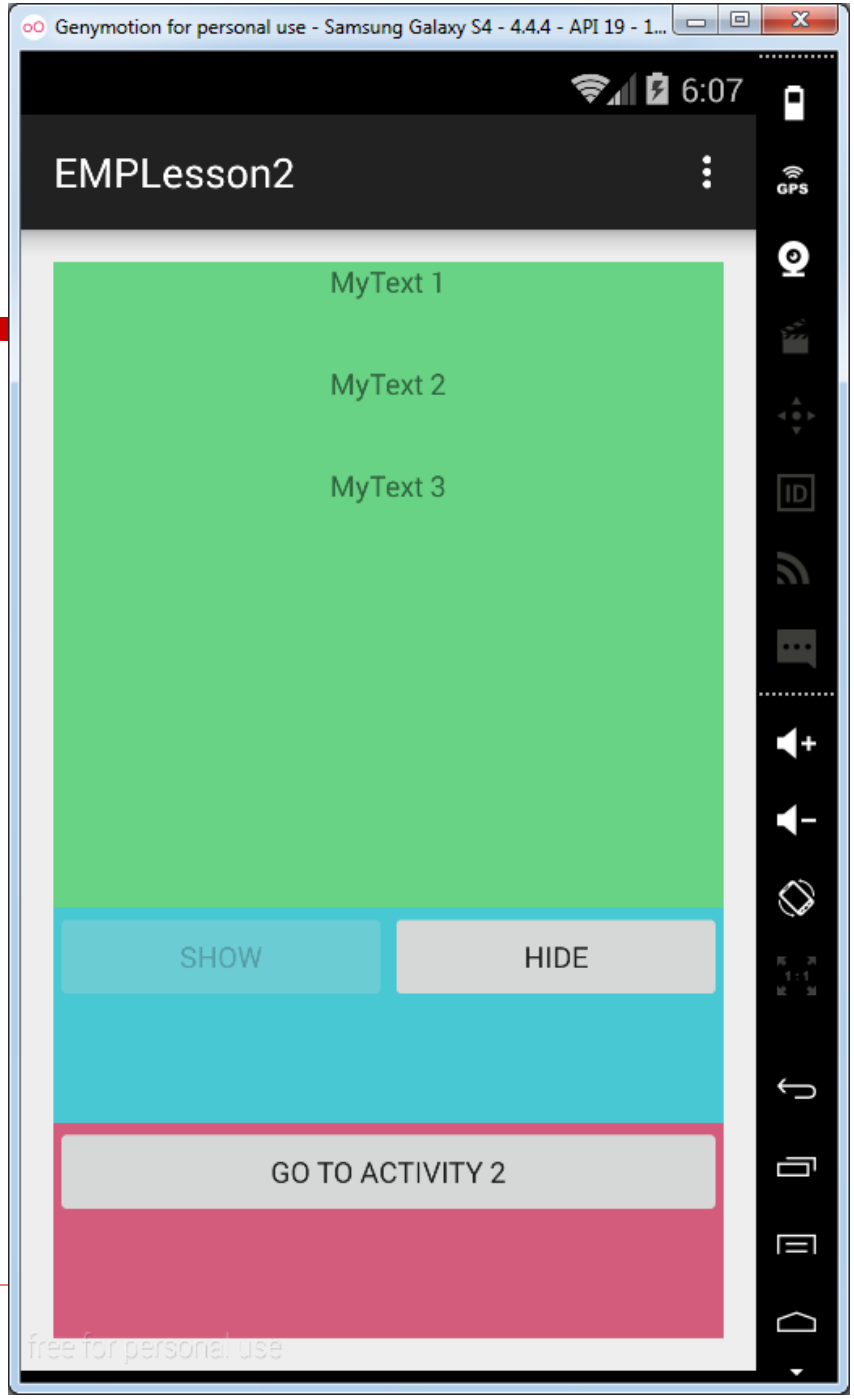
```
@Override
  protected void onStart() {
      super.onStart();
      L.d("MainActivity: onStart");      // добавляем логи
  }

  @Override
  protected void onResume() {
      super.onResume();
      L.d("MainActivity: onResume"); // добавляем логи
  }
```

и т.д. (всего 6 методов, включая onCreate())

Жизненный цикл Activity

Запустили приложение:
MainActivity: onCreate()
MainActivity: onStart()
MainActivity: onResume()

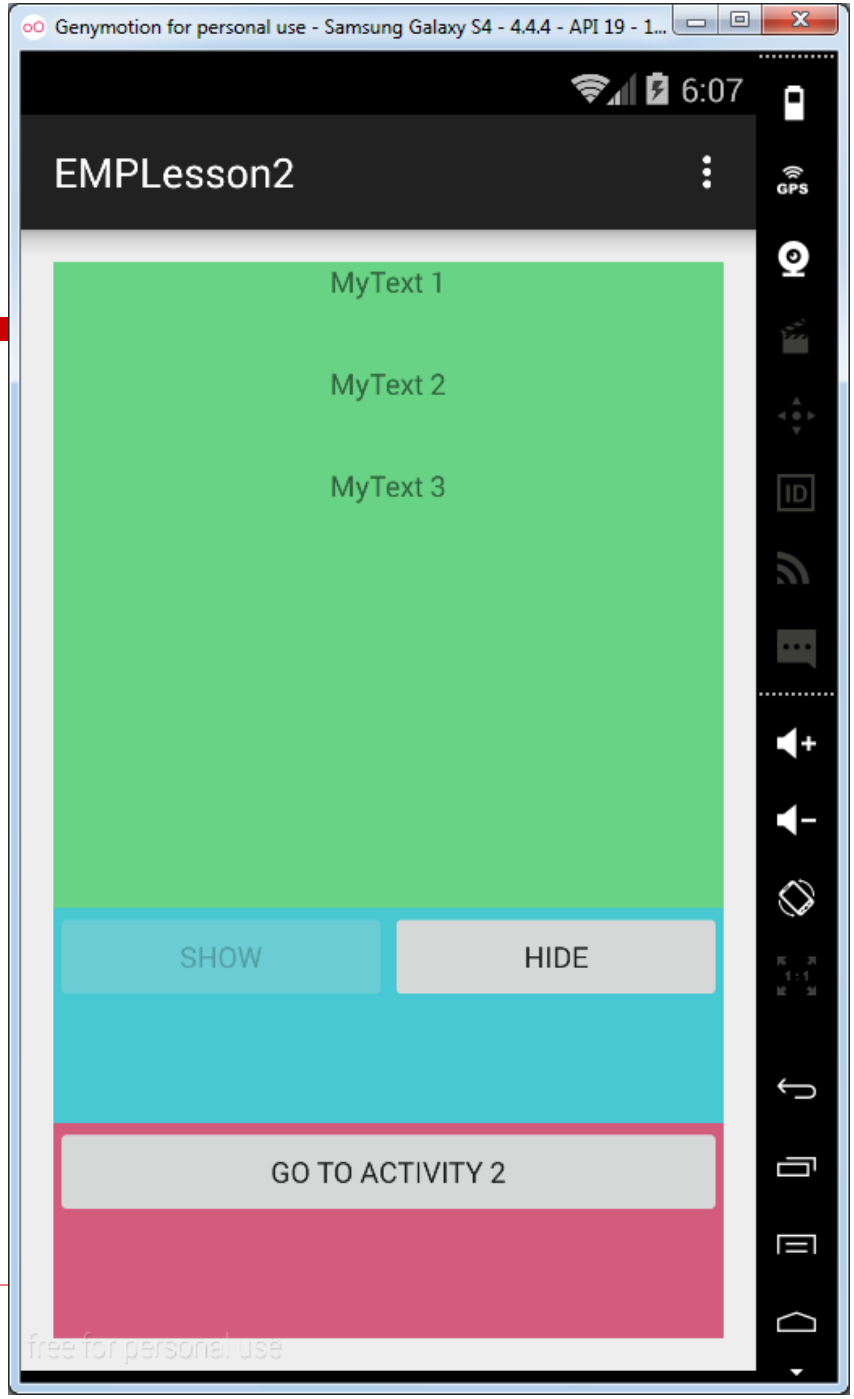
Нажали Back:
MainActivity: onPause()
MainActivity: onStop()
MainActivity: onDestroy()



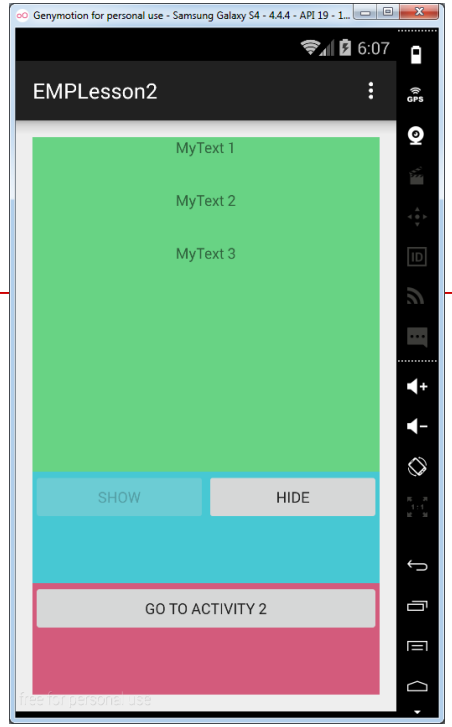
Жизненный цикл Activity

Запустили приложение:
MainActivity: onCreate()
MainActivity: onStart()
MainActivity: onResume()

Нажали Home:
MainActivity: onPause()
MainActivity: onStop()
(Activity поместили в Task)



Жизненный цикл Activity

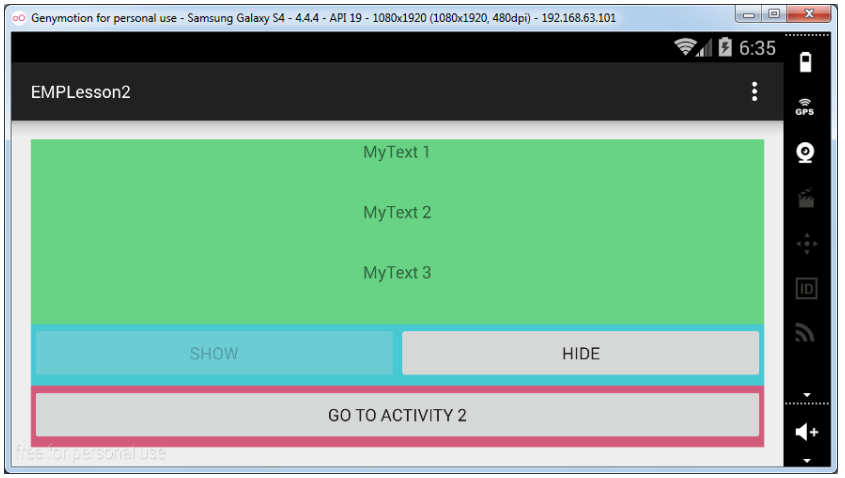


Запустили приложение:

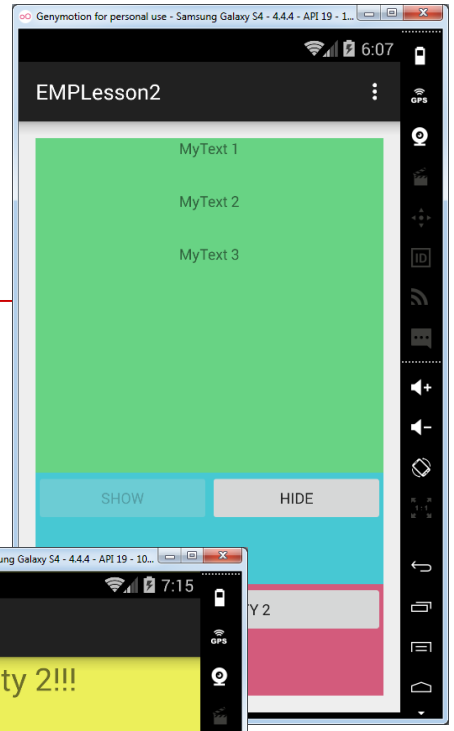
MainActivity: onCreate()
MainActivity: onStart()
MainActivity: onResume()

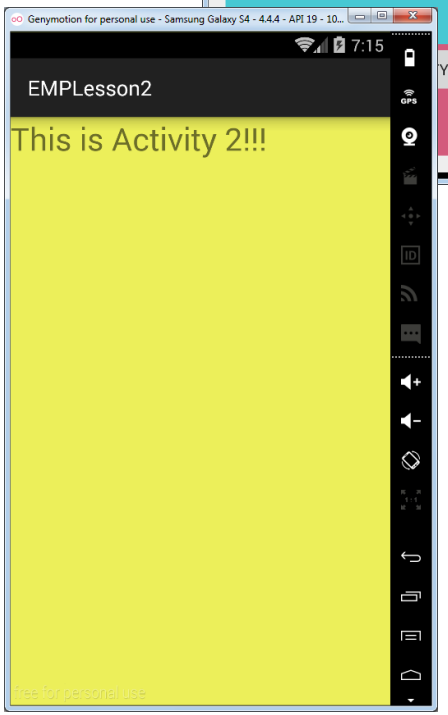
Поменяли ориентацию экрана:

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



Жизненный цикл Activity

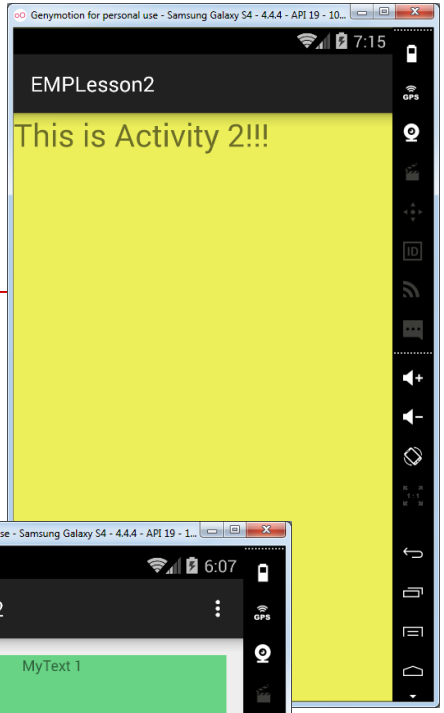


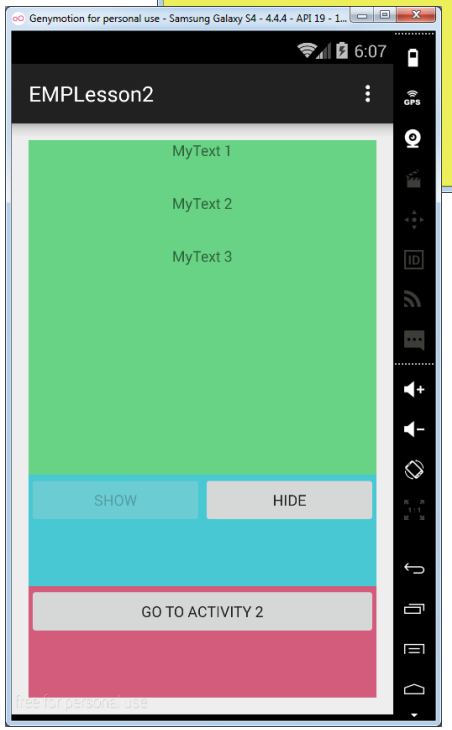


Запустили приложение:
MainActivity: onCreate()
MainActivity: onStart()
MainActivity: onResume()

Перешли ко второму Activity:
MainActivity: onPause()
SecondActivity: onCreate()
SecondActivity: onStart()
SecondActivity: onResume()
MainActivity: onStop()

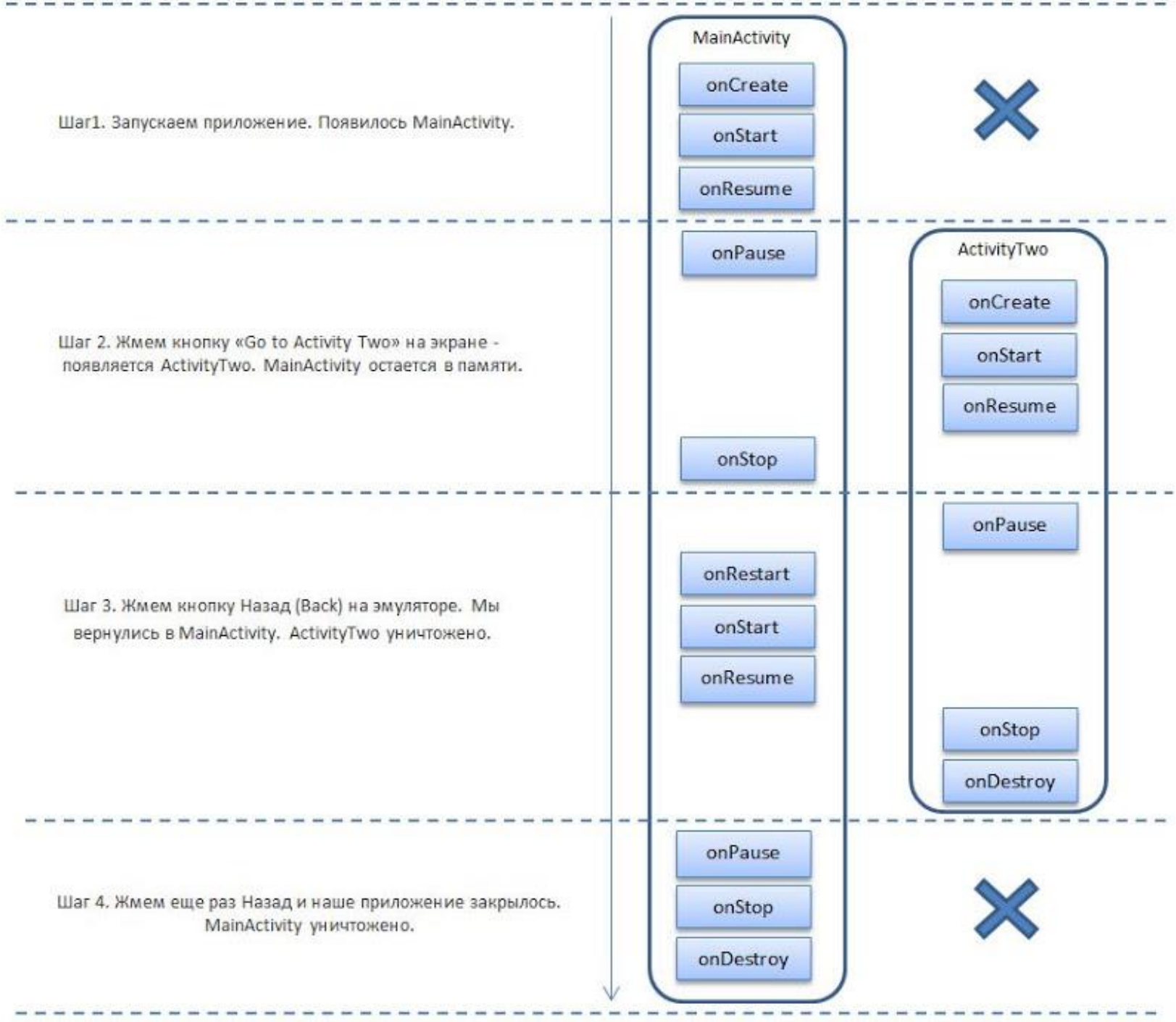
Жизненный цикл Activity





Возвращаемся «Назад»:
SecondActivity: onPause()
MainActivity: **onRestart()**
MainActivity: onStart()
MainActivity: onResume()
SecondActivity: onStop()
SecondActivity: onDestroy()

И снова «Назад»:
MainActivity: onPause()
MainActivity: onStop()
MainActivity: onDestroy()


MainActivity
onCreate
onStart
onResume
Шаг1. Запускаем приложение. Появилось MainActivity.
onPause
ActivityTwo
onCreate
onStart
onResume
Шаг 2. Жмем кнопку «Go to Activity Two» на экране - появляется ActivityTwo. MainActivity остается в памяти.
onStop
onPause
onRestart
onStart
onResume
Шаг 3. Жмем кнопку Назад (Back) на эмуляторе. Мы вернулись в MainActivity. ActivityTwo уничтожено.
onStop
onDestroy
onPause
onStop
onDestroy
Шаг 4. Жмем еще раз Назад и наше приложение закрылось. MainActivity уничтожено.

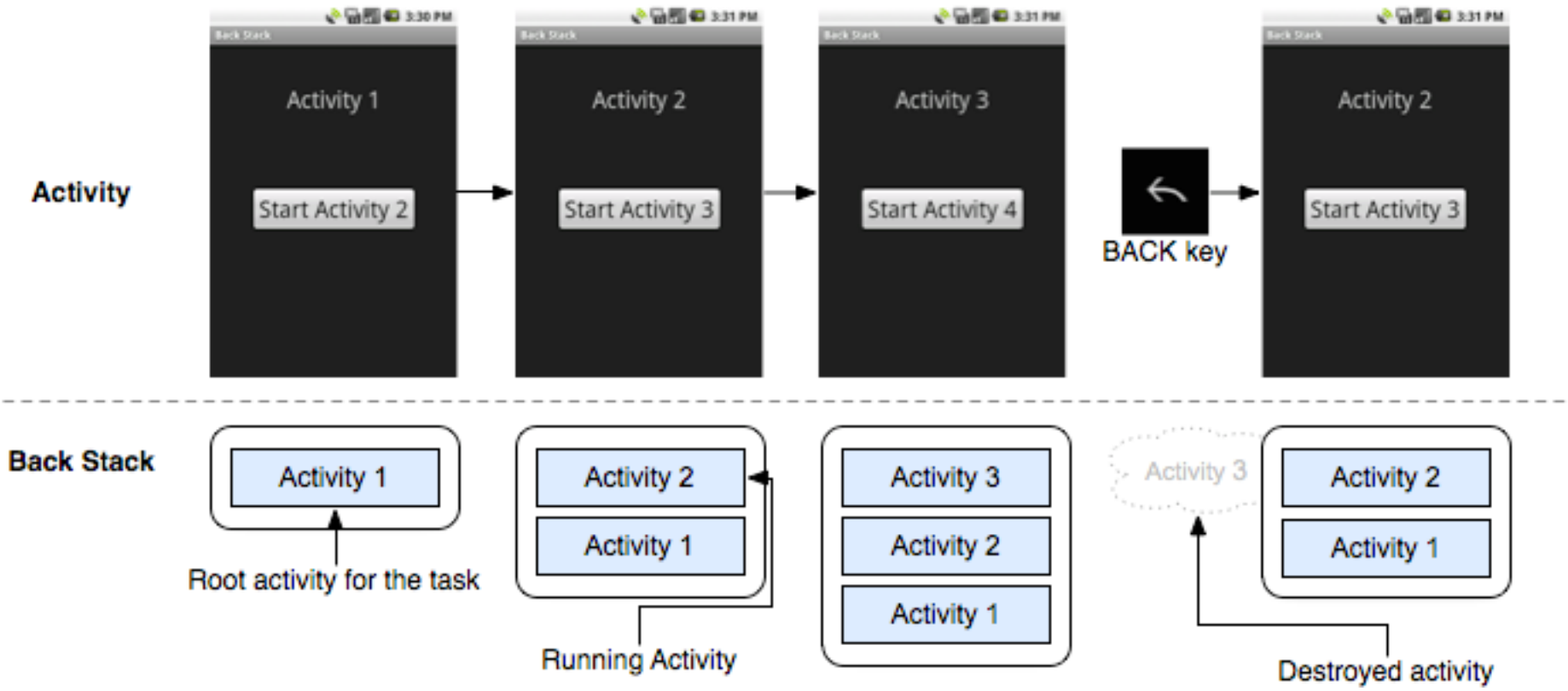
Понятие стэка: Task

Task – стэк из Activity.

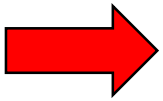
Обычно стартовая позиция для создания Task – это экран Домой (Home).Находясь в Home вы вызываете какое-либо приложение из списка приложений или через ярлык. Создается Task. И Activity приложения (которое отмечено как MAIN в манифест-файле) помещается в этот Task как корневое. Task выходит на передний фон.

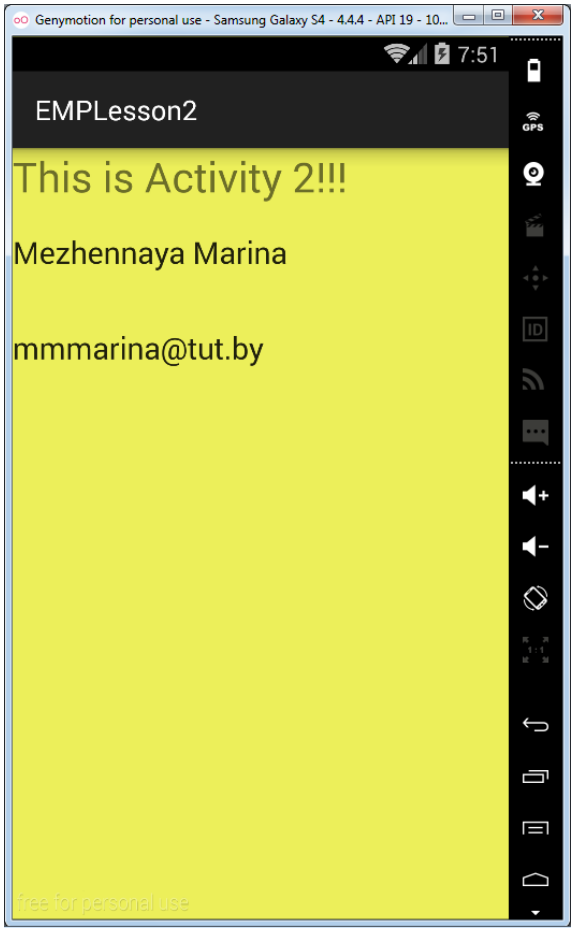
Когда Activity_A вызывает Activity_B, то Activity_B помещается на верх (в топ) Task и получает фокус. Activity_A остается в Task, но находится в состоянии Stopped (его не видно и оно не в фокусе). Далее, если пользователь жмет Back находясь в Activity_B, то Activity_B удаляется из Task и уничтожается.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

Понятие стэка: Task



Передача данных между Activity с помощью Intent





Передача данных между Activity с помощью Intent

1. **Метод putExtra**: добавляет к объекту пару: первый параметр – это ключ(имя), второй - значение.

MainActivity.java:

```
Intent intent = new Intent(this, SecondActivity.class);

intent.putExtra("Name", et_Name.getText().toString());
intent.putExtra("Email", et_Email.getText().toString());

startActivity(intent);
```

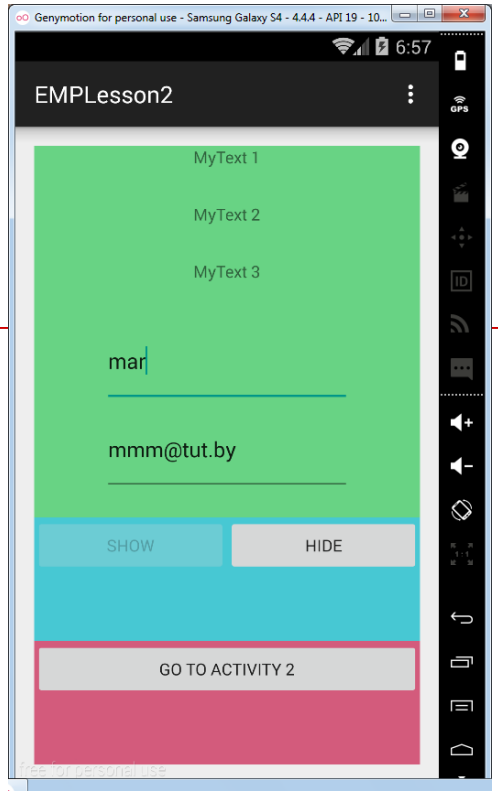
Передача данных между Activity с помощью Intent

2. Метод getExtra: извлекает значение по ключу.

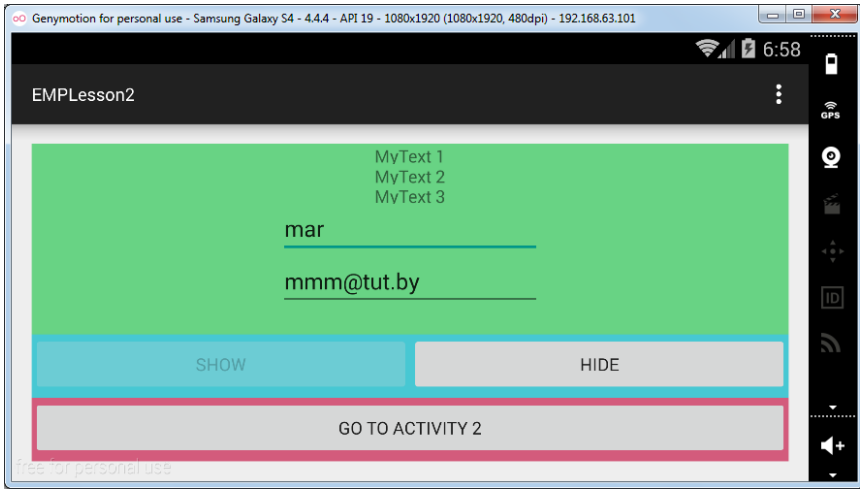
SecondActivity.java:

```
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
        initViews();

        Intent intent = getIntent();
        tv_Name.setText(intent.getStringExtra("Name"));
        tv_Email.setText(intent.getStringExtra("Email"));
    }
```

Смена ориентации экрана



При смене ориентации экрана системе надо заново его создавать и восстанавливать данные, которые были утеряны при уничтожении.

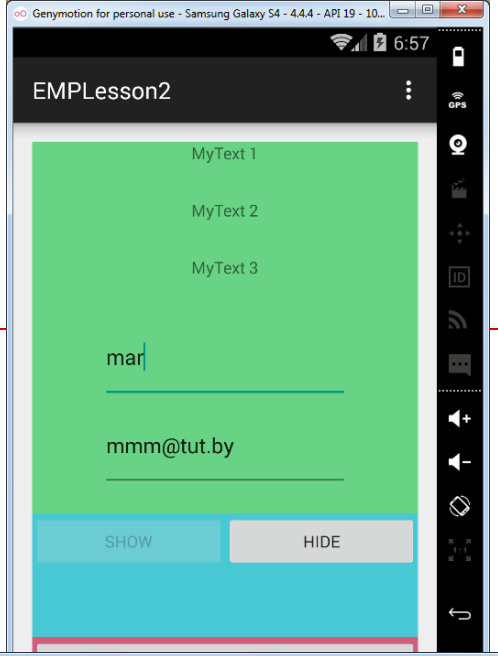


Для этих целей Activity предоставляет методы:
первый позволяет сохранить данные – onSaveInstanceState,
а второй – восстановить - onRestoreInstanceState.

Сохранение данных при повороте экрана

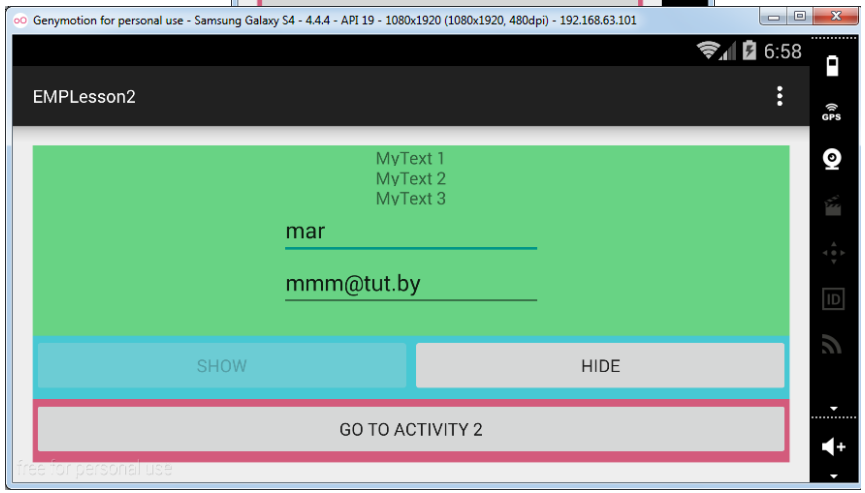
```
@Override
    protected void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);
        L.d("MainActivity: onSaveInstanceState");
}

    @Override
    protected void onRestoreInstanceState(Bundle
savedInstanceState) {
        super.onRestoreInstanceState(savedInstanceState);
        L.d("MainActivity: onRestoreInstanceState");
}
```

Жизненный цикл Activity



Запустили приложение:

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

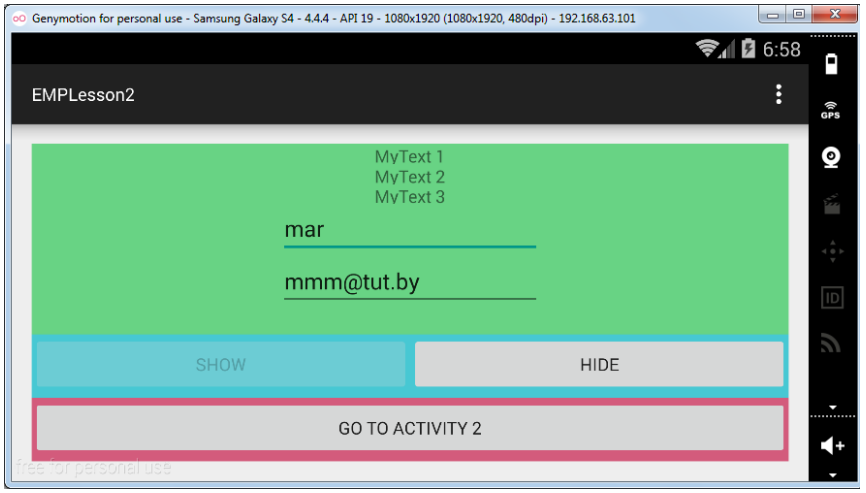


Поменяли ориентацию экрана:

MainActivity: onPause()
MainActivity: onSaveInstanceState()
MainActivity: onStop()
MainActivity: onDestroy()
MainActivity: onCreate()
MainActivity: onStart()
MainActivity: onRestoreInstanceState()
MainActivity: onResume()

Смена ориентации экрана

Даже если не реализовать эти методы, у них есть реализация по умолчанию, которая сохранит и восстановит данные в экранных компонентах. Это выполняется для всех экранных компонентов, у которых есть ID.



Если все же нужно сохранить данные:

```
@Override
    protected void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);
        L.d("MainActivity: onSaveInstanceState");
        outState.putString("SavedMessage", tv_Message.getText().toString());
}
    @Override
    protected void onRestoreInstanceState(Bundle savedInstanceState) {
        super.onRestoreInstanceState(savedInstanceState);
        L.d("MainActivity: onRestoreInstanceState");
        tv_Message.setText(savedInstanceState.getString("SavedMessage"));
}
```

Метод startActivityForResult

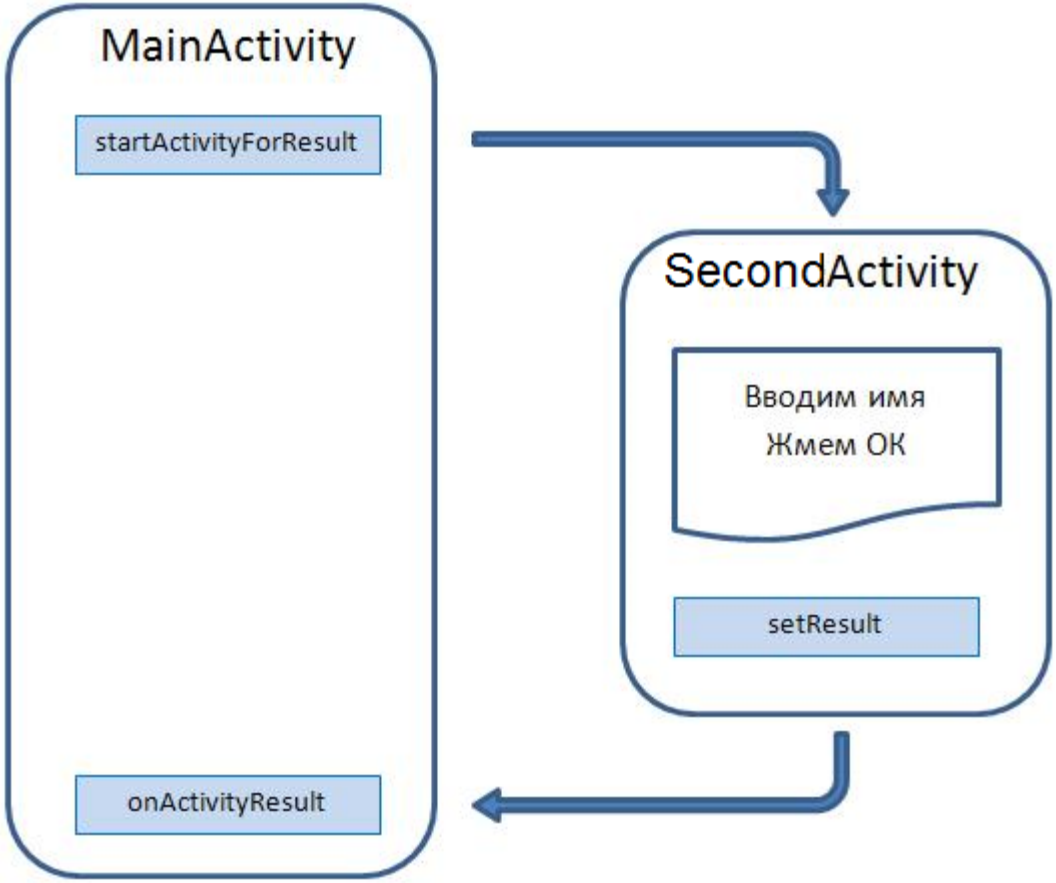
Метод startActivityForResult используется, когда необходимо вызвать Activity, выполнить на нем какое-либо действие и вернуться с результатом.

MainActivity.java:

…

```
case R.id.btnGotoActivity2:
        Intent intent = new Intent(this, SecondActivity.class);
        intent.putExtra("Name", et_Name.getText().toString());
        intent.putExtra("Email", et_Email.getText().toString());
        startActivityForResult(intent, 1);
        break;
```

Метод startActivityForResult



Метод startActivityForResult

MainActivity.java:

```
 @Override
    protected void onActivityResult(int requestCode, int resultCode,
Intent data) {
        if (data == null) {
            return;
        }
        String message = data.getStringExtra("Message");
        tv_Message.setText(message);
    }
```

Метод startActivityForResult

startActivityForResult:
Отличие от обычного startActivity в том, что MainActivity становится «родителем» для SecondActivity. И когда SecondActivity закрывается, вызывается метод onActivityResult в MainActivity, тем самым давая нам знать, что закрылось Activity, которое мы вызывали методом startActivityForResult.

Синтаксис метода:
startActivityForResult(Intent intent, int requestCode);

requestCode – идентификатор для определения, с какого Activity пришел результат.

Метод startActivityForResult

protected void **onActivityResult**(int requestCode, int resultCode, Intent data);

requestCode – тот же идентификатор, что и в startActivityForResult. По нему определяем, с какого Activity пришел результат.
resultCode – код возврата. Определяет успешно прошел вызов или нет.
data – Intent, в котором возвращаются данные.

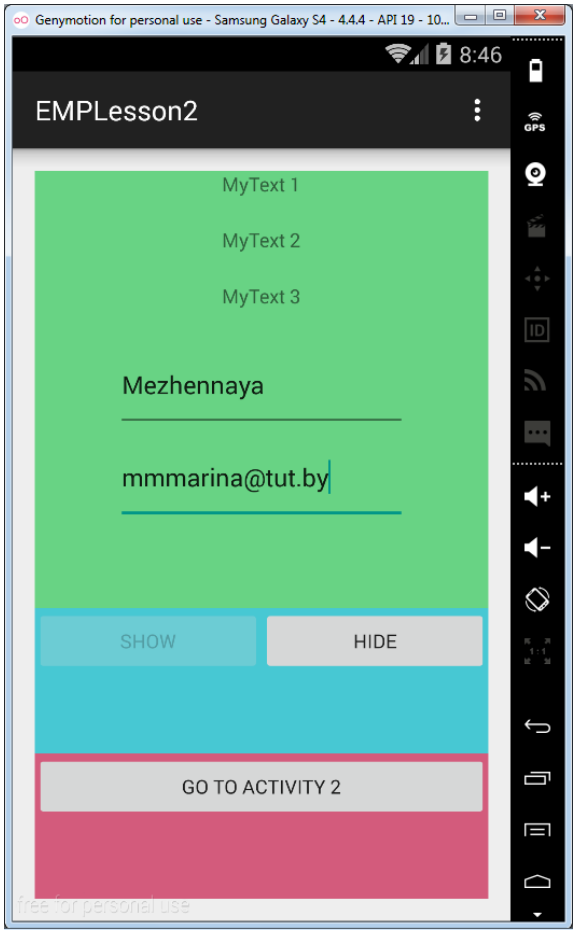
Метод startActivityForResult

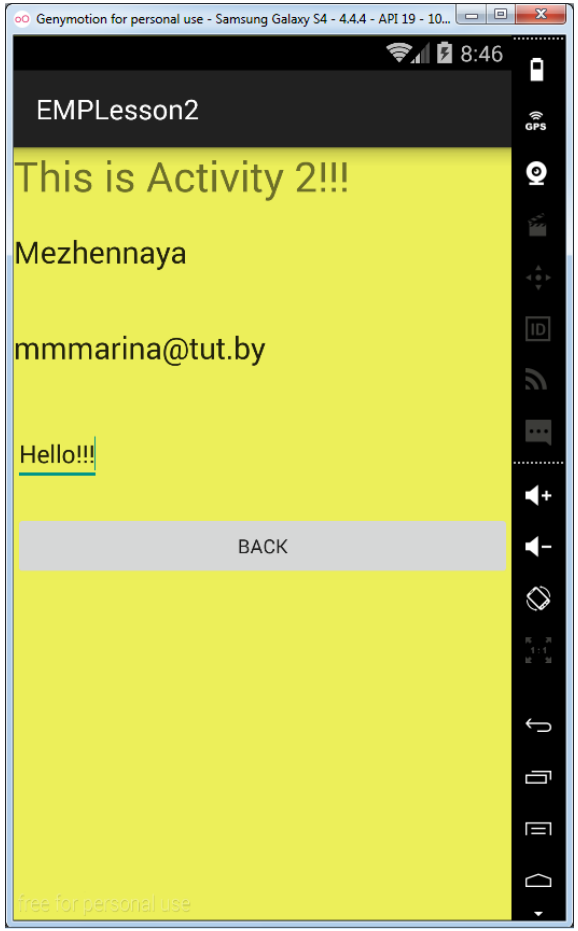
Метод setResult(int resultCode, Intent intent);
адресует данные с intent в «родительское» Activity.

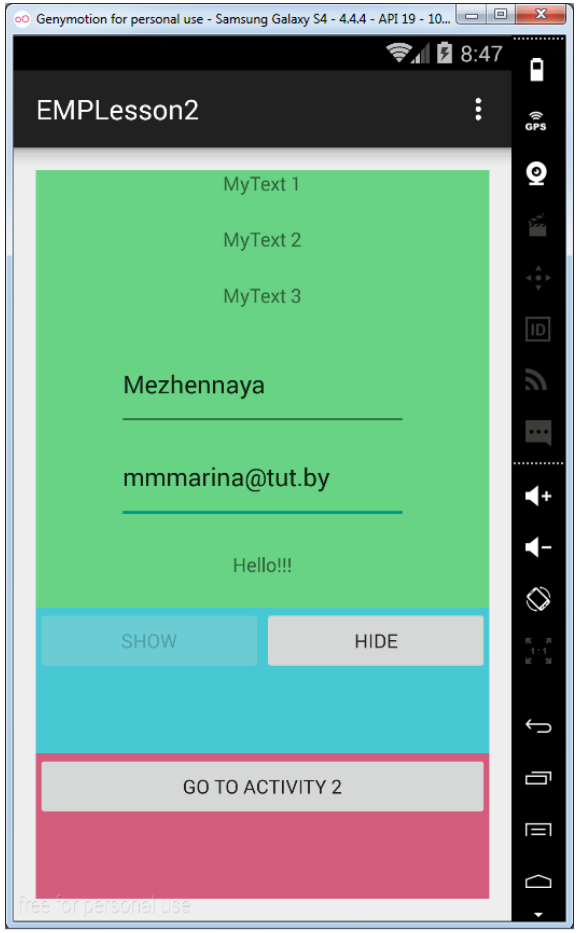
SecondActivity.java:

```
@Override
  public void onClick(View view) {
      Intent intent = new Intent();
      intent.putExtra("Message", et_Message.getText().toString());
      setResult(RESULT_OK, intent);
      finish();
  }
```

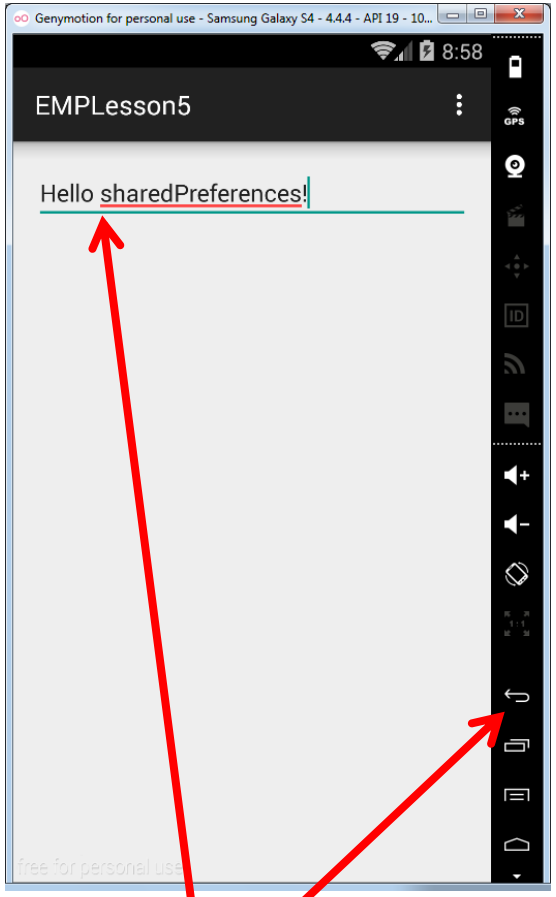
Метод startActivityForResult



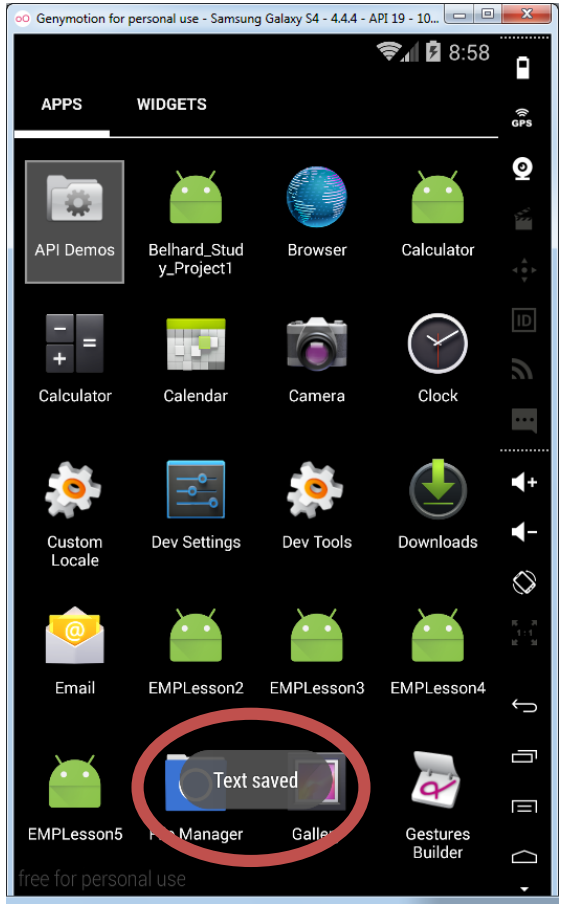




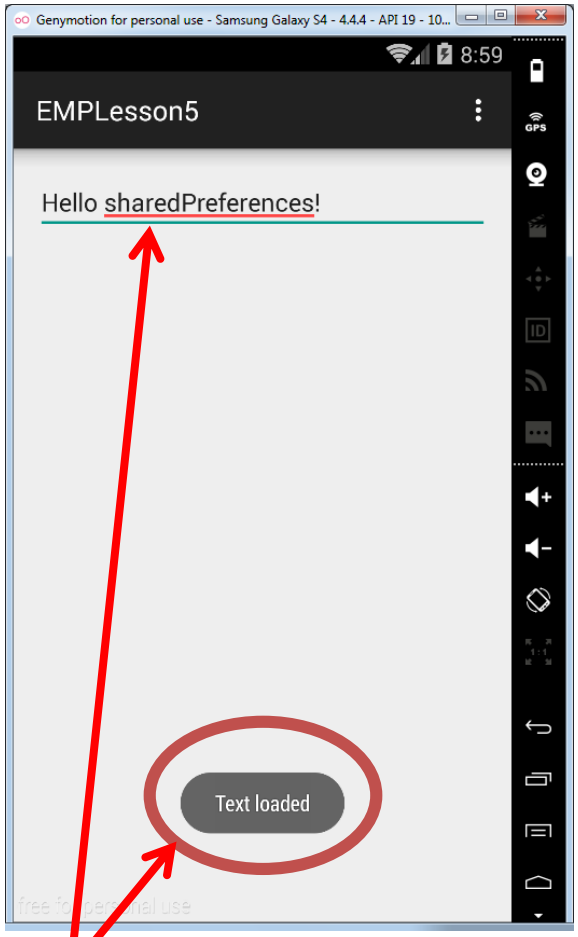
Хранение данных в виде пары: имя, значение с использованием класса SharedPreferences



1 Сохраняем данные



2 Закрываем приложение



3 Загружаем данные

```
public class MainActivity extends ActionBarActivity {

    private EditText myEditText;
    SharedPreferences sPref;
    final String SAVED_TEXT = "my_saved_text";

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

        initView();
        loadText();
    }

    private void initView() {
        myEditText = (EditText) findViewById(R.id.myEditText);
    }

 @Override
    protected void onDestroy() {
        super.onDestroy();
        saveText();
    }
```

```
private void saveText() {
        sPref = getPreferences(MODE_PRIVATE);
        SharedPreferences.Editor editor = sPref.edit();
        editor.putString(SAVED_TEXT, myEditText.getText().toString());
        editor.commit();

        Toast.makeText(this, "Text saved", Toast.LENGTH_SHORT).show();
    }


private void loadText() {
        sPref = getPreferences(MODE_PRIVATE);
        String savedText = sPref.getString(SAVED_TEXT, "");
        myEditText.setText(savedText);

        Toast.makeText(this, "Text loaded", Toast.LENGTH_SHORT).show();
    }
```