**Принципы работы c жестами вводимыми пользователями**

**A**

 |

[версия для печати](http://www.intuit.ru/intuit?destination=studies%2Fcurriculums%2F16235%2Fcourses%2F1191%2Fprint_lecture%2F21995)

[< Самостоятельная работа 5](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21994) || **Самостоятельная работа 6**: **1**[2](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=2)[3](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=3) || [Самостоятельная работа 7 >](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21996)

**Аннотация:**Разработка приложения, помогающего понять принципы работы c жестами вводимыми пользователями.

**Ключевые слова:**[android](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword1), [API](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword2), [gesture](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword4), [приложение](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword5), [AVD](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword8), [распознавание](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword10), [поле](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword16), [SD](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword17), [файл](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword18), [путь](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword21), [SDK](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword23), [perspective](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword29), [DDMS](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword30), [Директория](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#keyword32), [XML](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=2#keyword36),[Java](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=2#keyword41), [активность](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=2#keyword42), [класс](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=2#keyword43), [информация](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=2#keyword45)

**Цель лабораторной работы:**

Разработка приложения, помогающего понять принципы работы c жестами вводимыми пользователями.

**Задачи лабораторной работы:**

* создать набор жестов
* использовать созданные жесты в приложении

**11.1 Введение**

Начиная с версии 1.6, *Android* предоставляет *API* для работы с жестами, который располагается в пакете *android*.*gesture* и позволяет сохранять, загружать, создавать и распознавать жесты.

В данной работе рассмотрим процесс создания набора жестов, для создания жестов будем использовать *приложение* *Gesture* Builder, которое предустановлено на виртуальные устройства *Android* (*AVD*), начиная с версии 1.6.

После этого разработаем *приложение*, в котором предполагается *распознавание* и использование созданных жестов.

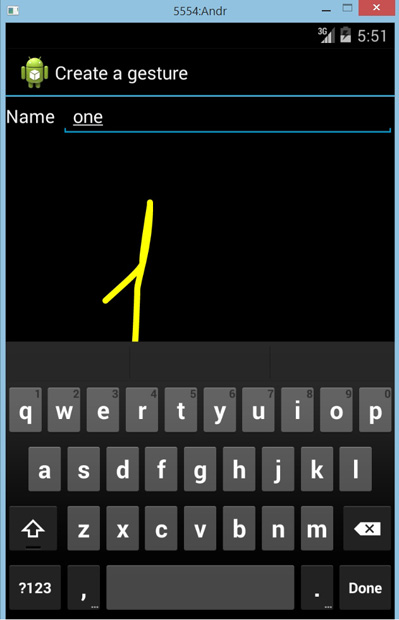
**11.2 Создание набора жестов**

Для начала создадим новое *приложение*.

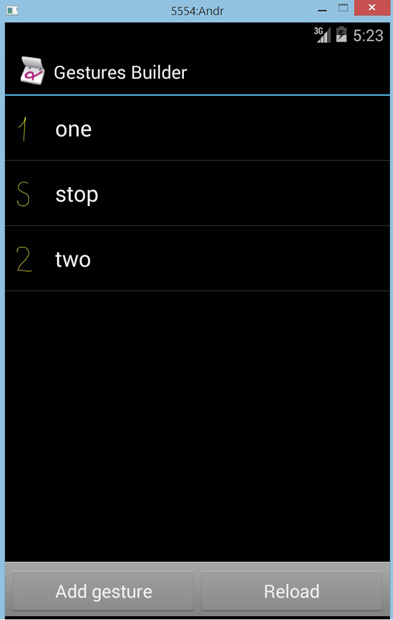
Далее запустим эмулятор и используем *приложение* *Gesture* Builder для создания жестов "1", "2" и "S". Жест всегда связан с именем, но имя не обязательно должно быть уникальным, в действительности, для повышения точности в распознавании жеста рекомендуется сохранять несколько жестов с одним и тем же именем.

На [рис. 11.1](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#image.11.1) можно увидеть *приложение* *Gesture* Builder в работе, чтобы добавить жест, необходимо нажать на кнопку **Add gesture**, в свободном пространстве изобразить жест (обычно он рисуется желтым цветом), в *поле* ввода задать имя жеста. Результат последовательного добавления жестов можно увидеть на [рис. 11.2](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#image.11.2).

Жесты сохраняются на *SD* карте эмулятора, чтобы использовать их в приложении необходимо импортировать *файл* жестов в проект.

[](http://www.intuit.ru/EDI/26_09_16_2/1474842043-31192/tutorial/1258/objects/11/files/11_01.jpg)

[увеличить изображение](http://www.intuit.ru/EDI/26_09_16_2/1474842043-31192/tutorial/1258/objects/11/files/11_01.jpg)  
**Рис. 11.1.**Создание жестов с помощью приложения Gesture Builder

[](http://www.intuit.ru/EDI/26_09_16_2/1474842043-31192/tutorial/1258/objects/11/files/11_02.jpg)

[увеличить изображение](http://www.intuit.ru/EDI/26_09_16_2/1474842043-31192/tutorial/1258/objects/11/files/11_02.jpg)  
**Рис. 11.2.**Набор жестов, созданных в приложении Gesture Builder

**Комментарий:** *Gesture* Builder может сообщить, что ему некуда сохранять жесты, в этом случае необходимо запустить эмулятор с образом *SD*карты.

Сначала образ нужно создать с помощью утилиты **mksdcard** (расположена в папке **<AndrSDK>/sdk/tools**, где **<AndrSDK>** - *путь*, куда установлен *Android* *SDK*). В командной строке напишем:

mksdcard -l mySdCard 64M gesture.img

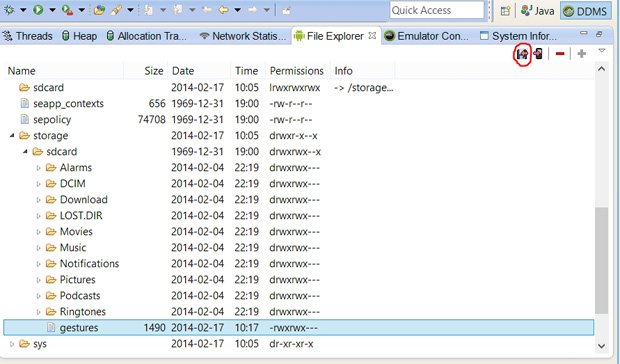
Следующим шагом будет создать и запустить эмулятор с образом *gesture*.img, для этого зайдем в папку **<AndrSDK>/sdk/tools** и выполним команду:

emulator -avd nameEmulator -sdcard gestures.img

nameEmulator - имя, которое присвоено эмулятору при создании.

Интересная статья на эту тему по ссылке: <http://habrahabr.ru/post/120016/>

Каждый раз, когда мы создаем или редактируем жесты с помощью *Gesture* Builder, создается *файл* gestures на *SD* карте эмулятора. Необходимо импортировать этот *файл* в директорию **res/raw/**, созданного проекта, в котором планируем использовать жесты.

[](http://www.intuit.ru/EDI/26_09_16_2/1474842043-31192/tutorial/1258/objects/11/files/11_03.jpg)

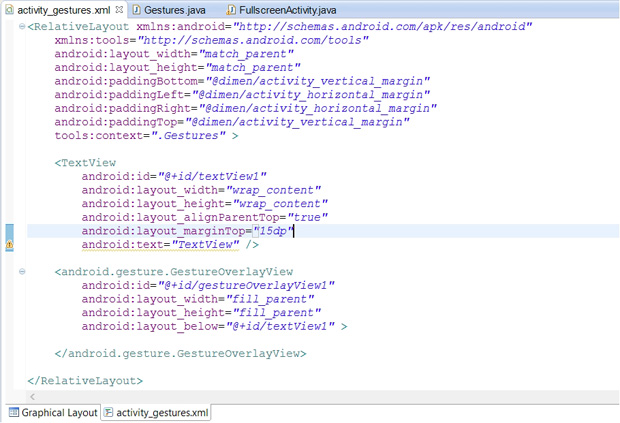
[увеличить изображение](http://www.intuit.ru/EDI/26_09_16_2/1474842043-31192/tutorial/1258/objects/11/files/11_03.jpg)  
**Рис. 11.3.**Расположение файла gestures

Самый простой способ импортировать жесты в проект заключается использовании вкладки **File Explorer** в компоновке (*perspective*) *DDMS*. (Если компоновки *DDMS* нет найти ее можно следующим образом: **Window->Open Perspective->Other...->DDMS**. Если вкладки **File Explorer** нет, можно добавить: **Window-> Show View-> File Explorer**). На вкладке **File Explorer** найти директорию **sdcard/** (в нашем случае оказалась *директория* **storage/sdcard/**, имеет смысл при создании жестов обратить внимание в какую директорию *Gesture* Builder их сохраняет). На [рис. 11.3](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#image.11.3) показана вкладка **File Explorer** в компоновке *DDMS*.

Чтобы скопировать *файл* жестов с эмулятора в проект, необходимо выбрать его и нажать кнопку "**Pull a file from the device**", выделенную на[рис. 11.3](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=1#image.11.3) красным подобием окружности. Откроется диалог с предложением выбрать папку, в которую необходимо скопировать жесты, здесь надо найти папку проекта, в ней папку **res/raw/** (если папки **raw/** нет, ее необходимо создать) и нажать кнопку **Сохранить**. Теперь жесты есть в нашем проекте и их можно использовать.

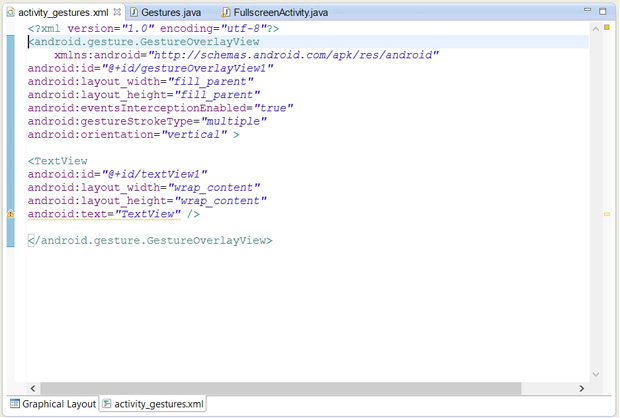
**11.3 Использование созданных жестов в приложении**

Для распознавания жестов необходимо добавить элемент GestureOverlayView в *XML* *файл* активности. И этот *файл* может выглядеть, например, как показано на [рис. 11.4](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=2#image.11.4):

[](http://www.intuit.ru/EDI/26_09_16_2/1474842043-31192/tutorial/1258/objects/11/files/11_04.jpg)

[увеличить изображение](http://www.intuit.ru/EDI/26_09_16_2/1474842043-31192/tutorial/1258/objects/11/files/11_04.jpg)  
**Рис. 11.4.**XML файл активности приложения, элемент GestureOverlayView обычный компонент интерфейса пользователя

Можно добавить элемент GestureOverlayView поверх всех компонентов, как прозрачный слой, в этом случае *XML* *файл* активности может выглядеть так, как показано на [рис. 11.5](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=2#image.11.5).

[](http://www.intuit.ru/EDI/26_09_16_2/1474842043-31192/tutorial/1258/objects/11/files/11_05.jpg)

[увеличить изображение](http://www.intuit.ru/EDI/26_09_16_2/1474842043-31192/tutorial/1258/objects/11/files/11_05.jpg)  
**Рис. 11.5.**XML файл активности приложения, элемент GestureOverlayView поверх всех компонентов интерфейса пользователя

Далее необходимо обработать ввод жеста пользователя, сравнить с загруженными жестами, и либо определить жест, либо сообщить пользователю, что такого жеста нет. Теперь вся работа будет выполняться в *java* файле, описывающем главную (и единственную) *активность*приложения. Внесем в этот *класс* следующие дополнения:

* Класс активности должен реализовывать интерфейс OnGesturePerformedListener, для этого в объявление класса добавим конструкцию:
* implements OnGesturePerformedListener;

* Нам понадобятся экземпляры классов GestureLibrary и GestureOverlayView, поэтому в качестве полей класса активности объявим следующие переменные:
* GestureLibrary gLib;
* GestureOverlayView gestures;

* В методе onCreate() выполним следующие действия:
* gLib = GestureLibraries.fromRawResource(this, R.raw.gestures);
* if (!gLib.load()) {
* finish();
* }

В первой строке выполнена инициализация переменной gLib жестами, загруженными из файла gestures папки **res/raw/**.

Оператор if выполняет проверку загружены ли жесты, если нет, выполняется выход из приложения.

* Добавим в метод onCreate() еще две строчки:
* gestures = (GestureOverlayView) findViewById(R.id.gestureOverlayView1);
* gestures.addOnGesturePerformedListener(this);

Для инициализации переменной gesture и подключения к ней слушателя событий появления жеста.

* И наконец напишем реализацию метода OnGesturePerformed(), который и будет вызываться при появлении события, соответствующего какому-либо жесту.
* public void onGesturePerformed(GestureOverlayView overlay, Gesture gesture) {
* //Создаёт ArrayList c загруженными из gestures жестами
* ArrayList<Prediction> predictions = gLib.recognize(gesture);
* if (predictions.size() > 0) {
* //если загружен хотябы один жест из gestures
* Prediction prediction = predictions.get(0);
* if (prediction.score > 1.0) {
* if (prediction.name.equals("one"))
* tvOut.setText("1");
* else if (prediction.name.equals("stop"))
* tvOut.setText("stop");
* else if (prediction.name.equals("two"))
* tvOut.setText("2");
* }else{
* tvOut.setText("Жест неизвестен");
* }
* }
* }

В приложении всего лишь распознаются жесты и в информационное *поле* выводится *информация* о том, что за жест был использован. В[листинге 11.1](http://www.intuit.ru/studies/curriculums/16235/courses/1191/lecture/21995?page=3#example.11.1) представлен возможный код приложения.

**11.4 Заключение**

В качестве практического задания предлагаем реализовать жестами ввод чисел в приложении "Угадайка", разработанном в лабораторной работе второй темы. Создать жесты "0", "1", "2", "3", "4", "5", "6", "7", "8", "9" для ввода цифр и жест "S" для остановки ввода числа. В*приложение* добавить *распознавание* этих жестов, преобразование их в число и сравнение полученного числа с загаданным.

Еще варианты для самостоятельной работы:

1. разработать простой калькулятор с жестовым вводом чисел и операций;
2. разработать блокнотик для заметок с рукописным вводом текста.

package com.example.lab5\_2\_gestures;

import java.util.ArrayList;

import android.os.Bundle;

import android.app.Activity;

import android.gesture.Gesture;

import android.gesture.GestureLibraries;

import android.gesture.GestureLibrary;

import android.gesture.GestureOverlayView;

import android.gesture.GestureOverlayView.OnGesturePerformedListener;

import android.gesture.Prediction;

import android.view.Menu;

import android.widget.TextView;

public class Gestures extends Activity implements OnGesturePerformedListener{

GestureLibrary gLib;

GestureOverlayView gestures;

TextView tvOut;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_gestures);

tvOut=(TextView)findViewById(R.id.textView1);

//Загрузка жестов (gestures) из res/raw/gestures

gLib = GestureLibraries.fromRawResource(this, R.raw.gestures);

if (!gLib.load()) {

//Если жесты не загружены, то выход из приложения

finish();

}

gestures = (GestureOverlayView) findViewById(R.id.gestureOverlayView1);

gestures.addOnGesturePerformedListener(this);

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.gestures, menu);

return true;

}

public void onGesturePerformed(GestureOverlayView overlay, Gesture gesture) {

//Создаёт ArrayList c загруженными из gestures жестами

ArrayList<Prediction> predictions = gLib.recognize(gesture);

if (predictions.size() > 0) {

//если загружен хотя бы один жест из gestures

Prediction prediction = predictions.get(0);

if (prediction.score > 1.0) {

if (prediction.name.equals("one"))

tvOut.setText("1");

else if (prediction.name.equals("stop"))

tvOut.setText("stop");

else if (prediction.name.equals("two"))

tvOut.setText("2");

}else{

tvOut.setText("Жест неизвестен");

}

}

}

}

Листинг 11.1. Распознавание жестов загруженных в файл res/raw/gestures