Создаем Android-приложение для управления домашним роботом через Bluetooth

Опубликовано 16.02.2014

В этой статье представлена пошаговая инструкция, которая поможет вам самостоятельно создать приложение для Android-смартфона, предназначенное для управления чем-либо через Bluetooth. Для демонстрации мы подробно разберем пример мигания светодиодом на Arduino по командам с телефона или планшета. В результате выполнения наших инструкций вы научитесь делать вот так:
Для управления домашним роботом достаточно добавить кнопок и обработать их команды на стороне Arduino.

Что для этого потребуется

1. Любая Arduino-совместимая плата
2. Bluetooth-модуль
3. Устройство на котором установлена ОС Android

В качестве Bluetooth-модуля лучше всего использовать HC-05. Его легко купить в китайском интернет магазине или на eBay. Модуль питается от 3.3 В, но его линии I/O могут работать и с 5-вольтовой логикой, что позволяет подключать его UART к Arduino.

Bluetooth-модуль HC-05

Подключение Bluetooth-модуля к Arduino

Так теперь нам нужно подключить нашу Arduino с Bluetooth. Если на Arduino нет вывода с 3.3В , а только 5В то нужен будет поставить стабилизатор чтобы снизить питание. Назначение выводов HC-05 легко найти в интернете. Для использования рекомендуем вам сделать плату с выведенными линиями питания, Rx и Tx. Подключение к Arduino необходимо производить в следующем порядке:

- вывод Arduino 3.3В или (5В через стабилизатор!) — к 12 пину модуля Bluetooth
- вывод Arduino GND — к 13 пину модуля Bluetooth
- вывод Arduino TX — к 2 пину модуля RX Bluetooth
- вывод Arduino RX — к 1 пину модуля TX Bluetooth

После подключения необходимо проверить работоспособность Bluetooth модуля. Подключим Светодиод к 12 выводу Arduino и загрузим на плату следующий скетч:

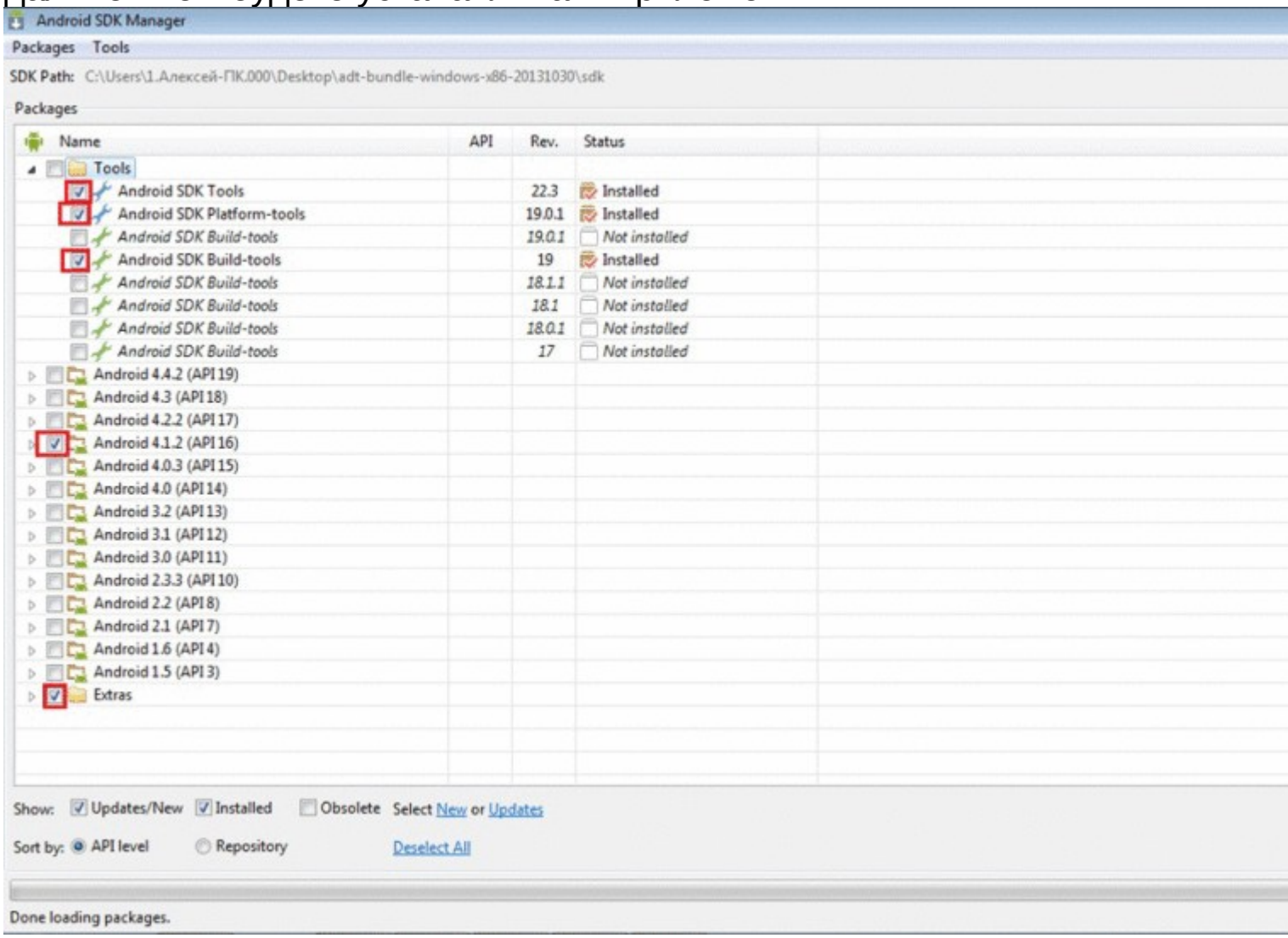
```
1. char incomingByte;  // входящие данные
2. int  LED = 12;      // LED подключен к 12 пину
3.
4. void setup() {
5.   Serial.begin(9600); // инициализация порта
6.   pinMode(LED, OUTPUT); //Устанавливаем 12 вывод как выход
7.   Serial.println("Press 1 to LED ON or 0 to LED OFF...");
8. }
9.
10. void loop() {
11.   if (Serial.available() > 0) {  //если пришли данные
12.     incomingByte = Serial.read(); // считываем байт
13.     if(incomingByte == '0')
14.     {
15.       digitalWrite(LED, LOW);  // если 1, то выключаем LED
16.       Serial.println("LED OFF. Press 1 to LED ON!");  // и выводим
    обратно сообщение
17.     }
18.     if(incomingByte == '1') {
19.       digitalWrite(LED, HIGH); // если 0, то включаем LED
20.       Serial.println("LED ON. Press 0 to LED OFF!");
21.     }
22.   }
23. }
```

Теперь скачиваем из Play Market программу Bluetooth-терминал и устанавливаем его. Включаем нашу Arduino. В приложении Нажимаем кнопку меню->Connect a device-Secure.

Тем самым ваше устройство начнём искать Bluetooth поблизости. Наш модуль должен называться HC-05. Вам потребуется выписать его MAC-адрес, так он понадобится в дальнейшем. Как только он обнаружит устройство HC-05 выберите его. Пароль, если потребуется: 1234 (это стандартный код). После того как вы подключились к нему у вас должно вывести сообщение которое пришло в Bluetooth терминал от Arduino: “Press 1 to LED ON or 0 to LED OFF..” Далее введите 1 и нажмите отправить. Тем самым вы посылаете цифру 1 через Bluetooth на Arduino. Как только он примет цифру 1 должен загореться светодиод подключенный к 12 выводу Arduino. После введите цифру 0 и светодиод должен погаснуть. Если всё получилось переходим дальше.

Установка Android SDK

Скачиваем с официального сайта программу для создания приложений для android любых моделей. Распаковываем архив, запускаем SDK Manager.exe и устанавливаем программу. Вам предложат установить API, и версию android для которой вы в дальнейшем будете устанавливать приложения.



После нажимаем кнопку Install, ждём когда завершиться установка и закрываем окно.
Заходим в саму программу, она находиться в папке eclipse/eclipse.exe.
После открытия программы в диалоговом окне необходимо указать директорию для хранения будущих проектов. Лучше создавать папку на локальном диске, используя при этом только латинские буквы.

Создание приложения

Выбираем File->New->Project.

Так как мы создаём приложение для android, выбираем Android-> Android Application Project, и нажимаем Next

Следующее диалоговое окно:

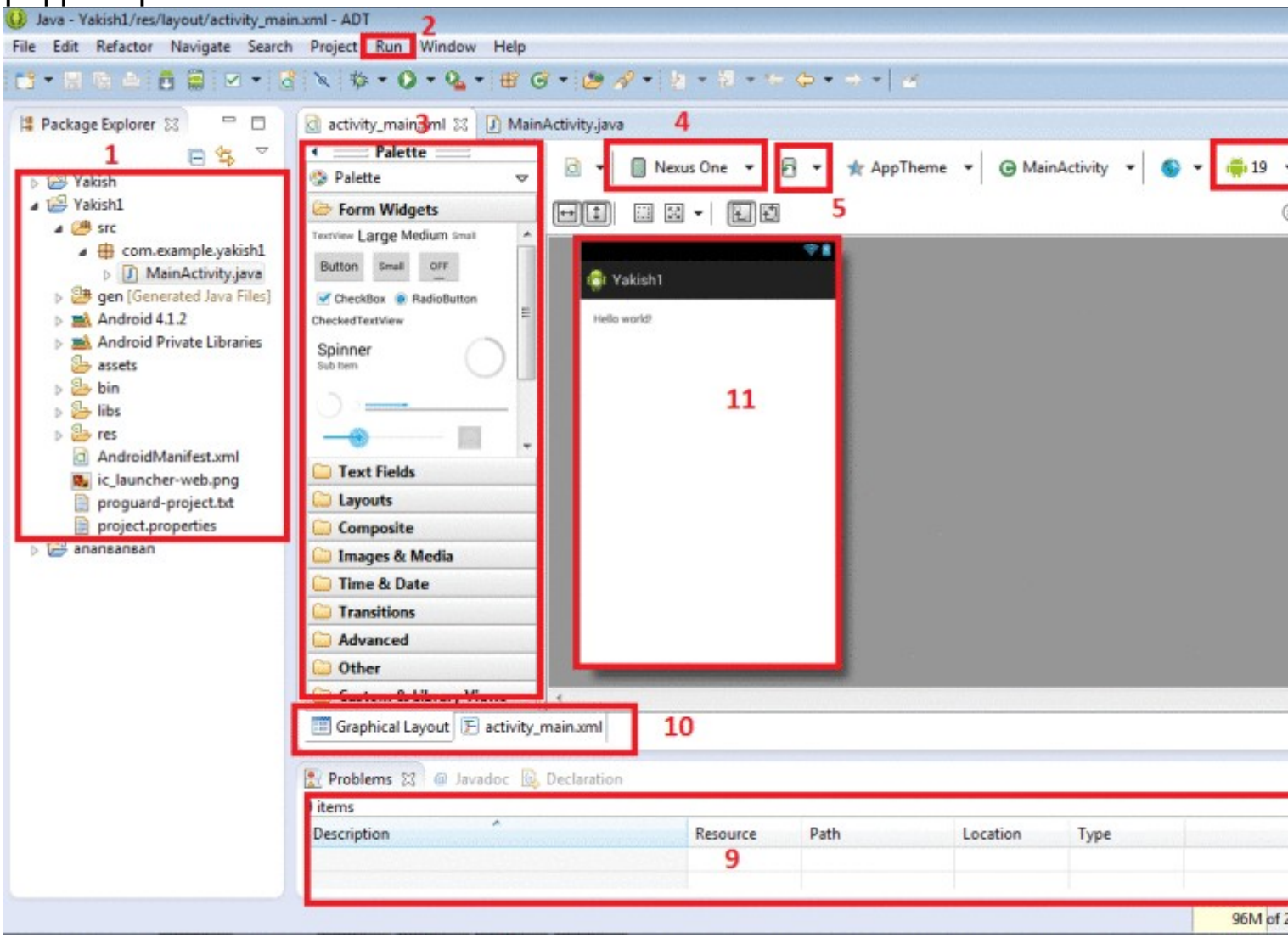
Application Name -> пишем имя приложение,
Project Name -> пишем имя проекта,
Package Name -> Ни чего не пишем он создается автоматически!
Minimum Required SDK -> это минимальные требование указываем нашу версию Android у меня 4.1 её я и выбираю.
Target SDK -> выбираем вашу версию Android
Compile with -> выбираем вашу версию Android
Theme: для начала я бы советовал выбрать None.
Нажимаем Next.
В следующем окне ничего менять не нужно. Просто жмем Next.

Далее нам предлагают создать свою иконку для приложения, можете изменить стандартный ярлык загрузив свою картинку, я же для начала предлагаю просто нажать Next.

В следующем необходимо выбрать пункт меню Blank Activity и нажимать Next.

Жмем Finish и через несколько секунд открывается главное окно нашей программы. Выбираем вкладку Activity_main.xml и видим наш

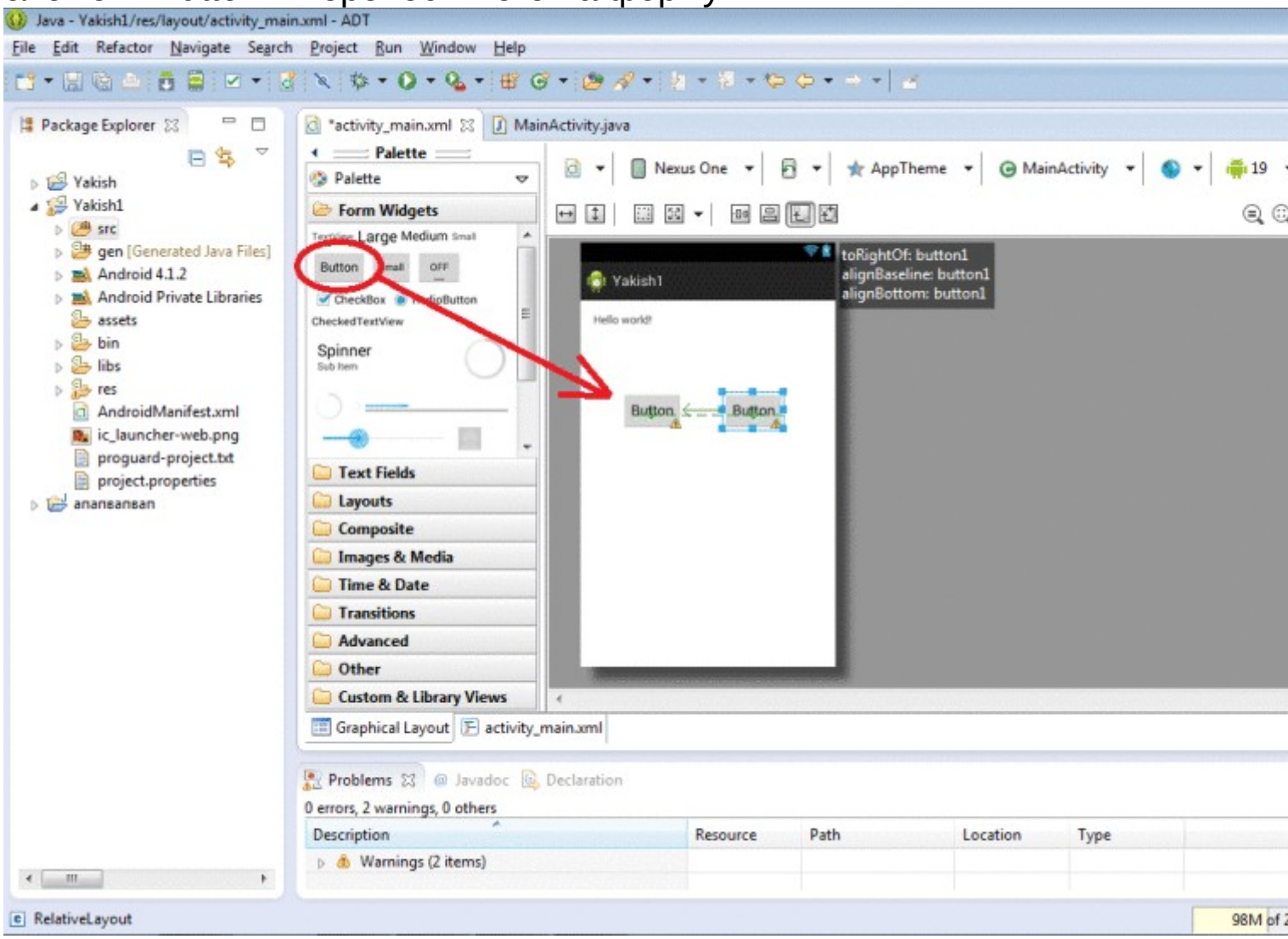
редактор:



1. Файлы нашего проекта.
2. Run Запуск эмулятора для проверки программы на наличие ошибок
3. Панель кнопок текста и многое другое от сюда вы будите выбирать что вам нужно и добавлять в качестве элементов приложения
4. Для выбора размера дисплея вашего телефона или планшета
5. Выбор ориентации. Два вида: горизонтальный и вертикальный
6. API уровень (лучше не трогать)
7. Тут будет отображаться всё то что вы добавили в приложение, так же тут можно переименовать ваши добавленные элементы или удалять их.
8. Показывает свойства элемента, его размер цвет и т.д., так же тут можно редактировать элемент
9. Показывает наличие ошибок.

10. Выбор редактирования (графический либо текстовой). Для начинающих конечно лучше пользоваться графическим режимом
11. Окно вашего приложения , можно видеть интерфейс будущего приложения

Теперь добавим две кнопки в интерфейс приложения. Выбираем элемент Button и переносим его на форму.



Справа вверху мы видим объекты которые мы добавили. Так же важно, какой из объектов выбран в данный момент. Справа внизу можно редактировать кнопку, давайте изменим текст подписи кнопки и его цвет.

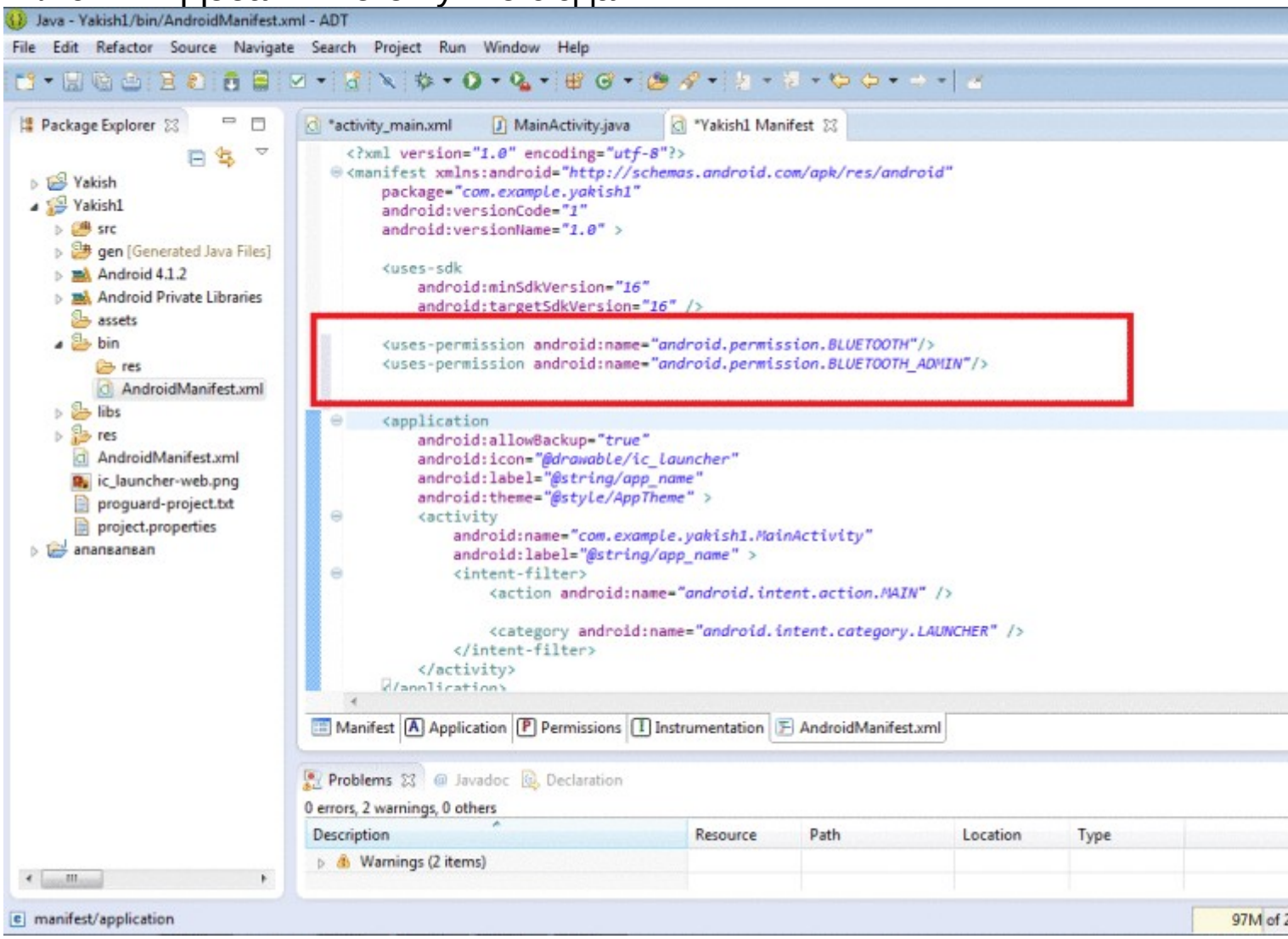
Для этого в поле свойств элемента «Text» введите, вместо button1, значение «ВКЛ», а у button2 — “ВЫКЛ”. Должно получиться вот так:

Мы можем запустить только что созданное приложение на эмуляторе. Идем в настройки запуска «Run» → Run Configurations», в левой части нажимаем на «Android Application». Появляется новая конфигурация «New_configuration». В правой части окна выбираем вкладку «Target» и выбираем опцию «Launch on all compatible devices/AVD» и добавляем

устройство. Проверяем что кнопки появились и их можно нажимать. Если всё хорошо — продолжаем дальше.
Теперь в файлах проекта выбираем bin->AndroidManifest.hml

Теперь нажмём снизу на AndroudManifest.hml

В этот файл нам нужно будет добавить две строки:
Они будут запрашивать включение Bluetooth при старте приложение, если он будет выключен приложение попросит пользователя его включить. Добавить его нужно сюда:



Далее откроем другой файл: src->com.example(name)

В этом файле и будет наш основной код. Все его содержимое нужно удалить и вставить вот этот код:

```
1. package com.example.NAME;//Вместо NAME введите имя
   вашего проекта
2.
3. import java.io.IOException;
4. import java.io.OutputStream;
```

```
5. import java.util.UUID;
6.
7. import com.example.NAME.R;
8.
9. import android.app.Activity;
10. import android.bluetooth.BluetoothAdapter;
11. import android.bluetooth.BluetoothDevice;
12. import android.bluetooth.BluetoothSocket;
13. import android.content.Intent;
14. import android.os.Bundle;
15. import android.util.Log;
16. import android.view.View;
17. import android.view.View.OnClickListener;
18. import android.widget.Button;
19. import android.widget.Toast;
20.
21. public class MainActivity extends Activity {
22.   private static final String TAG = "bluetooth1";
23.
24.   Button button1, button2;//Указываем id наших кнопок
25.
26.   private static final int REQUEST_ENABLE_BT = 1;
27.   private BluetoothAdapter btAdapter = null;
28.   private BluetoothSocket btSocket = null;
29.   private OutputStream outStream = null;
30.
31.   // SPP UUID сервиса
32.   private static final UUID MY_UUID =
   UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");
33.
34.   // MAC-адрес Bluetooth модуля
35.   private static String address = "00:00:00:00:00";  //Вместо
   “00:00” Нужно нудет ввести MAC нашего bluetooth
36.
37.   /** Called when the activity is first created. */
38.   @Override
```

```
39.  public void onCreate(Bundle savedInstanceState) {
40.    super.onCreate(savedInstanceState);
41.
42.    setContentView(R.layout.activity_main);
43.
44.    button1 = (Button) findViewById(R.id.button1); //Добавляем
   сюда имена наших кнопок
45.    button2 = (Button) findViewById(R.id.button2);
46.
47.    btAdapter = BluetoothAdapter.getDefaultAdapter();
48.    checkBTState();
49.
50.    button1.setOnClickListener(new OnClickListener()  //Если будет
   нажата кнопка 1 то
51.    {
52.      public void onClick(View v)
53.      {
54.        sendData("1");        // Посылаем цифру 1 по bluetooth
55.        Toast.makeText(getBaseContext(), "Включаем LED",
   Toast.LENGTH_SHORT).show();  //выводим на устройстве
   сообщение
56.      }
57.    });
58.
59.    button2.setOnClickListener(new OnClickListener() {
60.      public void onClick(View v)
61.       {
62.        sendData("0"); // Посылаем цифру 1 по bluetooth
63.        Toast.makeText(getBaseContext(), "Выключаем LED",
   Toast.LENGTH_SHORT).show();
64.      }
65.    });
66.  }
67.
68.  @Override
69.  public void onResume() {
```

```
70.     super.onResume();
71.
72.     Log.d(TAG, "...onResume - попытка соединения...");
73.
74.     // Set up a pointer to the remote node using it's address.
75.     BluetoothDevice device =
    btAdapter.getRemoteDevice(address);
76.
77.     // Two things are needed to make a connection:
78.     //   A MAC address, which we got above.
79.     //   A Service ID or UUID.  In this case we are using the
80.     //     UUID for SPP.
81.     try {
82.       btSocket =
    device.createRfcommSocketToServiceRecord(MY_UUID);
83.     } catch (IOException e) {
84.       errorExit("Fatal Error", "In onResume() and socket create
    failed: " + e.getMessage() + ".");
85.     }
86.
87.     // Discovery is resource intensive.  Make sure it isn't going on
88.     // when you attempt to connect and pass your message.
89.     btAdapter.cancelDiscovery();
90.
91.     // Establish the connection.  This will block until it connects.
92.     Log.d(TAG, "...Соединяемся...");
93.     try {
94.       btSocket.connect();
95.       Log.d(TAG, "...Соединение установлено и готово к
    передачи данных...");
96.     } catch (IOException e) {
97.       try {
98.         btSocket.close();
99.       } catch (IOException e2) {
100.                errorExit("Fatal Error", "In onResume() and unable to
    close socket during connection failure" + e2.getMessage() + ".");
```

```
101.             }
102.           }
103.
104.           // Create a data stream so we can talk to server.
105.           Log.d(TAG, "...Создание Socket...");
106.
107.           try {
108.             outStream = btSocket.getOutputStream();
109.           } catch (IOException e) {
110.             errorExit("Fatal Error", "In onResume() and output
    stream creation failed:" + e.getMessage() + ".");
111.           }
112.         }
113.
114.         @Override
115.         public void onPause() {
116.           super.onPause();
117.
118.           Log.d(TAG, "...In onPause()...");
119.
120.           if (outStream != null) {
121.             try {
122.               outStream.flush();
123.             } catch (IOException e) {
124.               errorExit("Fatal Error", "In onPause() and failed to flush
    output stream: " + e.getMessage() + ".");
125.             }
126.           }
127.
128.           try     {
129.             btSocket.close();
130.           } catch (IOException e2) {
131.             errorExit("Fatal Error", "In onPause() and failed to close
    socket." + e2.getMessage() + ".");
132.           }
133.         }
```

```
134.
135.         private void checkBTState() {
136.           // Check for Bluetooth support and then check to make
  sure it is turned on
137.           // Emulator doesn't support Bluetooth and will return
  null
138.           if(btAdapter==null) {
139.             errorExit("Fatal Error", "Bluetooth не
  поддерживается");
140.           } else {
141.             if (btAdapter.isEnabled()) {
142.               Log.d(TAG, "...Bluetooth включен...");
143.             } else {
144.               //Prompt user to turn on Bluetooth
145.               Intent enableBtIntent = new
  Intent(btAdapter.ACTION_REQUEST_ENABLE);
146.               startActivityForResult(enableBtIntent,
  REQUEST_ENABLE_BT);
147.             }
148.           }
149.         }
150.
151.         private void errorExit(String title, String message){
152.           Toast.makeText(getBaseContext(), title + " - " + message,
  Toast.LENGTH_LONG).show();
153.           finish();
154.         }
155.
156.         private void sendData(String message) {
157.           byte[] msgBuffer = message.getBytes();
158.
159.           Log.d(TAG, "...Посылаем данные: " + message + "...");
160.
161.           try {
162.             outStream.write(msgBuffer);
163.           } catch (IOException e) {
```

```
164.             String msg = "In onResume() and an exception occurred
   during write: " + e.getMessage();
165.             if (address.equals("00:00:00:00:00:00"))
166.               msg = msg + ".\n\nВ переменной address у вас
   прописан 00:00:00:00:00:00, вам необходимо прописать
   реальный MAC-адрес Bluetooth модуля";
167.               msg = msg +  ".\n\nПроверьте поддержку SPP UUID:
   " + MY_UUID.toString() + " на Bluetooth модуле, к которому вы
   подключаетесь.\n\n";
168.
169.               errorExit("Fatal Error", msg);
170.           }
171.         }
172.       }
```

ОБЯЗАТЕЛЬНО! Введите вместо 00:00:00:00:00 MAC вашего Bluetooth модуля, который можно узнать через Bluetooth терминал!!!

Ваше приложение готово. Теперь нам нужно проверить, как оно поведёт себя на устройстве. Запустите для этого симулятор. Если он запустился нормально без ошибок, то в папке, где вы создавали свой проект, будет создан файл с вашей программой. Его необходимо скопировать и установить на свое устройство.

Работа приложения

При нажатии на кнопку “Вкл” ваше Android-устройство передаст через Bluetooth цифру 1 и, как только Arduino примет цифру 1, светодиод загорится. При нажатии на кнопку “Выкл” передается цифра 2 и светодиод выключится, как показано на видео в начале статьи. Всё просто))

Мы будем очень рады, если вы поддержите наш ресурс и посетите магазин наших товаров shop.customelectronics.ru.

Передача данных Bluetooth

После успешного подключения к устройству Bluetooth каждое из них будет подключено BluetoothSocket. Теперь вы можете обмениваться информацией между устройствами. BluetoothSocketОбщая процедура передачи данных при использовании :

1. Получите InputStreamи OutputStream, которые обрабатывают передачу через сокет,
используя getInputStream() и getOutputStream()соответственно.
2. Чтение и запись данных в потоки с
помощью read(byte[])и write(byte[]).

Конечно, необходимо учитывать детали реализации. В частности, вам следует использовать выделенный поток для чтения из потока и записи в него. Это важно, поскольку оба
метода read(byte[])и write(byte[])блокируют
вызовы. Метод read(byte[])блокируется до тех пор, пока из потока не будет что-то прочитать. Этот write(byte[])метод обычно не блокируется, но он может блокироваться для управления потоком, если удаленное устройство не вызывает read(byte[]) достаточно быстро и в результате промежуточные буферы заполняются. Итак, вам следует посвятить свой основной цикл потока чтению из
файла InputStream. Вы можете использовать отдельный общедоступный метод в потоке, чтобы инициировать запись в
файл OutputStream.

Пример

Ниже приведен пример того, как можно передавать данные между двумя устройствами, подключенными через Bluetooth:

```
public class MyBluetoothService {
  private static final String TAG = "MY_APP_DEBUG_TAG";
  private Handler handler; // handler that gets info from Bluetooth
service

  // Defines several constants used when transmitting messages
between the
  // service and the UI.
```

```
private interface MessageConstants {
    public static final int MESSAGE_READ = 0;
    public static final int MESSAGE_WRITE = 1;
    public static final int MESSAGE_TOAST = 2;

    // ... (Add other message types here as needed.)
}

private class ConnectedThread extends Thread {
    private final BluetoothSocket mmSocket;
    private final InputStream mmInStream;
    private final OutputStream mmOutStream;
    private byte[] mmBuffer; // mmBuffer store for the stream

    public ConnectedThread(BluetoothSocket socket) {
        mmSocket = socket;
        InputStream tmpIn = null;
        OutputStream tmpOut = null;

        // Get the input and output streams; using temp objects because
        // member streams are final.
        try {
            tmpIn = socket.getInputStream();
        } catch (IOException e) {
            Log.e(TAG, "Error occurred when creating input stream", e);
        }
        try {
            tmpOut = socket.getOutputStream();
        } catch (IOException e) {
            Log.e(TAG, "Error occurred when creating output stream", e);
        }

        mmInStream = tmpIn;
        mmOutStream = tmpOut;
    }
```

```
public void run() {
    mmBuffer = new byte[1024];
    int numBytes; // bytes returned from read()

    // Keep listening to the InputStream until an exception occurs.
    while (true) {
        try {
            // Read from the InputStream.
            numBytes = mmInStream.read(mmBuffer);
            // Send the obtained bytes to the UI activity.
            Message readMsg = handler.obtainMessage(
                    MessageConstants.MESSAGE_READ, numBytes, -1,
                    mmBuffer);
            readMsg.sendToTarget();
        } catch (IOException e) {
            Log.d(TAG, "Input stream was disconnected", e);
            break;
        }
    }
}

// Call this from the main activity to send data to the remote device.
public void write(byte[] bytes) {
    try {
        mmOutStream.write(bytes);

        // Share the sent message with the UI activity.
        Message writtenMsg = handler.obtainMessage(
                MessageConstants.MESSAGE_WRITE, -1, -1, mmBuffer);
        writtenMsg.sendToTarget();
    } catch (IOException e) {
        Log.e(TAG, "Error occurred when sending data", e);

        // Send a failure message back to the activity.
        Message writeErrorMsg =
```

```
handler.obtainMessage(MessageConstants.MESSAGE_TOAST);
            Bundle bundle = new Bundle();
            bundle.putString("toast",
                  "Couldn't send data to the other device");
            writeErrorMsg.setData(bundle);
            handler.sendMessage(writeErrorMsg);
        }
    }

    // Call this method from the main activity to shut down the
connection.
    public void cancel() {
        try {
            mmSocket.close();
        } catch (IOException e) {
            Log.e(TAG, "Could not close the connect socket", e);
        }
    }
  }
}
```

После того как конструктор получит необходимые потоки, поток ожидает прохождения данных
через InputStream. При read(byte[])возврате данных из потока данные отправляются в основное действие с использованием
члена Handlerродительского класса. Затем поток ожидает чтения дополнительных байтов из файла InputStream.

Чтобы отправить исходящие данные, вы вызываете метод
потока write()из основного действия и передаете байты для отправки. Этот метод вызывает write(byte[])отправку данных на удаленное устройство. Если IOExceptionпри вызове выдается сообщение write(byte[]), поток отправляет всплывающее уведомление основному действию, объясняя пользователю, что устройство не может отправить данные байты другому (подключенному) устройству.

Метод потока cancel()позволяет вам разорвать соединение в любой момент, закрыв файл BluetoothSocket. Всегда вызывайте этот метод после завершения использования соединения Bluetooth.

MainActivity.java HC-05

```
package ru.kx13.extractvidid;

import java.io.IOException;
import java.io.OutputStream;
import java.util.UUID;

//import com.example.NAME.R;

import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.Toast;
import android.widget.TextView;

public class MainActivity extends Activity {
  private static final String TAG = "bluetooth1";

  Button button1, button2, button3, button4, button5;//Указываем id наших кнопок
  TextView text;

  private static final int REQUEST_ENABLE_BT = 1;
  private BluetoothAdapter btAdapter = null;
  private BluetoothSocket btSocket = null;
  private OutputStream outStream = null;

  // SPP UUID сервиса
  private static final UUID MY_UUID = UUID.fromString("00001101-0000-1000-8000-
00805F9B34FB");

  // MAC-адрес Bluetooth модуля
  private static String address = "98:DA:60:04:3E:AB";  //Вместо “00:00” Нужно нудет
ввести MAC нашего bluetooth

  /** Called when the activity is first created. */
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
```

```
    setContentView(R.layout.activity_main);
    button1 = (Button) findViewById(R.id.button1); //Добавляем сюда имена наших кнопок
    button2 = (Button) findViewById(R.id.button2);
    button3 = (Button) findViewById(R.id.button3);
    button4 = (Button) findViewById(R.id.button4);
    button5 = (Button) findViewById(R.id.button5);
    text   = (TextView)findViewById(R.id.my_text);
    text.setText("AZFOX Bluetooth HC-05");

    btAdapter = BluetoothAdapter.getDefaultAdapter();
    checkBTState();

    button1.setOnClickListener(new OnClickListener()  //Если будет нажата кнопка 1 то
    {
      public void onClick(View v)
      {
        sendData("1");         // Посылаем цифру 1 по bluetooth
        Toast.makeText(getBaseContext(), "Отправка 1", Toast.LENGTH_SHORT).show();
//выводим сообщение
      }
    });

    button2.setOnClickListener(new OnClickListener() {
      public void onClick(View v)
        {
        sendData("2"); // Посылаем цифру 2 по bluetooth
        Toast.makeText(getBaseContext(), "Отправка 2", Toast.LENGTH_SHORT).show();
      }
    });

    button3.setOnClickListener(new OnClickListener() {
      public void onClick(View v)
        {
        sendData(" AzonFox!!! "); // Посылаем текст по bluetooth
        Toast.makeText(getBaseContext(), "Отправка текста",
Toast.LENGTH_SHORT).show();
      }
    });

    button4.setOnClickListener(new OnClickListener() {
      public void onClick(View v)
        {
        sendData(" READ-??? "); // Посылаем текст по bluetooth
        Toast.makeText(getBaseContext(), "Прием текста", Toast.LENGTH_SHORT).show();
      }
    });

   button5.setOnClickListener(new OnClickListener() {
```

```
    public void onClick(View v)
      {
                // Завершаем работу
      sendData(" Exit Application! "); // Посылаем текст по bluetooth
      Toast.makeText(getBaseContext(), "Завершаем работу",
Toast.LENGTH_SHORT).show();
                finishAffinity();
                System.exit(0);
    }
   });

}



 @Override
 public void onResume() {
  super.onResume();

  Log.d(TAG, "...onResume - попытка соединения...");

  // Set up a pointer to the remote node using it's address.
  BluetoothDevice device = btAdapter.getRemoteDevice(address);

  // Two things are needed to make a connection:
  //   A MAC address, which we got above.
  //   A Service ID or UUID.  In this case we are using the
  //     UUID for SPP.
  try {
   btSocket = device.createRfcommSocketToServiceRecord(MY_UUID);
  } catch (IOException e) {
   errorExit("Fatal Error", "In onResume() and socket create failed: " + e.getMessage() + ".");
  }

  // Discovery is resource intensive.  Make sure it isn't going on
  // when you attempt to connect and pass your message.
  btAdapter.cancelDiscovery();

  // Establish the connection.  This will block until it connects.
  Log.d(TAG, "...Соединяемся...");
  try {
   btSocket.connect();
   Log.d(TAG, "...Соединение установлено и готово к передачи данных...");
  } catch (IOException e) {
   try {
    btSocket.close();
   } catch (IOException e2) {
```

```
      errorExit("Fatal Error", "In onResume() and unable to close socket during connection
failure" + e2.getMessage() + ".");
     }
    }

    // Create a data stream so we can talk to server.
    Log.d(TAG, "...Создание Socket...");

    try {
     outStream = btSocket.getOutputStream();
    } catch (IOException e) {
     errorExit("Fatal Error", "In onResume() and output stream creation failed:" +
e.getMessage() + ".");
    }
  }

  @Override
  public void onPause() {
    super.onPause();

    Log.d(TAG, "...In onPause()...");

    if (outStream != null) {
     try {
       outStream.flush();
     } catch (IOException e) {
       errorExit("Fatal Error", "In onPause() and failed to flush output stream: " + e.getMessage()
+ ".");
     }
    }

    try     {
     btSocket.close();
    } catch (IOException e2) {
     errorExit("Fatal Error", "In onPause() and failed to close socket." + e2.getMessage() + ".");
    }
  }

  private void checkBTState() {
    // Check for Bluetooth support and then check to make sure it is turned on
    // Emulator doesn't support Bluetooth and will return null
    if(btAdapter==null) {
     errorExit("Fatal Error", "Bluetooth не поддерживается");
    } else {
     if (btAdapter.isEnabled()) {
       Log.d(TAG, "...Bluetooth включен...");
     } else {
       //Prompt user to turn on Bluetooth
```

```
      Intent enableBtIntent = new Intent(btAdapter.ACTION_REQUEST_ENABLE);
      startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);
     }
    }
   }

   private void errorExit(String title, String message){
    Toast.makeText(getBaseContext(), title + " - " + message, Toast.LENGTH_LONG).show();
    finish();
   }

   private void sendData(String message) {
    byte[] msgBuffer = message.getBytes();

    Log.d(TAG, "...Посылаем данные: " + message + "...");

    try {
     outStream.write(msgBuffer);
    } catch (IOException e) {
     String msg = "In onResume() and an exception occurred during write: " + e.getMessage();
     if (address.equals("00:00:00:00:00:00"))
      msg = msg + ".\n\nВ переменной address у вас прописан 00:00:00:00:00:00, вам
необходимо прописать реальный MAC-адрес Bluetooth модуля";
      msg = msg +  ".\n\nПроверьте поддержку SPP UUID: " + MY_UUID.toString() + " на
Bluetooth модуле, к которому вы подключаетесь.\n\n";

      errorExit("Fatal Error", msg);
    }
   }
}
```

Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="ru.kx13.extractvidid"
        versionCode="2"
        versionName="0.1">
   <uses-sdk android:minSdkVersion="29"/>

   <uses-permission android:name="android.permission.BLUETOOTH"/>
   <uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>

   <application android:label="BT-HC05" android:icon="@drawable/icon" >
      <activity android:name=".MainActivity">
         <intent-filter>
            <action android:name="android.intent.action.MAIN"/>
            <category android:name="android.intent.category.LAUNCHER"/>
         </intent-filter>
      </activity>
   </application>
</manifest>
```

Layout

```
<?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:gravity="center"
   android:orientation="vertical"
   android:background="@color/bgwin">

   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:fontStyle="italic"
      android:fontFamily="@font/fsb"
      android:textSize="25sp"
      android:textColor="@color/textColor"
      android:id="@+id/my_text"/>

       <Button
            android:id="@+id/button1"
            android:background="@color/btn1"
            android:layout_marginTop="20sp"
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:text="BUT_1" />

       <Button
            android:id="@+id/button2"
            android:background="@color/btn1"
            android:layout_marginTop="20sp"
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:text="BUT-2" />

       <Button
            android:id="@+id/button3"
            android:background="@color/btn1"
            android:layout_marginTop="20sp"
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
```

```
            android:text="BUT-Text" />

        <Button
            android:id="@+id/button4"
            android:background="@color/btn2"
        android:textColor="@color/readColor"
            android:layout_marginTop="20sp"
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:text="READ" />

        <Button
            android:id="@+id/button5"
            android:background="@color/btn5"
            android:layout_marginTop="50sp"
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:text="EXIT" />
</LinearLayout>
```

---

color

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="bgwin">#6200EE</color>
    <color name="textColor">#0FF</color>
    <color name="btn1">#0F0</color>
    <color name="btn2">#00F</color>
    <color name="btn5">#F00</color>
    <color name="readColor">#FFF</color>
</resources>
```

---

string

```
<resources>
    <string name="btn2">Наш другойс текст</string>
</resources>
```