

## Передача данных по Bluetooth между Android и Arduino

<https://cxem.net/arduino/arduino64.php>

Мы рассмотрим два примера, в первом мы будем передавать данные от Android-устройства к arduino, а во втором примере мы рассмотрим двусторонний обмен данными между устройствами. Второй пример сложнее и в части понимания и по сложности кода, т.к. используются потоки (thread).

Мы будем использовать Java код, с явным указанием MAC-адреса устройства, к которому мы будем подключаться. Т.к. если делать интерфейс обнаружения Bluetooth-устройств, их выбора, подключения к ним и т.д., то код будет очень большой и для некоторых читателей труднопонимаем. Но для тех, кому интересно могут посмотреть стандартный пример Bluetooth Chat.

Узнать MAC-адрес можно к примеру в программе для Android'a: Bluetooth Terminal:



Нас интересует устройство BOLUTEK (наш модуль HC-06, подключенный к Arduino), его MAC адрес: 00:15:FF:F2:19:4C. Его и надо будет в дальнейшем прописать в программе.

### Android - передаем данные в Arduino

Первая программа очень простая, главное окно активности будет содержать 2 кнопки: включить LED и выключить LED. При нажатии на кнопку включения LED, по Bluetooth будет передаваться "1", при нажатии на выключение LED - "0".



В файле манифеста необходимо прописать 2 строки разрешения работы с Bluetooth.

Сам код главного активити:

```
?  
1 package com.example.bluetooth1;  
2  
3 import java.io.IOException;  
4 import java.io.OutputStream;  
5 import java.util.UUID;  
6  
7 import com.example.bluetooth1.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 btnOn, btnOff;  
25  
26     private static final int REQUEST_ENABLE_BT = 1;  
27
```

```

28     private BluetoothAdapter btAdapter = null;
29     private BluetoothSocket btSocket = null;
30     private OutputStream outputStream = null;
31
32     // SPP UUID сервиса
33     private static final UUID MY_UUID = UUID.fromString("000
34 1000-8000-00805F9B34FB");
35
36     // MAC-адрес Bluetooth модуля
37     private static String address = "00:15:FF:F2:19:4C";
38
39     /** Called when the activity is first created. */
40     @Override
41     public void onCreate(Bundle savedInstanceState) {
42         super.onCreate(savedInstanceState);
43
44         setContentView(R.layout.activity_main);
45
46         btnOn = (Button) findViewById(R.id.btnOn);
47         btnOff = (Button) findViewById(R.id.btnOff);
48
49         btAdapter = BluetoothAdapter.getDefaultAdapter();
50         checkBTState();
51
52         btnOn.setOnClickListener(new OnClickListener() {
53             public void onClick(View v) {
54                 sendData("1");
55                 Toast.makeText(getApplicationContext(), "Включаем LED",
56 Toast.LENGTH_SHORT).show();
57             }
58         });
59
60         btnOff.setOnClickListener(new OnClickListener() {
61             public void onClick(View v) {
62                 sendData("0");
63                 Toast.makeText(getApplicationContext(), "Выключаем LED"
64 Toast.LENGTH_SHORT).show();
65             }
66         });
67     }
68
69     @Override

```

```

70     public void onResume() {
71         super.onResume();
72
73         Log.d(TAG, "...onResume - попытка соединения...");
74
75         // Set up a pointer to the remote node using it's address
76         BluetoothDevice device = btAdapter.getRemoteDevice(address);
77
78         // Two things are needed to make a connection:
79         //     A MAC address, which we got above.
80         //     A Service ID or UUID.  In this case we are using the
81         //         UUID for SPP.
82         try {
83             btSocket = device.createRfcommSocketToServiceRecord(UUID.fromString("00001101-0000-1000-8000-00805F9B34FB"));
84         } catch (IOException e) {
85             errorExit("Fatal Error", "In onResume() and socket
86 failed: " + e.getMessage() + ".");
87         }
88
89         // Discovery is resource intensive.  Make sure it is
90         // when you attempt to connect and pass your message
91         btAdapter.cancelDiscovery();
92
93         // Establish the connection.  This will block until
94         Log.d(TAG, "...Соединяемся...");
95         try {
96             btSocket.connect();
97             Log.d(TAG, "...Соединение установлено и готово к передаче
98 данных...");
99         } catch (IOException e) {
100             try {
101                 btSocket.close();
102             } catch (IOException e2) {
103                 errorExit("Fatal Error", "In onResume() and unable to
104 socket during connection failure" + e2.getMessage() + ".");
105             }
106         }
107
108         // Create a data stream so we can talk to server.
109         Log.d(TAG, "...Создание Socket...");
110
111         try {

```

```

112         outputStream = btSocket.getOutputStream();
113     } catch (IOException e) {
114         errorExit("Fatal Error", "In onResume() and output
115 creation failed:" + e.getMessage() + ".");
116     }
117 }
118
119 @Override
120 public void onPause() {
121     super.onPause();
122
123     Log.d(TAG, "...In onPause()...");
124
125     if (outStream != null) {
126         try {
127             outStream.flush();
128         } catch (IOException e) {
129             errorExit("Fatal Error", "In onPause() and faile
130 output stream: " + e.getMessage() + ".");
131         }
132     }
133
134     try {
135         btSocket.close();
136     } catch (IOException e2) {
137         errorExit("Fatal Error", "In onPause() and failed
138 socket." + e2.getMessage() + ".");
139     }
140 }
141
142 private void checkBTState() {
143     // Check for Bluetooth support and then check to mak
144 turned on
145     // Emulator doesn't support Bluetooth and will retur
146     if (btAdapter == null) {
147         errorExit("Fatal Error", "Bluetooth не поддерживае
148     } else {
149         if (btAdapter.isEnabled()) {
150             Log.d(TAG, "...Bluetooth включен...");
151         } else {
152             //Prompt user to turn on Bluetooth
153             Intent enableBtIntent = new

```

```

        Intent(btAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(enableBtIntent, REQUEST_E
        }
    }
}

    private void errorExit(String title, String message){
        Toast.makeText(getBaseContext(), title + " - " + mess
Toast.LENGTH_LONG).show();
154     finish();
155 }
156
157 private void sendData(String message) {
158     byte[] msgBuffer = message.getBytes();
159
160     Log.d(TAG, "...Посылаем данные: " + message + "...");
161
162     try {
163         outputStream.write(msgBuffer);
164     } catch (IOException e) {
165         String msg = "In onResume() and an exception occur
166 write: " + e.getMessage();
167         if (address.equals("00:00:00:00:00:00"))
168             msg = msg + ".\n\nВ переменной address у вас про
169 00:00:00:00:00:00, вам необходимо прописать реальный MAC
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);
        }
    }
}

```

Данный код найден на одном из зарубежных блогов и слегка модернизирован. Как видно выше, на кнопки мы вешаем обработчики событий. При нажатии на кнопку передается строка 1 или 0 через sendData() в буфер Bluetooth адаптера. Полный проект с исходными кодами приведен ниже. Для работы программы, необходим Android не ниже версии API15, т.е. 4.0.3 и выше.

## Android - прием и передача данных к Arduino

А вот здесь пришлось повозиться. Дело в том, что в Android'е для приема данных от какого-либо устройства необходимо создавать отдельный фоновый поток, чтобы у нас не зависало основное активити. Для этого мы задействуем thread и все данные будут приниматься в отдельном потоке.

На окно главного активити мы добавим новый элемент TextView, который будет служить для отображения принятых данных от Arduino. Сам java-код главного активити я постарался хорошо прокомментировать, чтобы сделать его удобочитаемым:

```
?  
1 package com.example.bluetooth2;  
2  
3 import java.io.IOException;  
4 import java.io.InputStream;  
5 import java.io.OutputStream;  
6 import java.util.UUID;  
7  
8 import com.example.bluetooth2.R;  
9  
10 import android.app.Activity;  
11 import android.bluetooth.BluetoothAdapter;  
12 import android.bluetooth.BluetoothDevice;  
13 import android.bluetooth.BluetoothSocket;  
14 import android.content.Intent;  
15 import android.os.Bundle;  
16 import android.os.Handler;  
17 import android.util.Log;  
18 import android.view.View;  
19 import android.view.View.OnClickListener;  
20 import android.widget.Button;  
21 import android.widget.TextView;  
22 import android.widget.Toast;  
23  
24 public class MainActivity extends Activity {  
25     private static final String TAG = "bluetooth2";  
26  
27     Button btnOn, btnOff;  
28     TextView txtArduino;  
29     Handler h;
```

```

30
31 private static final int REQUEST_ENABLE_BT = 1;
32 final int RECIEVE_MESSAGE = 1;          // Статус для Hand
33 private BluetoothAdapter btAdapter = null;
34 private BluetoothSocket btSocket = null;
35 private StringBuilder sb = new StringBuilder();
36
37 private ConnectedThread mConnectedThread;
38
39 // SPP UUID сервиса
40 private static final UUID MY_UUID = UUID.fromString("0000
41 00805F9B34FB");
42
43 // MAC-адрес Bluetooth модуля
44 private static String address = "00:15:FF:F2:19:4C";
45
46 /** Called when the activity is first created. */
47 @Override
48 public void onCreate(Bundle savedInstanceState) {
49     super.onCreate(savedInstanceState);
50
51     setContentView(R.layout.activity_main);
52
53     btnOn = (Button) findViewById(R.id.btnOn);
54     btnOff = (Button) findViewById(R.id.btnOff);
55     // выключения
56     txtArduino = (TextView) findViewById(R.id.txtArduino);
57     // текста, полученного от Arduino
58
59     h = new Handler() {
60         public void handleMessage(android.os.Message msg) {
61             switch (msg.what) {
62                 case RECIEVE_MESSAGE:
63                     // если приняли сообщение в Handler
64                     byte[] readBuf = (byte[]) msg.obj;
65                     String strIncom = new String(readBuf, 0,
66                     sb.append(strIncom);
67                     // формируем строку
68                     int endOfLineIndex = sb.indexOf("\r\n");
69                     // определяем символы конца строки
70                     if (endOfLineIndex > 0)
71 {

```

```

// если встр

```



```

72             String sbprint = sb.substring(0, endC
73 // то извлекаем строку
74             sb.delete(0, sb.length());
75 // и очищаем sb
76             txtArduino.setText("Ответ от Arduino:
77 // обновляем TextView
78             btnOff.setEnabled(true);
79             btnOn.setEnabled(true);
80         }
81         //Log.d(TAG, "...Строка:" + sb.toString()
82 "...");
83         break;
84     }
85 };
86 };
87
88     btAdapter = BluetoothAdapter.getDefaultAdapter();
89 Bluetooth адаптер
90     checkBTState();
91
92     btnOn.setOnClickListener(new OnClickListener() {
93 обработчик при нажатии на кнопку
94     public void onClick(View v) {
95         btnOn.setEnabled(false);
96         mConnectedThread.write("1");    // Отправляем чер
97         //Toast.makeText(getBaseContext(), "Включаем LED'
98 Toast.LENGTH_SHORT).show();
99     }
100 });
101
102     btnOff.setOnClickListener(new OnClickListener() {
103     public void onClick(View v) {
104         btnOff.setEnabled(false);
105         mConnectedThread.write("0");    // Отправляем чер
106         //Toast.makeText(getBaseContext(), "Выключаем LED
107 Toast.LENGTH_SHORT).show();
108     }
109 });
110 }
111
112 @Override
113 public void onResume() {

```

```

114     super.onResume();
115
116     Log.d(TAG, "...onResume - попытка соединения...");
117
118     // Set up a pointer to the remote node using it's address
119     BluetoothDevice device = btAdapter.getRemoteDevice(address);
120
121     // Two things are needed to make a connection:
122     //     A MAC address, which we got above.
123     //     A Service ID or UUID. In this case we are using the
124     //         UUID for SPP.
125     try {
126         btSocket = device.createRfcommSocketToServiceRecord(UUID.fromString("00001101-0000-1000-8000-00805F9B34FB"));
127     } catch (IOException e) {
128         errorExit("Fatal Error", "In onResume() and socket creation failed: " + e.getMessage() + ".");
129     }
130
131
132     // Discovery is resource intensive. Make sure it isn't running
133     // when you attempt to connect and pass your message.
134     btAdapter.cancelDiscovery();
135
136     // Establish the connection. This will block until it succeeds
137     // or times out.
138     try {
139         btSocket.connect();
140         Log.d(TAG, "...Соединение установлено и готово к передаче...");
141     } catch (IOException e) {
142         try {
143             btSocket.close();
144         } catch (IOException e2) {
145             errorExit("Fatal Error", "In onResume() and unable to close socket during connection failure" + e2.getMessage() + ".");
146         }
147     }
148
149
150     // Create a data stream so we can talk to server.
151     Log.d(TAG, "...Создание Socket...");
152
153     mConnectedThread = new ConnectedThread(btSocket);
154     mConnectedThread.start();
155 }

```

```

156
157 @Override
158 public void onPause() {
159     super.onPause();
160
161     Log.d(TAG, "...In onPause()...");
162
163     try {
164         btSocket.close();
165     } catch (IOException e2) {
166         errorExit("Fatal Error", "In onPause() and failed t
167 e2.getMessage() + ".");
168     }
169 }
170
171 private void checkBTState() {
172     // Check for Bluetooth support and then check to make
173     // Emulator doesn't support Bluetooth and will return
174     if(btAdapter==null) {
175         errorExit("Fatal Error", "Bluetooth не поддерживает
176     } else {
177         if (btAdapter.isEnabled()) {
178             Log.d(TAG, "...Bluetooth включен...");
179         } else {
180             //Prompt user to turn on Bluetooth
181             Intent enableBtIntent = new Intent(btAdapter.ACTI
182             startActivityForResult(enableBtIntent, REQUEST_EM
183         }
184     }
185 }
186
187 private void errorExit(String title, String message){
188     Toast.makeText(getBaseContext(), title + " - " + mess
189 Toast.LENGTH_LONG).show();
190     finish();
191 }
192
193 private class ConnectedThread extends Thread {
194     private final BluetoothSocket mmSocket;
195     private final InputStream mmInStream;
196     private final OutputStream mmOutStream;
197

```

```

198     public ConnectedThread(BluetoothSocket socket) {
199         mmSocket = socket;
200         InputStream tmpIn = null;
201         OutputStream tmpOut = null;
202
203         // Get the input and output streams, using te
204         // member streams are final
205         try {
206             tmpIn = socket.getInputStream();
207             tmpOut = socket.getOutputStream();
208         } catch (IOException e) { }
209
210         mmInStream = tmpIn;
211         mmOutStream = tmpOut;
212     }
213
214     public void run() {
215         byte[] buffer = new byte[256]; // buffer sto
216         int bytes; // bytes returned from read()
217
218         // Keep listening to the InputStream until an
219         while (true) {
220             try {
221                 // Read from the InputStream
222                 bytes = mmInStream.read(buffer);
223                 и само сообщение в байтовый массив "buffer"
224                 h.obtainMessage(RECIEVE_MESSAGE, byte
225 buffer).sendToTarget(); // Отправляем в очередь сообщ
226             } catch (IOException e) {
227                 break;
228             }
229         }
230
231         /* Call this from the main activity to send data
232         public void write(String message) {
233             Log.d(TAG, "...Данные для отправки: " + messa
234             byte[] msgBuffer = message.getBytes();
235             try {
236                 mmOutStream.write(msgBuffer);
237             } catch (IOException e) {
238                 Log.d(TAG, "...Ошибка отправки данных: "

```

```

    "...");
    }
}

/* Call this from the main activity to shutdown t
public void cancel() {
    try {
        mmSocket.close();
    } catch (IOException e) { }
}
}
}
}

```

В данном примере для отправки данных мы используем отдельный поток Thread. Тоже самое и для приема данных - метод run(). Также обратите внимание на класс Handler, который служит для организации очереди сообщений и их вывода в главную активити. Дело в том, что в фоновом потоке нельзя напрямую выводить что-либо в главную активити, т.к. это приведет к "крашу" программы.

Класс StringBuilder используется для формирования строки из принятых данных. После, происходит поиск конца строки с символами \r\n, и если они найдены, то строка отображается на активити и объект sb очищается, чтобы не произошло склейка с последующими принятыми данными.

К статье прилагаются скомпилированные файлы для Android: bluetooth1.apk и bluetooth2.apk, а также исходники проекта для Arduino IDE и Eclipse

# package com.example.bluetooth1;

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

```
import com.example.bluetooth1.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;
```

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

```
    Button btnOn, btnOff;
```

```
    private static final int REQUEST_ENABLE_BT = 1;
    private BluetoothAdapter btAdapter = null;
    private BluetoothSocket btSocket = null;
    private OutputStream outputStream = 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 = "00:15:FF:F2:19:4C";
```

```
    /** Called when the activity is first created. */
```

```
    @Override
```

```
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```

setContentView(R.layout.activity_main);

btnOn = (Button) findViewById(R.id.btnOn);
btnOff = (Button) findViewById(R.id.btnOff);

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

btnOn.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
        sendData("1");
        Toast.makeText(getApplicationContext(), "Включаем LED",
Toast.LENGTH_SHORT).show();
    }
});

btnOff.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
        sendData("0");
        Toast.makeText(getApplicationContext(), "Выключаем LED",
Toast.LENGTH_SHORT).show();
    }
});
}

@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 {
        outputStream = 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 (outputStream != null) {
        try {
            outputStream.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);
            startActivityResult(enableBtIntent, REQUEST_ENABLE_BT);
        }
    }
}

```

```

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

```

```

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

```

```

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

```

```

try {
    outputStream.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\n";

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

```

---

```

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

```

```

<Button
    android:id="@+id/btnOff"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="true"
    android:text="@string/btn_OFF" />

```

```

<Button
    android:id="@+id/btnOn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@+id/btnOff"
    android:layout_centerHorizontal="true"
    android:text="@string/btn_ON" />

```

```

<ImageView
    android:id="@+id/imageView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"

```

```
    android:layout_alignParentBottom="true"
    android:layout_alignParentRight="true"
    android:alpha="0.5"
    android:src="@drawable/cxemnet_logo" />
```

```
</RelativeLayout>
```

---

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.bluetooth1"
    android:versionCode="1"
    android:versionName="1.0" >
```

```
    <uses-sdk
        android:minSdkVersion="15"
        android:targetSdkVersion="15" />
```

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

```
    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
```

```
</manifest>
```

## **package com.example.bluetooth2;**

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

```
import com.example.bluetooth2.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.os.Handler;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;
```

```
public class MainActivity extends Activity {
    private static final String TAG = "bluetooth2";
```

```
    Button btnOn, btnOff;
    TextView txtArduino;
    Handler h;
```

```
    private static final int REQUEST_ENABLE_BT = 1;
    final int RECIEVE_MESSAGE = 1;          // Статус для Handler
    private BluetoothAdapter btAdapter = null;
    private BluetoothSocket btSocket = null;
    private StringBuilder sb = new StringBuilder();
```

```
    private ConnectedThread mConnectedThread;
```

```
    // SPP UUID сервиса
```

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

```

// MAC-адрес Bluetooth модуля
private static String address = "00:15:FF:F2:19:4C";

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

    setContentView(R.layout.activity_main);

    btnOn = (Button) findViewById(R.id.btnOn);                // кнопка
    включения
    btnOff = (Button) findViewById(R.id.btnOff);              // кнопка
    выключения
    txtArduino = (TextView) findViewById(R.id.txtArduino);    // для
    вывода текста, полученного от Arduino

    h = new Handler() {
        public void handleMessage(android.os.Message msg) {
            switch (msg.what) {
                case RECIEVE_MESSAGE:
                    // если приняли сообщение в Handler
                    byte[] readBuf = (byte[]) msg.obj;
                    String strIncom = new String(readBuf, 0, msg.arg1);
                    sb.append(strIncom);
                    // формируем строку
                    int endOfLineIndex = sb.indexOf("\r\n");
                    // определяем символы конца строки
                    if (endOfLineIndex > 0) {
                        // если встречаем конец строки,
                        String sbprint = sb.substring(0, endOfLineIndex);
                        // то извлекаем строку
                        sb.delete(0, sb.length());
                        // и очищаем sb
                        txtArduino.setText("Ответ от Arduino: " + sbprint);    //
    обновляем TextView
                        btnOff.setEnabled(true);
                        btnOn.setEnabled(true);
                    }
                    //Log.d(TAG, "...Строка:" + sb.toString() + "Байт:" + msg.arg1 + "...");
                    break;
            }
        }
    };

```

```

};
};

btAdapter = BluetoothAdapter.getDefaultAdapter();    // получаем
локальный Bluetooth адаптер
checkBTState();

btnOn.setOnClickListener(new OnClickListener() {      // определяем
обработчик при нажатии на кнопку
    public void onClick(View v) {
        btnOn.setEnabled(false);
        mConnectedThread.write("1"); // Отправляем через Bluetooth цифру 1
        //Toast.makeText(getBaseContext(), "Включаем LED",
Toast.LENGTH_SHORT).show();
    }
});

btnOff.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
        btnOff.setEnabled(false);
        mConnectedThread.write("0"); // Отправляем через Bluetooth цифру 0
        //Toast.makeText(getBaseContext(), "Выключаем LED",
Toast.LENGTH_SHORT).show();
    }
});
}

@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...");

    mConnectedThread = new ConnectedThread(btSocket);
    mConnectedThread.start();
}

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

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

    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 class ConnectedThread extends Thread {
    private final BluetoothSocket mmSocket;
    private final InputStream mmInStream;
    private final OutputStream mmOutStream;

    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();
            tmpOut = socket.getOutputStream();
        } catch (IOException e) { }
    }

```



```

        mmInStream = tmpIn;
        mmOutStream = tmpOut;
    }

    public void run() {
        byte[] buffer = new byte[256]; // buffer store for the stream
        int bytes; // bytes returned from read()

        // Keep listening to the InputStream until an exception occurs
        while (true) {
            try {
                // Read from the InputStream
                bytes = mmInStream.read(buffer); // Получаем кол-во
байт и само сообщение в байтовый массив "buffer"
                h.obtainMessage(RECIEVE_MESSAGE, bytes, -1,
buffer).sendToTarget(); // Отправляем в очередь сообщений Handler
            } catch (IOException e) {
                break;
            }
        }
    }

    /* Call this from the main activity to send data to the remote device */
    public void write(String message) {
        Log.d(TAG, "...Данные для отправки: " + message + "...");
        byte[] msgBuffer = message.getBytes();
        try {
            mmOutStream.write(msgBuffer);
        } catch (IOException e) {
            Log.d(TAG, "...Ошибка отправки данных: " + e.getMessage() +
"...");
        }
    }

    /* Call this from the main activity to shutdown the connection */
    public void cancel() {
        try {
            mmSocket.close();
        } catch (IOException e) { }
    }
}

```

---

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
```

```
<Button
    android:id="@+id/btnOff"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="true"
    android:text="@string/btn_OFF" />
```

```
<Button
    android:id="@+id/btnOn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@+id/btnOff"
    android:layout_centerHorizontal="true"
    android:text="@string/btn_ON" />
```

```
<TextView
    android:id="@+id/txtArduino"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:text="" />
```

```
<ImageView
    android:id="@+id/imageView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_alignParentRight="true"
    android:alpha="0.5"
    android:src="@drawable/cxemnet_logo" />
```

```
</RelativeLayout>
```

---

```

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.bluetooth2"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="15"
        android:targetSdkVersion="15" />

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

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>

```