**蓝牙部分说明**

[一、整体概述 1](#_Toc1920)

[1.实现 1](#_Toc14753)

[2.使用的是传统蓝牙 2](#_Toc13731)

[3.蓝牙模块的UUID为 2](#_Toc9497)

[4.该项目将不支持与Android平板连接 2](#_Toc7222)

[5.平板做蓝牙客户端 2](#_Toc22164)

[6.主要参考文档 2](#_Toc14281)

[7.相关流程图 2](#_Toc21737)

[8.文件位置 3](#_Toc9792)

[二、Mixly测试代码 3](#_Toc14153)

[三、LuaActivity.java 6](#_Toc14195)

[1.onFinishCodeGeneration()方法 6](#_Toc5585)

[2. initReadBluethThread() 6](#_Toc13329)

[四、BluetoothActivity.java 6](#_Toc27509)

[1. judgeHaveBluetooth() 7](#_Toc5201)

[2. initBroadcast() 7](#_Toc23636)

[3. Timer类 7](#_Toc28022)

[4.btRestart 7](#_Toc11201)

[5.btBack 7](#_Toc21558)

[6.requestBluetoothPower() 7](#_Toc18311)

[五、ItemAdapter.java 8](#_Toc9082)

[1. addItem() 8](#_Toc14881)

[2. clearAllData() 8](#_Toc21019)

[3.onCreateViewHolder() 8](#_Toc9228)

[4. getItemCount() 8](#_Toc6023)

[5. onBindViewHolder() 8](#_Toc7265)

[6.类 MyViewHolder 8](#_Toc24167)

[六、ConnectThread 9](#_Toc6980)

[1.在构造方法中初始化mmSocket 9](#_Toc15821)

[2. run() 9](#_Toc25857)

[3.readMessage() 9](#_Toc13422)

[4.cancel() 9](#_Toc25281)

[七、其他 10](#_Toc32487)

[1.静态权限 10](#_Toc32322)

[2.每个活动都添加下面两个属性 10](#_Toc26935)

### 一、整体概述

1.实现Android平板端，将生成lua代码传输到蓝牙模块设备，并能够接受蓝牙设备传递回来的信息（但因为此点不是项目要求，且蓝牙设备无法同时实现接受后传递信息，所以只是放了一个接受线程，有需要的话需要继续完善）。

2.使用的是传统蓝牙，因为现在的蓝牙模块不支持BLE。

3.蓝牙模块的UUID为：

00001101-0000-1000-8000-00805f9b34fb

然而一般Android设备的UUID为：

0000110a-0000-1000-8000-00805f9b34fb //高级音频分发配置文件（A2DP） 连不上

0000111f-0000-1000-8000-00805f9b34fb //免提 连得上

00001112-0000-1000-8000-00805f9b34fb //耳机配置文件 连得上

这些UUID作用在网站：

https://www.bluetooth.com/specifications/assigned-numbers/service-discovery

所以在设置时，我设置的是

uuid=00001101-0000-1000-8000-00805f9b34fb

该项目将不支持与Android平板连接**，仅支持与蓝牙模块连接**

4.平板做蓝牙客户端，蓝牙模块做蓝牙服务器

5.主要参考文档：

Android开发官网：

<https://developer.android.com/guide/topics/connectivity/bluetooth.html?hl=zh-cn>

6.相关流程图：



7.文件位置：

主要在包：Bluetooth中：这个包位于：

Blockly-dev\blocklydemo\src\main\java\com\google\blockly\android\demo\Bluetooth

一部分代码在：LuaActivity.java中，文件位置：

Blockly-dev\src\blocklydemo\src\main\java\com\google\blockly\android\demo\LuaActivity.java

二、Mixly测试代码：

Mixly代码（实现串口接受和发送数据，用于验证Android平板端程序）：



|  |
| --- |
| #include <SoftwareSerial.h>  SoftwareSerial mySerial(8,9);  void setup()  {  Serial.begin(9600);  mySerial.begin(9600);  mySerial.println("AT");  delay(100);  mySerial.println("AT+NAME=hhh");  delay(100);  mySerial.println("AT+ROLE=0");  delay(100);  mySerial.println("AT+PSWD=hhh");  delay(100);  mySerial.println("AT+ROLE=0");  delay(100);  mySerial.println("AT+UART=9600,0,0");  delay(100);  mySerial.println("AT+RMAAD");  mySerial.begin(9600);  Serial.begin(9600);  }  void loop()  {  if (mySerial.available() > 0) {  Serial.write(mySerial.read());  }  if (Serial.available() > 0) {  mySerial.write(Serial.read());  }  } |

**注意：**

**在代码烧录主板将近结束时，需要不断点击蓝牙模块上那唯一的一个按钮；**

**在测试过程中，打开串口监视器。**

### 三、LuaActivity.java

文件位置：

Blockly-dev\src\blocklydemo\src\main\java\com\google\blockly\android\demo\LuaActivity.java

## 1.onFinishCodeGeneration()方法

(1)onFinishCodeGeneration()在点击运行后调用，生成lua代码，在此方法中添加判断是否已经连接蓝牙：

**if** (ConnectThread.*mmSocket* == **null** || !ConnectThread.*mmSocket*.isConnected())

(2)如果没有连接蓝牙，那么跳转BluetoothActivity，去连接蓝牙

(3)如果已经连接蓝牙，将生成代码通过蓝牙套接字mmSocket发送给蓝牙设备，接着如果传输失败那么提示用户重新连接，如果用户允许重连，到(2)

2. initReadBluethThread()方法可开启蓝牙接受线程，项目中未要求接受要求蓝牙信息，并且蓝牙模块接受和发送逻辑同时有的时候，其中至少一个始终会出错，所以没有调用。它将接受到的信息通过handler变量传递到主线程，并用Toast.make.....方法显现接受到的结果。

### 四、BluetoothActivity.java

文件位置：

Blockly-dev\src\blocklydemo\src\main\java\com\google\blockly\android\demo\Bluetooth\BluetoothActivity.java

在此活动中，打开蓝牙，扫描蓝牙设备，连接选择的蓝牙设备。

变量：

Button **btBack**, **btRestart**;//返回按钮、重新扫描蓝牙的按钮  
RecyclerView **recyclerView**;//显示扫描到的蓝牙设备的列表  
ItemAdapter **mAdapter**;//服务**recyclerView**  
List<BluetoothDevice> **deviceList**;//存放扫描的的设备的数据  
BluetoothAdapter **mBluetoothAdapter**;//系统扫描的蓝牙设备存放的位置  
BroadcastReceiver **queryReceiver**, **stateReceiver**;  
**boolean isClickOpen** = **false**;*//已经点击打开蓝牙*ProgressDialog **progressDialog**;//显示正在连接蓝牙  
ProgressBar **progressBar**;//显示正在扫描  
Handler **handler**;//服务**progressBar，停止扫描**  
Timer **timer**;//服务**progressBar，扫描指定时间长度**

1. judgeHaveBluetooth()方法判断平板是否支持蓝牙

如果不支持蓝牙，那么就进行提示，并且不再后续操作

如果支持蓝牙，那么此方法初始化mBluetoothAdapter

2. initBroadcast()方法动态设置广播接收器

（1）queryReceiver广播接收器，接受蓝牙的广播，每当查询到设备，便将设备添加到mAdapter

（2）stateReceiver广播接收器，监控蓝牙状态，在打开蓝牙时，开始扫描

（3）在onDestroy()中销毁广播接收器

3. Timer类：计时线程

实现每次点击扫描，重新计时，扫描指定时间长度后通过handler传递 扫描时间结束的信息，停止扫描

4.btRestart 重新扫描点击事件监听

如果mBluetoothAdapter可用，清空recyclerView列表，并停止之前的扫描，重新开始扫描蓝牙设备；

如果mBluetoothAdapter不可用，则打开蓝牙

5.btBack销毁活动，退出活动

## 6.requestBluetoothPower()

蓝牙扫描，大多数平板需要位置访问权限。

为实现 Androd 6.0及其以上访问位置权限权限，在此动态申请权限。

### 五、ItemAdapter.java

文件位置：

Blockly-dev\src\blocklydemo\src\main\java\com\google\blockly\android\demo\Bluetooth\ItemAdapter.java

为BluetoothActivity.java中的recyclerView服务，加载蓝牙设备列表。

变量：

List<BluetoothDevice> **deviceList**;//存放扫描到的设备  
List<String> **nameList**;//存放设备名称  
Context **context**;  
BluetoothAdapter **bluetoothAdapter**;  
ProgressDialog **progressDialog**;  
ConnectThread **connectThread**;//维持连接的线程  
**boolean isConnecting**=**false**;//是否连接

Handler handler=....//在这里定义了接受信息：连接蓝牙成功/失败

1. addItem()//添加列表子项

每扫描到一个设备，使用该方法添加子项

2. clearAllData()//清空列表子项

每次重新扫描时，调用此方法清空之前扫描的结果

3.onCreateViewHolder()//加载布局

4. getItemCount()//列表子项的数量

5. onBindViewHolder()//绑定数据和监听点击事件

itemView.setOnClickListener()中

获取点击位置的设备对象,在实例化connectThread时，将其传入

6.类 MyViewHolder //列表子项的类

其中只有一个用于显示设备名和MAC地址的TextView

### 六、ConnectThread

维持连接的线程，公共的静态变量，即蓝牙客户端套接字mmSocket供其他地方调用，用mmSocket传递或接受信息。

文件位置：

Blockly-dev\src\blocklydemo\src\main\java\com\google\blockly\android\demo\Bluetooth\ConnectBluetoothThread.java

变量

**public static boolean** *isConnect*=**false**;//是否连接  
**public static** BluetoothSocket *mmSocket*=**null**;//连接套接字  
**private final** BluetoothDevice **mmDevice**;//设备  
BluetoothAdapter **mBluetoothAdapter**;//系统存连接设备的对象  
**private static final** String ***TAG*** = **"ConnectThread"**;  
Handler **handler**;//向主线程传递消息的handler  
Message **msg**;//传递消息的载体

## 1.在构造方法中初始化mmSocket

此处注意UUID是1101 用于串口连接，两个平板或者手机之间不可用

1. run()

方法：线程开始运行

首先停止继续扫描蓝牙设备

尝试连接蓝牙，如果成功连接，isConnect=true;

并将消息传递给主线程。

如果连接失败，isConnect=false;并将连接失败的消息传递给主线程，并尝试关闭套接字，失败的话给主线程传递错误。

3.readMessage()读取接受信息，本项目没有调用改方法

4.cancel()关闭套接字

# 七、其他

1.静态权限：蓝牙与位置访问权限：

|  |
| --- |
| <**uses-permission android:name="android.permission.BLUETOOTH"** /> <**uses-permission android:name="android.permission.BLUETOOTH\_ADMIN"** /> <**uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION"**/> |

2.每个活动都添加下面两个属性（在AndroidManifest.xm中，）

|  |
| --- |
| **android:launchMode="singleTask" android:theme="@style/Theme.AppCompat.DayNight.NoActionBar"** |