Bluetooth Temperature Data Logger DevelopDoc For Android SDK

Status	□ Draft □ Review [✓ Publish] □ Revise		
Version	3.4.39		
Author	Forrest	**Date**	2022.11.15
Approver	Lisa	**Date**	

Catalogue

Preface

The Bluetooth Temperature data logger refers to BT04/BT04B/BT05/BT05B/BT06/BT03/TempU06 L60/TempU06 L100/TempU06 L200 series devices. This document introduces APP development by using Java language on Android platform.

I. Using Android Studio to create a new project

Put com.tzone.btlogger.jar package into the libs folder, and then add content to dependencies in the build.gradle file under the present project, as follow compile files('libs/com.tzone.btlogger.jar')

Android Manifest file specification:

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

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

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

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

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

<uses-feature android:name="android.hardware.bluetooth_le" android:required="true" />
```

II. Initialize SDK

Initialize on the APP startup interface,as MainActivity interface

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


        ……
        BleManager.getInstance().init(getApplication());
                BleManager.getInstance()
                                .setReConnectCount(3, 5000)
```

```
                        .setConnectOverTime(20000)
                        .setOperateTimeout(5000);
            ……
    }
```

Reference sample program: com/tzone/btloggerexample/MainActivity.java

III. Search device

1. Scan device broadcast

As follow:

```
BleScanRuleConfig scanRuleConfig = new BleScanRuleConfig.Builder()
    .setScanTimeOut(1000 * 60).build();
BleManager.getInstance().initScanRule(scanRuleConfig);
BleManager.getInstance().scan(new BleScanCallback() {
        @Override
        public void onScanFinished(List<BleDevice> scanResultList) {
            Log.i(TAG, "onScanFinished => " + scanResultList.size());
        }

        @Override
        public void onScanStarted(boolean success) {
            Log.i(TAG, "onScanStarted => " + success);
        }

        @Override
        public void onScanning(BleDevice bleDevice) {
            if (bleDevice == null)
```

```
                return;

            Log.i(TAG, "onScanning => " + bleDevice.getMac() + " " +
        bleDevice.getName());

            Scan device = new Scan();

            device.fromBroadcast(bleDevice) ;

            //query condition

        }

  });
```

Reference sample program: com/tzone/btloggerexample/MainActivity.java

2. Specify device ID

Eg: search specify device ID = 012345678, as follow:

```
String deviceId = "012345678";
if(device.getID().equals(deviceId)){
    ……
}
```

3. Specify type of device list

Eg: search specify device type is TempU06 L60, as follow:

```
if(device.getDeviceType() == DeviceType.TempU06L60){
    ……
}
```

Description:

1.The value is -1000,it means null

IV. Extract device data report

1.Connect device

```
BleManager.getInstance().connect(_Device.getMac(), new BleGattCallback() {
        @Override
        public void onStartConnect() {
            Log.i(TAG, "onStartConnect: OK");
        }

        @Override
        public void onConnectFail(BleDevice bleDevice, BleException exception) {
            Log.i(TAG, "onConnectFail: bleDevice => " + (bleDevice != null ? bleDevice.getMac() : " null"));
            BleManager.getInstance().destroy();
        }

        @Override
        public void onConnectSuccess(BleDevice bleDevice, BluetoothGatt gatt, int status) {
            Log.i(TAG, "onConnectSuccess: bleDevice => " + (bleDevice != null ? bleDevice.getMac() : " null"));
            _BleDevice = bleDevice;
            InitDataManager();
        }

        @Override
        public void onDisConnected(boolean isActiveDisConnected, BleDevice device, BluetoothGatt gatt, int status) {
            Log.i(TAG, "onDisConnected: bleDevice => " + (device != null ? device.getMac() : " null"));
            _BleDevice = null;
        }
    });
```

2.Initialize DataManager object

```
public void InitDataManager() {
        if (_BleDevice == null)
            return;
        if (_DataManager == null) {
            if (_Device.getDeviceType() == DeviceType.TempU06L60)
              _DataManager = new com.tzone.bt.u06L60.DataManager();
        else if (_Device.getDeviceType() == DeviceType.TempU06L100)
              _DataManager = new com.tzone.bt.u06L100.DataManager();
          else if (_Device.getDeviceType() == DeviceType.TempU06L200)
                _DataManager = new com.tzone.bt.u06L200.DataManager();
            else if (_Device.getDeviceType() == DeviceType.BT04)
                _DataManager = new com.tzone.bt.bt04.DataManager();
            else if (_Device.getDeviceType() == DeviceType.BT04B)
                _DataManager = new com.tzone.bt.bt04b.DataManager();
            else if (_Device.getDeviceType() == DeviceType.BT05)
                _DataManager = new com.tzone.bt.bt05.DataManager();
            else if (_Device.getDeviceType() == DeviceType.BT05B)
                _DataManager = new com.tzone.bt.bt05b.DataManager();
            else if (_Device.getDeviceType() == DeviceType.BT06)
                _DataManager = new com.tzone.bt.bt06.DataManager();
            else if (_Device.getDeviceType() == DeviceType.BT03)
                _DataManager = new com.tzone.bt.bt03.DataManager();
            else {
                return;
            }

            _DataManager.InitSetting(_BleDevice, dataCallback);
        }

        _Report = null;
        if (_Device.getDeviceType() == DeviceType.TempU06L60
                    || _Device.getDeviceType() == DeviceType.TempU06L80
                    || _Device.getDeviceType() == DeviceType.TempU06L100
                    || _Device.getDeviceType() == DeviceType.TempU06L200
                    || _Device.getDeviceType() == DeviceType.BT06
                    || _Device.getDeviceType() == DeviceType.BT03){
            _DataManager.Notify();
        }else {
            _DataManager.Unlock(Password);
        }
}
```

3.Open the notification

_DataManager.Notify();

Description: BT04/BT04B/BT05/BT05B no notification need to be opened

4.Unlock

_DataManager.Unlock(Password);

Description: BT04/BT04B/ BT05/BT05B must be unlocked to operate. TempU06 series whether or not to unlock according to the password level. Please refer to the device user manual for details.

5.Get device status

_DataManager.GetLogStatus();

```
//Device status type
enum DeviceRecordType {
    Initialize, //Initialization state
    Stop, //Stop (device button stop)
    Stop_USB, //USB stop
    Stop_StorageFull, //Storage full stop
    Stop_App, //APP stop
    Recording, //recording
    Delay, //delay
}
```

6.Set extract time period

```
long beginTime = 0;
long endTime = 0;
_DataManager.SetConfig(beginTime, endTime );
```

Description: when the value is 0, it means all data extracted from the device

7.Get alarm setting

```
_DataManager.GetAlarm();
```

8.Get Mark setting

```
_DataManager.GetMark();
```

9.Get extract data information

```
_DataManager.RequestDataInfo();
```

```
String[] info = new String[13];
info[0] // Data Count
info[1] // Begin Time
info[2] // Time Span
info[3] // Record Status
info[4] // Delay Time
info[5] // Repeat Start
info[6] // Temperature Unit
info[7] // Stop Button
info[8] // Start Mode
info[9] // Start Time
info[10] // Description
info[11] // NOTE
info[12] // MKT
```

10.Open receive

```
_DataManager.Receive(true);
```

11.Close receive

```
_DataManager.Receive(false);
```

12.Generate report

```
_Report.Generate();
```

13.The introduction of other function

	Function	Function name	BT04	BT04B	BT05	BT05B	TempU06 series	BT06	BT03	
1	Start record	SetStart	V	V	V	V	V	V	V	
2	Stop record	SetStop	V	V	V	V	V	V	V	
3	Set Mark	SetMark	X	X	X	X	V	X	X	Do not support models of BT 04/BT04B/BT05/BT05B
4	Set flight mode	SetFligthMode	X	X	X	X	V	X	X	

Reference sample program: com/tzone/btloggerexample/DeviceActivity.java

V. Modify device parameter

1. Connect device

```
BleManager.getInstance().connect(_Device.getMac(), new BleGattCallback() {
        @Override
        public void onStartConnect() {
            Log.i(TAG, "onStartConnect: OK");
        }

        @Override
        public void onConnectFail(BleDevice bleDevice, BleException exception) {
            Log.i(TAG, "onConnectFail: bleDevice => " + (bleDevice != null ? bleDevice.getMac() : " null"));
            BleManager.getInstance().destroy();
        }

        @Override
        public void onConnectSuccess(BleDevice bleDevice, BluetoothGatt gatt, int status) {
            Log.i(TAG, "onConnectSuccess: bleDevice => " + (bleDevice != null ? bleDevice.getMac() : " null"));
            _BleDevice = bleDevice;
```

```
                InitConfigManager();
            }

            @Override
            public void onDisConnected(boolean isActiveDisConnected, BleDevice device, 
BluetoothGatt gatt, int status) {
                Log.i(TAG, "onDisConnected: bleDevice => " + (device != null ? device.getMac() : 
" null"));
                _BleDevice = null;
            }
        });
```

2. Initialize ConfigManager object

```
public void InitConfigManager() {
            if (_BleDevice == null)
                return;
            ConfigManagerBase _ConfigManager = null;
            if (_ConfigManager == null) {
                if (_Device.getDeviceType() == DeviceType.TempU06L60)
                    _ConfigManager = new com.tzone.bt.u06L60.ConfigManager();
                else if (_Device.getDeviceType() == DeviceType.TempU06L100)
                    _ConfigManager = new com.tzone.bt.u06L100.ConfigManager();
                else if (_Device.getDeviceType() == DeviceType.TempU06L200)
                    _ConfigManager = new com.tzone.bt.u06L200.ConfigManager();
                else if (_Device.getDeviceType() == DeviceType.BT04)
                    _ConfigManager = new com.tzone.bt.bt04.ConfigManager();
                else if (_Device.getDeviceType() == DeviceType.BT04B)
                    _ConfigManager = new com.tzone.bt.bt04b.ConfigManager();
                else if (_Device.getDeviceType() == DeviceType.BT05)
                    _ConfigManager = new com.tzone.bt.bt05.ConfigManager();
                else if (_Device.getDeviceType() == DeviceType.BT05B)
                    _ConfigManager = new com.tzone.bt.bt05b.ConfigManager();
                else if (_Device.getDeviceType() == DeviceType.BT06)
                    _ConfigManager = new com.tzone.bt.bt06.ConfigManager();
                else if (_Device.getDeviceType() == DeviceType.BT03)
                    _ConfigManager = new com.tzone.bt.bt03.ConfigManager();
                else {
                    return;
                }
                _ConfigManager.InitSetting(_BleDevice, configCallback);
            }

            if (_Device.getDeviceType() == DeviceType.TempU06L60
                        || _Device.getDeviceType() == DeviceType.TempU06L100
                        || _Device.getDeviceType() == DeviceType.TempU06L200
                        || _Device.getDeviceType() == DeviceType.BT06
                        || _Device.getDeviceType() == DeviceType.BT03){
                _ConfigManager.Notify();
            }else {
                _ConfigManager.Unlock(Password);
            }


        }
```

3. Open the notification

_DataManager.Notify();

Description: BT04/BT04B/BT05/BT05B no notification need to be opened

4. Unlock

_DataManager.Unlock(Password);

Description: BT04/BT04B/BT05/BT05B must be unlocked to operate. TempU06 series must be unlocked to operate in low/high level encryption.

5. The introduction of read,modify function parameter

	Function	Function name	BT04	BT04B	BT05	BT05B	TempU06 series	BT06	BT03	
1	Get device information	GetDeviceInfo	V	V	V	V	V	V	V	
2	Set password	SetPassword	V	V	V	V	V	V	V	
3	Set device name	SetDeviceName	V	V	V	V	V	V	V	
4	Set device time	SetDateTime	V	V	V	V	V	V	V	
5	Get device time	GetDateTime	V	V	V	V	V	V	V	
6	Set broadcast parameter	SetBroadcastSetting	V	V	V	V	X	V	V	TempU06 series do not support
7	Get broadcast parameter	GetBroadcastSetting	V	V	V	V	X	V	V	
8	Set log parameter	SetLogSetting	V	V	V	V	V	V	V	
9	Get log parameter	GetLogSetting	V	V	V	V	V	V	V	
10	Set PDF parameter	SetPDFSetting	V	V	V	V	V	X	X	BT04/BT04B/BT05/BT05B can set description and remark only
11	Get PDF Parameter	GetPDFSetting	V	V	V	V	V	X	X	

12	Set alarm parameter	SetAlarm	V	V	V	V	V	V	V	
13	Get alarm paramete	GetAlarm	V	V	V	V	V	V	V	

```
    /**
     * Get device information
     * @param status
     * @param deviceName
     * @param mac
     * @param hardwareType
     * @param version
     * @param locklevel
     *                        0 = unlocked
     *                        1 = low
     *                        2 = high
     */
    public void onDeviceInfo(boolean status,String deviceName, String mac, String hardwareType,
String version,int locklevel);

    /**
     * Set Password
     *
     * @param level
     * @param password    The six-digit password
     */
    public void SetPassword(int level, String password);

    /**
     * Set DeviceName
     * @param name
     */
    public void SetDeviceName(String name);

    /**
     * Set DateTime
     * Note: BT04/BT04B/BT05/BT05B setting parameters are invalid, do not support
customization, set to the current time of the phone.
     * @param timezone
     * @param dst           Daylight Saving Time
     * @param format
     *                         （MM/DD/YY）：0
     *                         （YY/MM/DD）：1
```

```
  *                              （DD /MM/YY）： 2
  * @param timestamp
  */
public void SetDateTime(int timezone, boolean dst, int format, long timestamp);
public void GetDateTime();

/**
  * Set BroadcastSetting
  * @param broadcastInterval
  * @param transmitPower
* @param transmitRate
  */
public void SetBroadcastSetting(long broadcastInterval, int transmitPower,int transmitRate);
public void GetBroadcastSetting();

/**
  * SetLogSetting
  * @param logInterval
  * @param startDelay
  * @param repeatStart
  * @param fullCoverage
  * @param unit Temperature of the unit
  *                        0 = degree centigrade
  *                        1 = Fahrenheit
  * @param disableStopButton
  * @param startMode
  * @param startTime
  */
public void SetLogSetting(long logInterval, long startDelay, boolean repeatStart,boolean
fullCoverage, int unit, boolean disableStopButton, int startMode, long startTime);
public void GetLogSetting();

/**
  * set PDF
  * @param language
  * @param showDataLis
  * @param description
  */
public void SetPDFSetting(String language,boolean showDataLis,String description);
public void GetPDFSetting();

/**
  * SetAlarm
  * @param tempAlarmList
  * @param rhAlarmList
  */
public void SetAlarm(List<AlarmSetting> tempAlarmList,List<AlarmSetting> rhAlarmList);
public void GetAlarm();
```

Description:1. Temperature, Humidity, Power, Voltage= -1000,it means null

2. Time = 0, it means time is error

3. Different types have different functions,if the device does not have this parameter interface, the return is null, please refer to the protocol document and operation manual for details.

VI. Cautions

1. The BLE function can be used only on Android version 4.3 and above, the location permission must be obtained to use the scan method on Android 6.0 and above, some mobile phone need to turn on the location function.
2. It is recommended to have a period of time between the two operations. For example, after the connection is successful, notify or write at interval of 100ms. For specific time data, you can try to select the shortest effective time on different mobile phone.