BleManager 类，持有 BleBluetooth、DefaultBleExceptionHandler、Context 的引用。

BleBluetooth是实现类。

BleManager 的功能：

扫描设备 scanDevice(ListScanCallback callback)

连接设备connectDevice(ScanResult scanResult,boolean autoConnect,BleGattCallback callback)

扫描并且连接scanNameAndConnect(String deviceName,long time\_out,boolean autoConnect,

BleGattCallback callback)

扫描并且连接（多个）scanNamesAndConnect(String[] deviceNames,long time\_out,

boolean autoConnect, BleGattCallback callback)

模糊扫描并连接scanfuzzyNameAndConnect(String fuzzyName,long time\_out,

boolean autoConnect, BleGattCallback callback)

扫描并连接（Mac）scanMacAndConnect(String deviceMac,long time\_out,

boolean autoConnect, BleGattCallback callback)

Notify操作notify(String uuid\_service, String uuid\_notify, BleCharacterCallback callback)

Indicate操作indicate(String uuid\_service, String uuid\_indicate, BleCharacterCallback callback)

停止Notify操作stopNotify(String uuid\_service, String uuid\_notify)

停止Indicate操作stopIndicate(String uuid\_service, String uuid\_indicate)

写数据操作 writeDevice(String uuid\_service, String uuid\_write,

byte[] data, BleCharacterCallback callback)

读数据操作 readDevice(String uuid\_service, String uuid\_read,

BleCharacterCallback callback)

来看这些操作的实现

实现类 BleBluetooth

两个字符串常量

private static final String CONNECT\_CALLBACK\_KEY = "connect\_key";

public static final String READ\_RSSI\_KEY = "rssi\_key";

状态常数

private static final int STATE\_DISCONNECTED = 0;

private static final int STATE\_SCANNING = 1;

private static final int STATE\_CONNECTING = 2;

private static final int STATE\_CONNECTED = 3;

private static final int STATE\_SERVICES\_DISCOVERED = 4;

private int connectionState = STATE\_DISCONNECTED; //当前状态

操作类

private Context context;

private BluetoothAdapter bluetoothAdapter;

private BluetoothGatt bluetoothGatt;

线程

private Handler handler = new Handler(Looper.getMainLooper());

HashMap 连接的集合

private HashMap<String, BluetoothGattCallback> callbackHashMap = new HashMap<>();

private PeriodScanCallback periodScanCallback;

实现了扫描startLeScan(PeriodScanCallback callback)：

this.periodScanCallback = callback;

callback.setBleBluetooth(this).notifyScanStarted();

boolean success = bluetoothAdapter.startLeScan(callback);//最终调用了adapter.startLeScan(callback)

if (success) {

connectionState = STATE\_SCANNING;

} else {

callback.removeHandlerMsg();

}

return success;

还有停止扫描stopScan(BluetoothAdapter.LeScanCallback callback)

实现了连接 public synchronized BluetoothGatt connect(ScanResult scanResult,

boolean autoConnect, BleGattCallback callback)

最终实现类 BleConnector

private static final String TAG = BleConnector.class.getSimpleName();

private static final String UUID\_CLIENT\_CHARACTERISTIC\_CONFIG\_DESCRIPTOR = "00002902-0000-1000-8000-00805f9b34fb";

private static final int MSG\_WRITE\_CHA = 1;

private static final int MSG\_WRIATE\_DES = 2;

private static final int MSG\_READ\_CHA = 3;

private static final int MSG\_READ\_DES = 4;

private static final int MSG\_READ\_RSSI = 5;

private static final int MSG\_NOTIFY\_CHA = 6;

private static final int MSG\_NOTIY\_DES = 7;

private static final int MSG\_INDICATE\_DES = 8;

private BluetoothGatt bluetoothGatt;

private BluetoothGattService service;

private BluetoothGattCharacteristic characteristic;

private BluetoothGattDescriptor descriptor;

private BleBluetooth bleBluetooth;

private static int timeOutMillis = 10000;

private Handler handler = new MyHandler();

BleConnector有四个构造方法。