主要操作类 BluetoothClient ：实现了 IBluetoothClient 接口，用于连接、读写、等蓝牙的基本功能以及蓝牙状态的监听，持有BluetoothClientImpl 的引用，BluetoothClientImpl是BluetoothClient的实现类。

BluetoothClientImpl 实现了IBluetoothClient, ProxyInterceptor, Callback，是真正的实现类。核心方法是safeCallBluetoothApi(int code, Bundle args, final BluetoothResponse response)，该方法调用了service.callBluetoothApi(code, args, response); service为Iservice接口，实现类是 BluetoothServiceImpl ,callBluetoothApi()，msg携带code和args作为参数，统一给handlerMessage(Message msg) 处理。

public boolean handleMessage(Message msg) {

Bundle args = msg.getData();

String mac = args.getString(EXTRA\_MAC);

UUID service = (UUID) args.getSerializable(EXTRA\_SERVICE\_UUID);

UUID character = (UUID) args.getSerializable(EXTRA\_CHARACTER\_UUID);

UUID descriptor = (UUID) args.getSerializable(EXTRA\_DESCRIPTOR\_UUID);

byte[] value = args.getByteArray(EXTRA\_BYTE\_VALUE);

BleGeneralResponse response = (BleGeneralResponse) msg.obj;

switch (msg.what) {

case CODE\_CONNECT:

BleConnectOptions options = args.getParcelable(EXTRA\_OPTIONS);

BleConnectManager.connect(mac, options, response);

break;

case CODE\_DISCONNECT:

BleConnectManager.disconnect(mac);

break;

case CODE\_READ:

BleConnectManager.read(mac, service, character, response);

break;

case CODE\_WRITE:

BleConnectManager.write(mac, service, character, value, response);

break;

case CODE\_WRITE\_NORSP:

BleConnectManager.writeNoRsp(mac, service, character, value, response);

break;

case CODE\_READ\_DESCRIPTOR:

BleConnectManager.readDescriptor(mac, service, character, descriptor, response);

break;

case CODE\_WRITE\_DESCRIPTOR:

BleConnectManager.writeDescriptor(mac, service, character, descriptor, value, response);

break;

case CODE\_NOTIFY:

BleConnectManager.notify(mac, service, character, response);

break;

case CODE\_UNNOTIFY:

BleConnectManager.unnotify(mac, service, character, response);

break;

case CODE\_READ\_RSSI:

BleConnectManager.readRssi(mac, response);

break;

case CODE\_SEARCH:

SearchRequest request = args.getParcelable(EXTRA\_REQUEST);

BluetoothSearchManager.search(request, response);

break;

case CODE\_STOP\_SESARCH:

BluetoothSearchManager.stopSearch();

break;

case CODE\_INDICATE:

BleConnectManager.indicate(mac, service, character, response);

break;

case CODE\_CLEAR\_REQUEST:

int clearType = args.getInt(EXTRA\_TYPE, 0);

BleConnectManager.clearRequest(mac, clearType);

break;

case CODE\_REFRESH\_CACHE:

BleConnectManager.refreshCache(mac);

break;

}

return true;

}

BleConnectWorker是主要的蓝牙实现类，主要实现了Handler.Callback, IBleConnectWorker, IBluetoothGattResponse, ProxyInterceptor, RuntimeChecker 接口。

蓝牙连接的BleConnectOptionsion 可以设置蓝牙重连次数，蓝牙响应超时设置，这个是怎么实现的？

相关代码：BleConnectOptions options = new BleConnectOptions.Builder()

.setConnectRetry(3)

.setConnectTimeout(20000)

.setServiceDiscoverRetry(3)

.setServiceDiscoverTimeout(10000)

.build();

ClientManager.getClient().connect(mDevice.getAddress(), options, new BleConnectResponse() {

@Override

public void onResponse(int code, BleGattProfile profile) {

BluetoothLog.v(String.format("profile:\n%s", profile));

mTvTitle.setText(String.format("%s", mDevice.getAddress()));

mPbar.setVisibility(View.GONE);

mListView.setVisibility(View.VISIBLE);

if (code == REQUEST\_SUCCESS) {

mAdapter.setGattProfile(profile);

}

}

});

ProxyUtils