

# Android BLE in Action

吴晶

@Android笔记

# Outline

- What's Bluetooth Low Energy (BLE)
- BLE on Android
- BLE Applications
- Debug tools
- Best practice

# Bluetooth intro.

- What's Bluetooth™
  - Bluetooth SIG
  - Feature
    - Short distance
    - Up to 24Mbps
    - Multiple connections
  - History
    - v1.0 → v2.1 → v3.0 → **v4.0** → v5.0

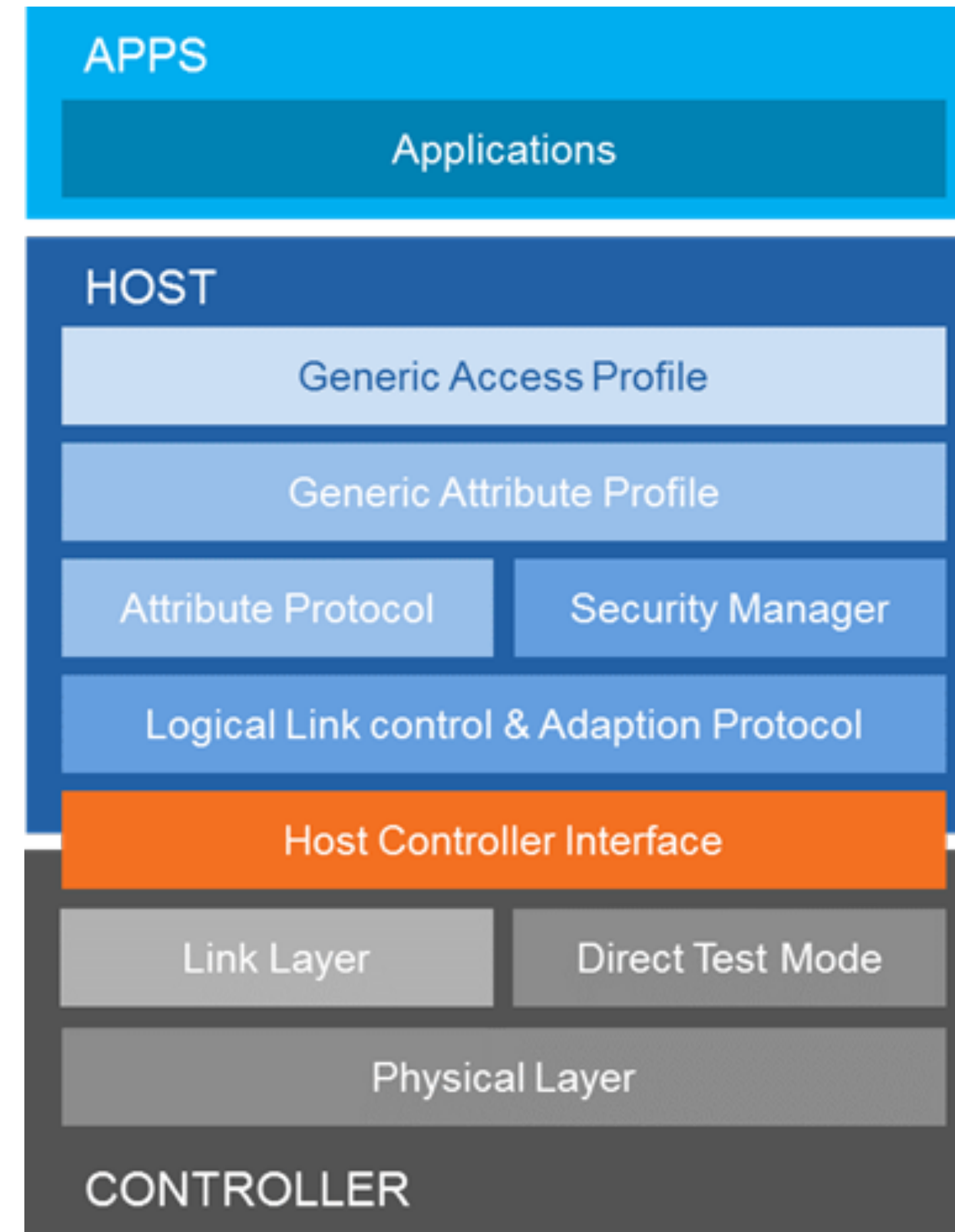


# What's BLE

- What's Bluetooth™ Low Energy
  - Low power
    - 15mA peak transmit, 1uA sleep
  - Low cost
  - Low latency connection (3ms)
  - High flexible (Multi-model)
  - High security (128bit AES CCM)

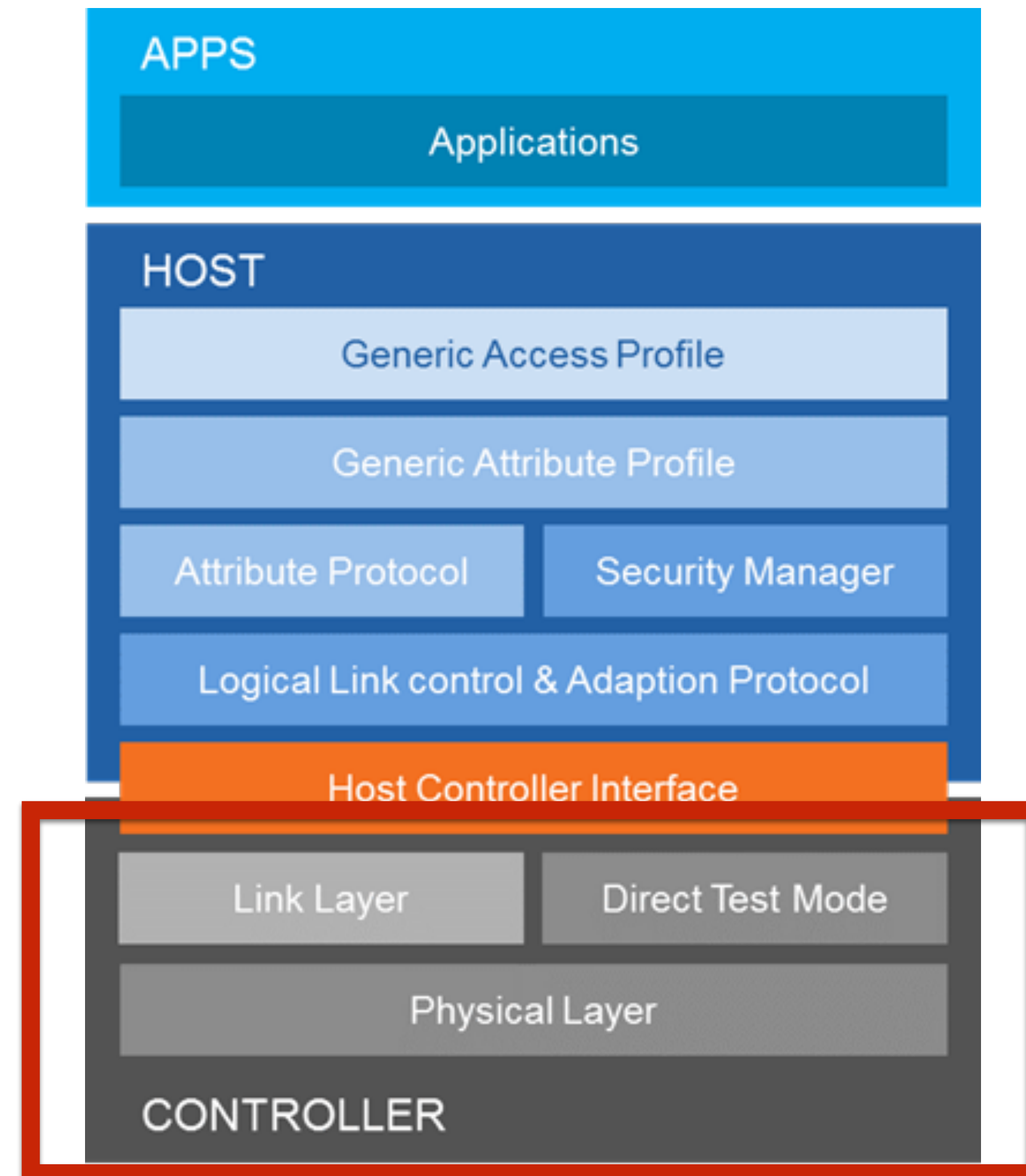
# BLE stack — Arch.

- Controller
- Host
- APPS



# BLE stack — Controller

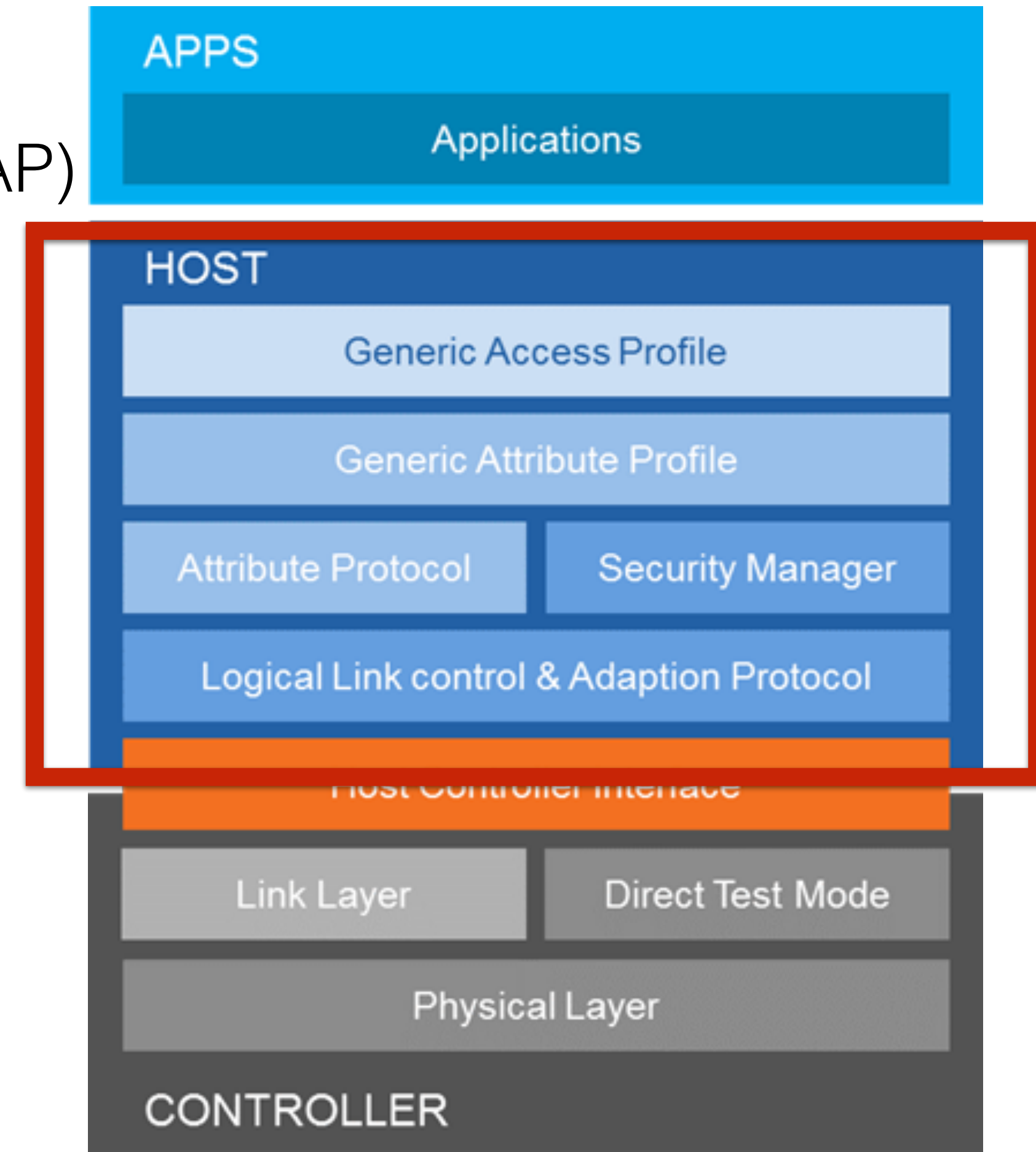
- Controller
  - Physical Layer
  - Link Layer
- HCI
- Host Controller interface





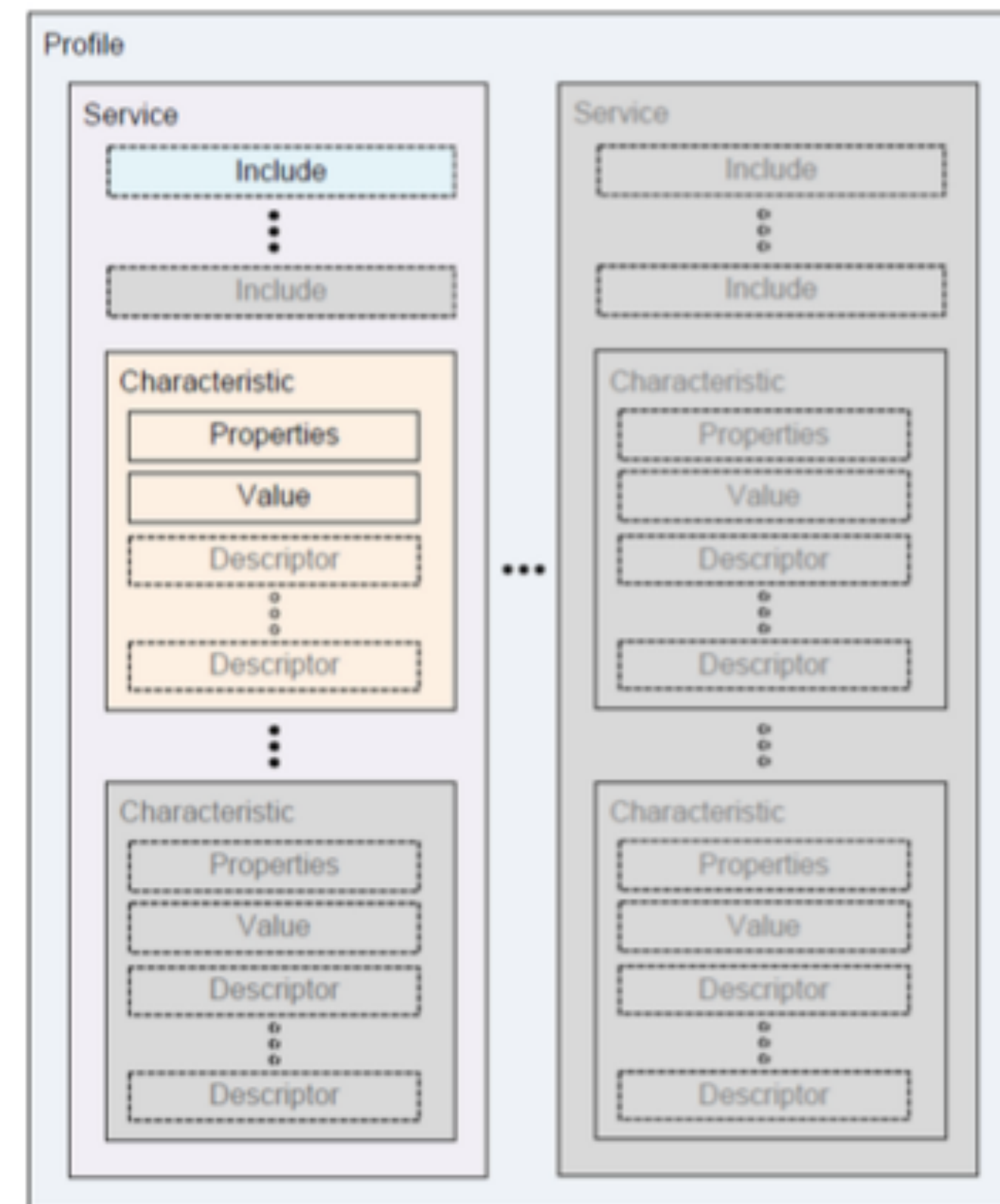
# BLE stack — Host

- Host
  - Logical Link control & Adaption Protocol(L2CAP)
  - Security Manager
  - Attribute Protocol(ATT)
    - UUID
    - Simple — design for low energy
    - Data as attribute
    - C-S Architecture



# What's BLE — ATT/GATT

- Host
  - Generic Attribute Profile (GATT)
    - Characteristic
      - One value, n descriptors
      - read/write/notify/indicate
    - Service
      - n characteristics
    - Profile
      - n services
      - Define by Bluetooth SIG





# UUID

- 128 bits value
- 8-4-4-12 hex text
  - Printable/Case insensitive
  - e.g. 123e4567-e89b-12d3-a456-426655440000
- 16-bit / 32-bit UUID
  - 0000XXXX-0000-1000-8000-00805F9B34FB

# What's BLE — GAP

- Generic Access Profile (GAP)
  - Define procedure
    - Advertising/Scan/Connect/Security
  - Define roles
    - Broadcaster/Observer
    - Peripheral/Central

# Outline

- What's Bluetooth Low Energy (BLE)
- BLE on Android
- BLE Applications
- Debug tools
- Best practice

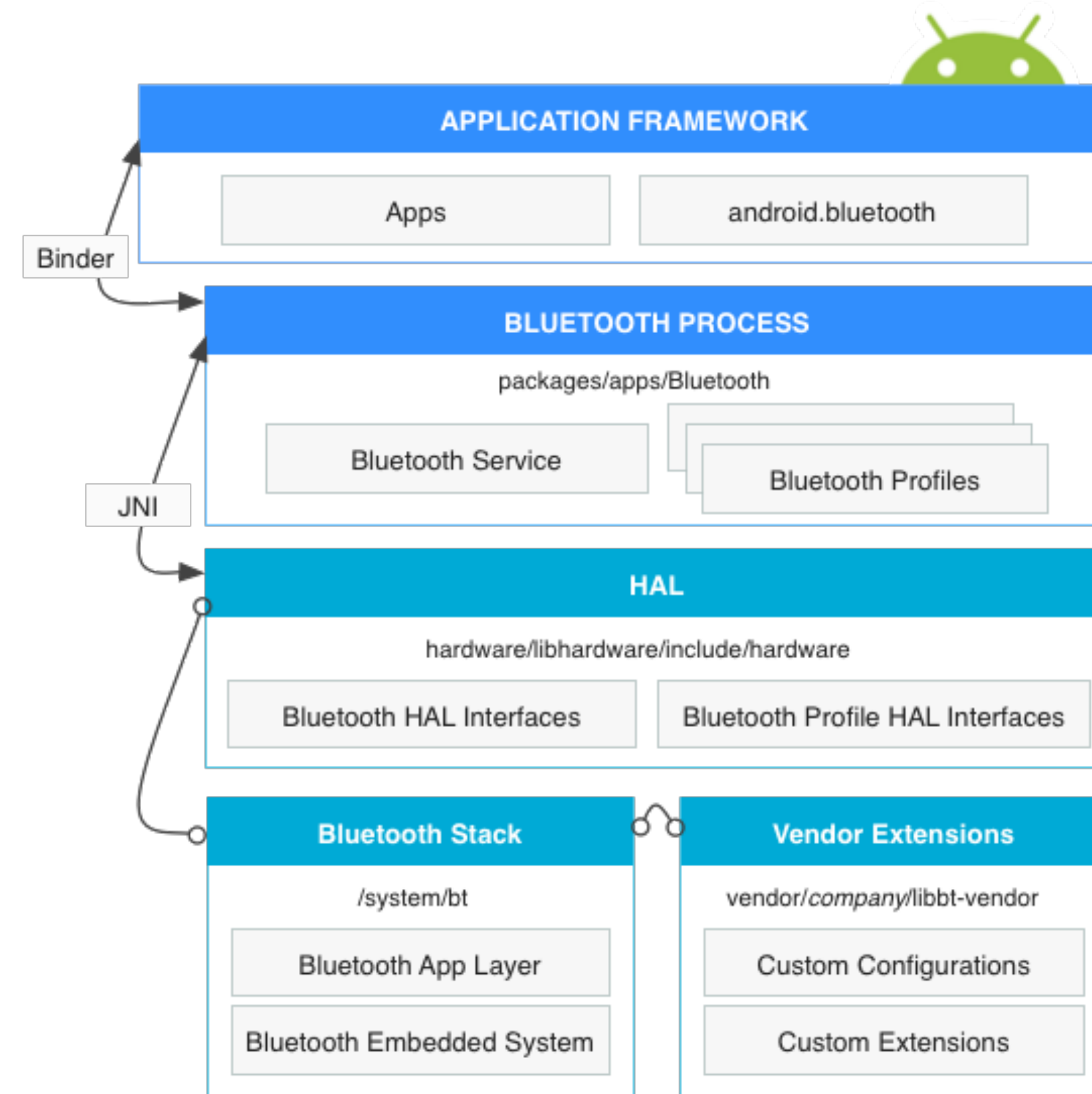
# BLE on Android

- BLE support on Android
  - Central mode — Android 4.3+ (API level  $\geq 18$ )
  - Peripheral mode — Android 5.0+ (API level  $\geq 21$ )
- SDK API — `android.bluetooth.(le)*`
  - `BluetoothGatt/BluetoothGattService/BluetoothGattCharacteristic`
- Permissions

```
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<!-- Android 6.0+ Need location permission -->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" /> <!-- or -->
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

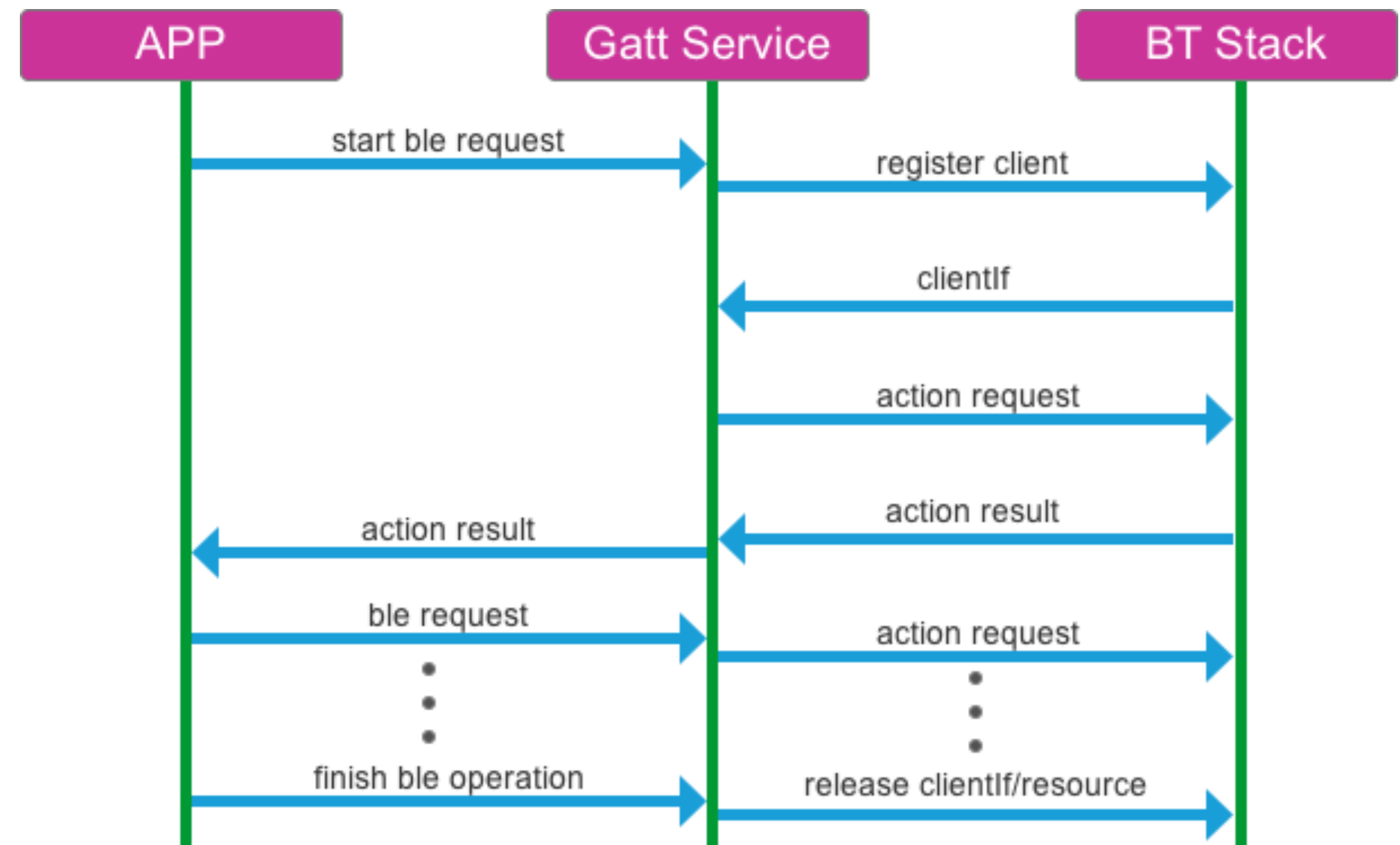
# BLE on Android

- Architecture on Android
  - Hardware  $\longleftrightarrow$  HAL  $\longleftrightarrow$  Android
  - BT process  $\longleftrightarrow$  Binder  $\longleftrightarrow$  Apps
  - BT stack
    - BlueDroid/Vendor
  - Bluetooth process
    - com.android.bluetooth
  - APP



# BLE on Android

- Asynchronous
- Request/Response
- Operation queue
- register/unregister client
- 32 client at most





# Outline

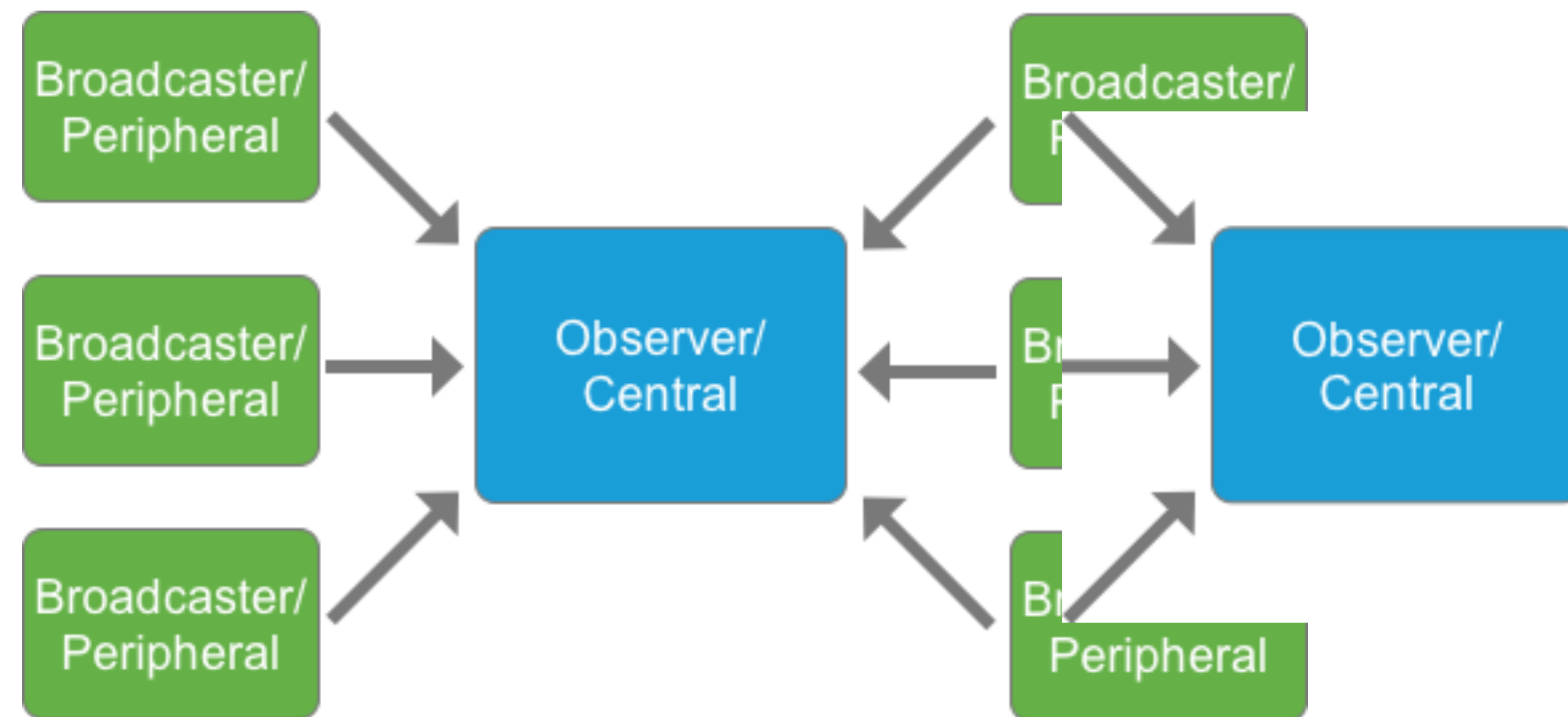
- What's Bluetooth Low Energy (BLE)
- BLE on Android
- BLE Applications
- Debug tools
- Best practice

# BLE applications

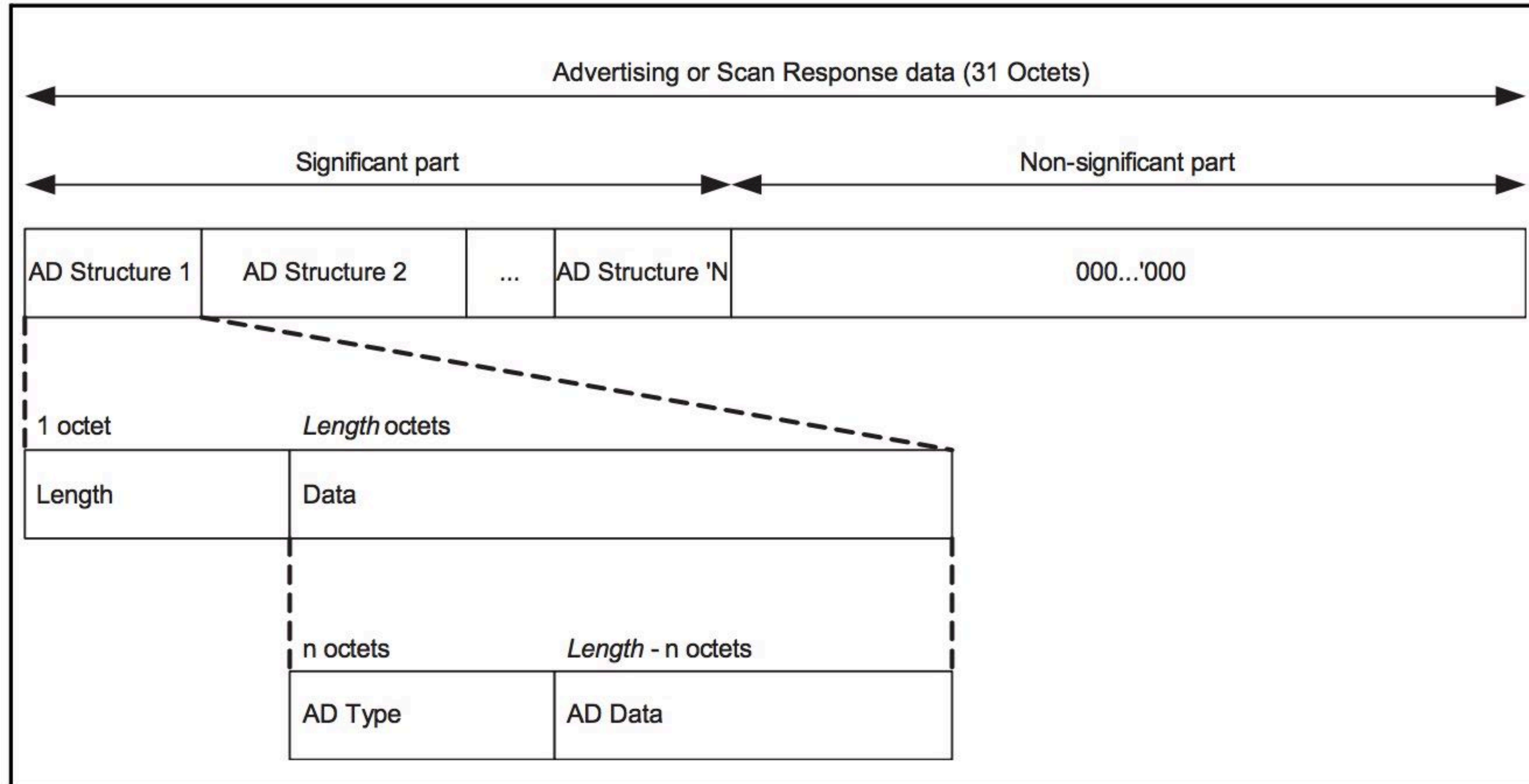
- None-connection/Beacon
  - Observer/Broadcaster
  - Broadcast data/RSSI
- Connection
  - Peripheral/Central
  - Profiles/Services

# BLE App. — Beacon

- Eddystone
- iBeacon
- Mesh — IoT



# Advertising



## Raw data:

```
0x0201041BFF5701007CD96CCE3A50EA  
7D9CA02F63F2F55D7602F59F4D839C61  
0A094D492042616E6420320302E0FE071  
6E0FEB20D0000
```



## Details:

LEN.	TYPE	VALUE
2	0x01	0x04
27	0xFF	0x5701007CD96CCE3A50EA7D9CA 02F63F2F55D7602F59F4D839C61
10	0x09	0x4D492042616E642032
3	0x02	0xE0FE
7	0x16	0xE0FEB20D0000

# BLE app. — Observer

```
// Device scan callback.
private BluetoothAdapter.LeScanCallback mLeScanCallback =
    new BluetoothAdapter.LeScanCallback() {
        @Override
        public void onLeScan(final BluetoothDevice device, int rssi,
            byte[] scanRecord) {
            // Do something with the device
        }
    };

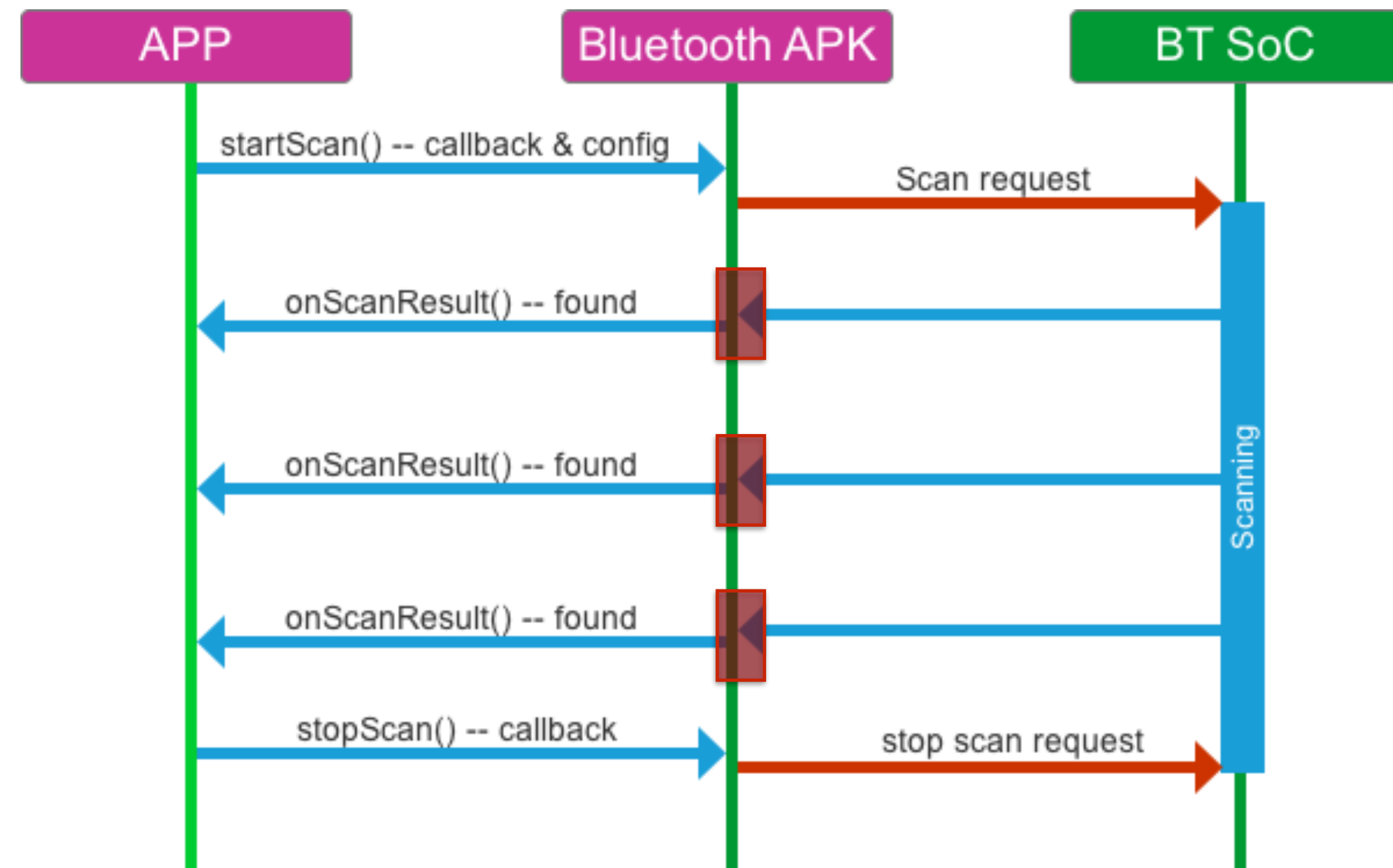
private BluetoothAdapter mBluetoothAdapter = ...;
mBluetoothAdapter.startLeScan(mLeScanCallback);
...
mBluetoothAdapter.stopLeScan(mLeScanCallback);

// Or Classic discover mode
mBluetoothAdapter.startDiscovery()
```



# Scan best practice

- Use new API on Android
- Scan as less as possible
  - Power drain
  - Slowdown connection
- Scan as specific as possible
- Stop scan appropriately
  - ClientIf leak



# BLE App. — Broadcaster

```
// Related API classes
import android.bluetooth.le.AdvertiseCallback;
import android.bluetooth.le.AdvertiseData;
import android.bluetooth.le.AdvertiseSettings;
import android.bluetooth.le.BluetoothLeAdvertiser;

private BluetoothAdapter mBluetoothAdapter = ...;
private AdvertiseCallback mAdvertiseCallback = ...;
private BluetoothLeAdvertiser mLeAdvertiser =
    mBluetoothAdapter.getBluetoothLeAdvertiser();

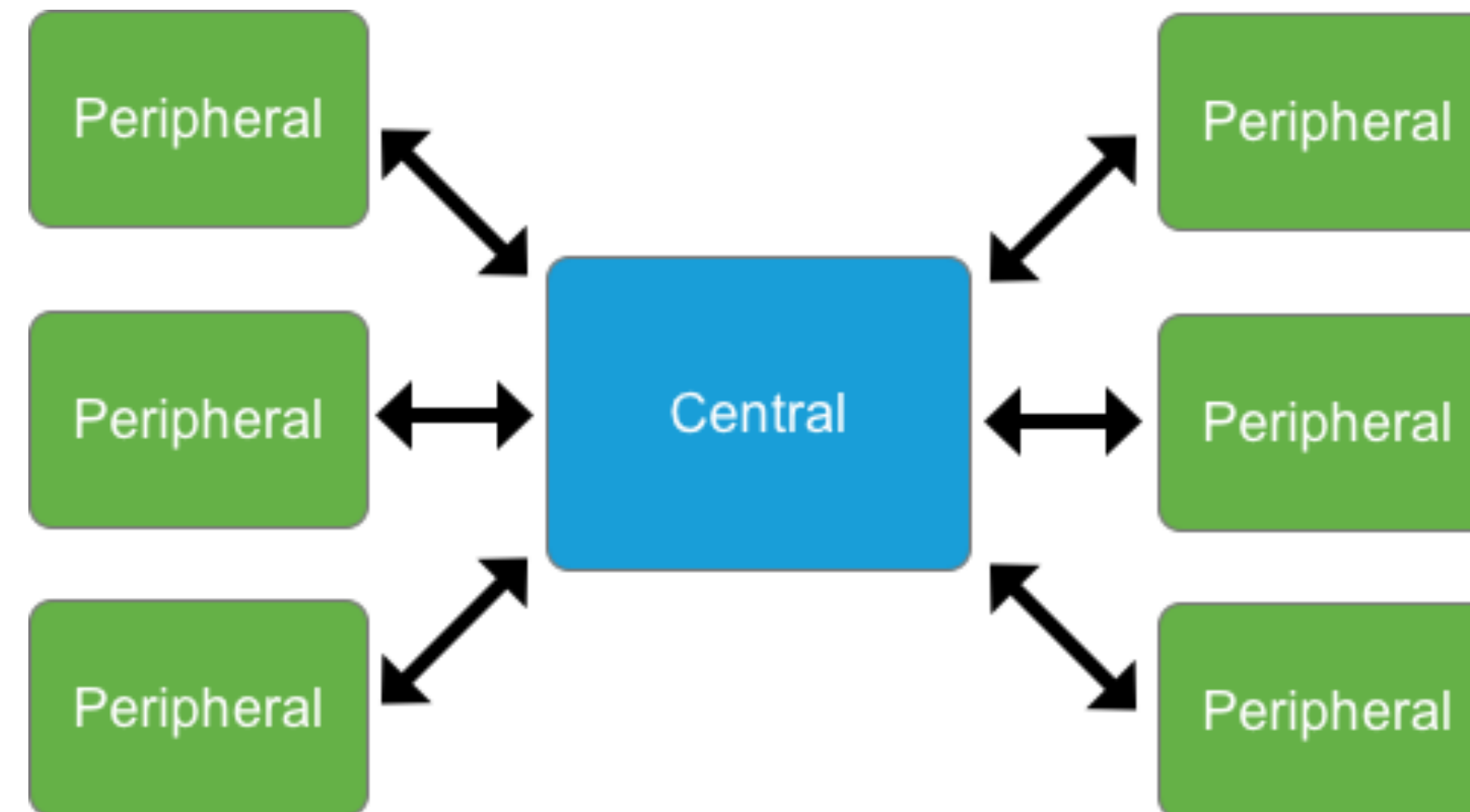
// Advertise configuration
AdvertiseData advData = ...;
AdvertiseData respData = ...;
AdvertiseSettings advSettings = ...;

// start
mLeAdvertiser.startAdvertising(advSettings, advData, respData,
    mAdvertiseCallback);

...
// stop
mLeAdvertiser.stopAdvertising(mAdvertiseCallback);
```

# BLE app. — Connection

- Topology
  - 1 central
  - n peripheral

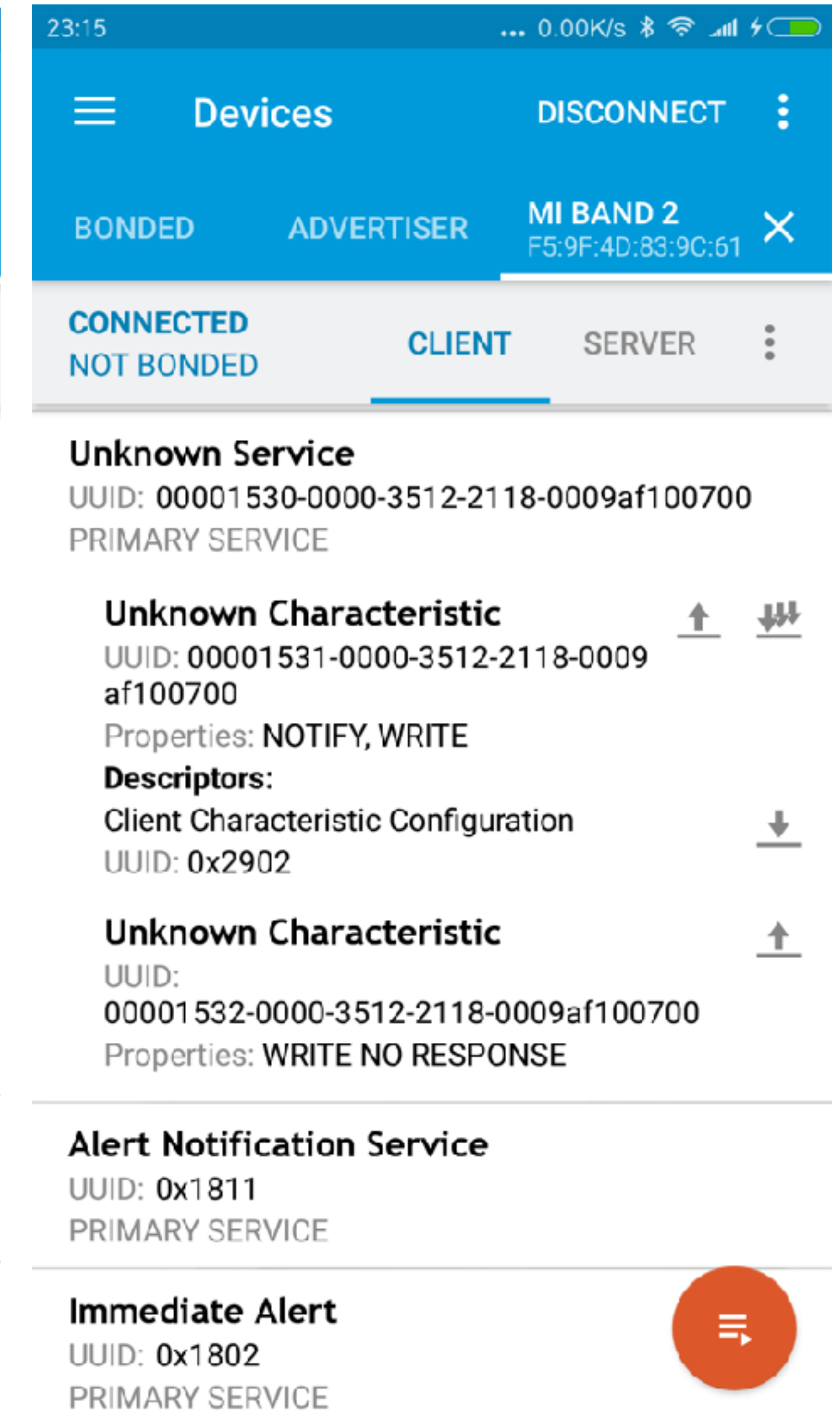
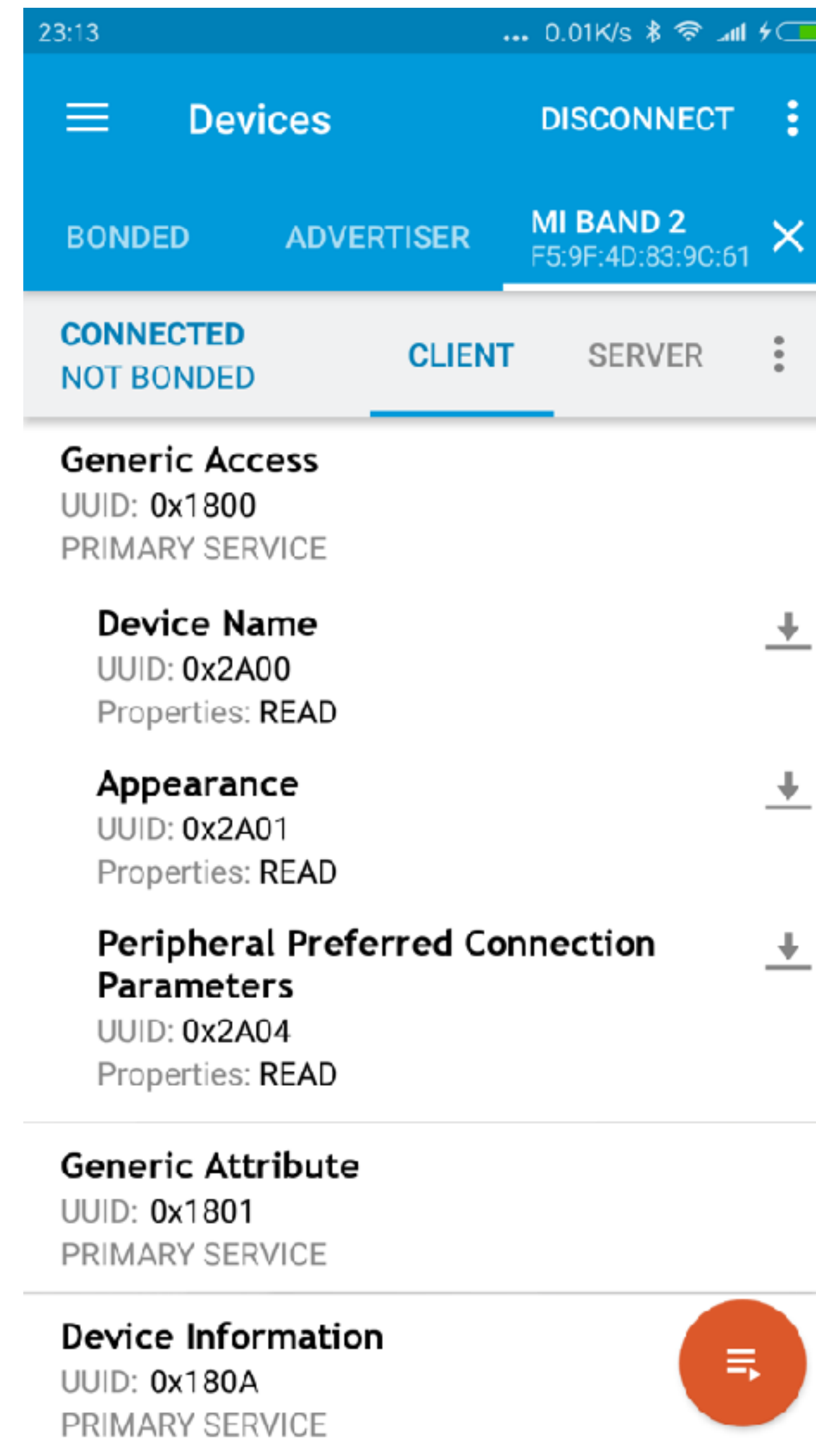


# BLE App. — Connection

- Procedure
  - GAP — Central → Peripheral
  - Connection Parameters
- GATT connection
  - Service/Characteristic/Descriptor
  - Characteristic-W/R/N/I

# GATT connection

- Example
  - Adopted Service
  - Adopted Characteristic
  - Assigned Service/Charact.
  - Custom Service/Charact.



# BLE App. — Central

```
// BLE related class
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothGatt;
import android.bluetooth.BluetoothGattCallback;
import android.bluetooth.BluetoothGattCharacteristic;
import android.bluetooth.BluetoothGattDescriptor;
import android.bluetooth.BluetoothGattService;
import android.bluetooth.BluetoothProfile;

private BluetoothDevice mDevice;
private BluetoothGattCallback mCallback = new BluetoothGattCallback() {
    ...
};

// Connect
private BluetoothGatt mGatt = mDevice.connectGatt(context, false,
    mCallback);

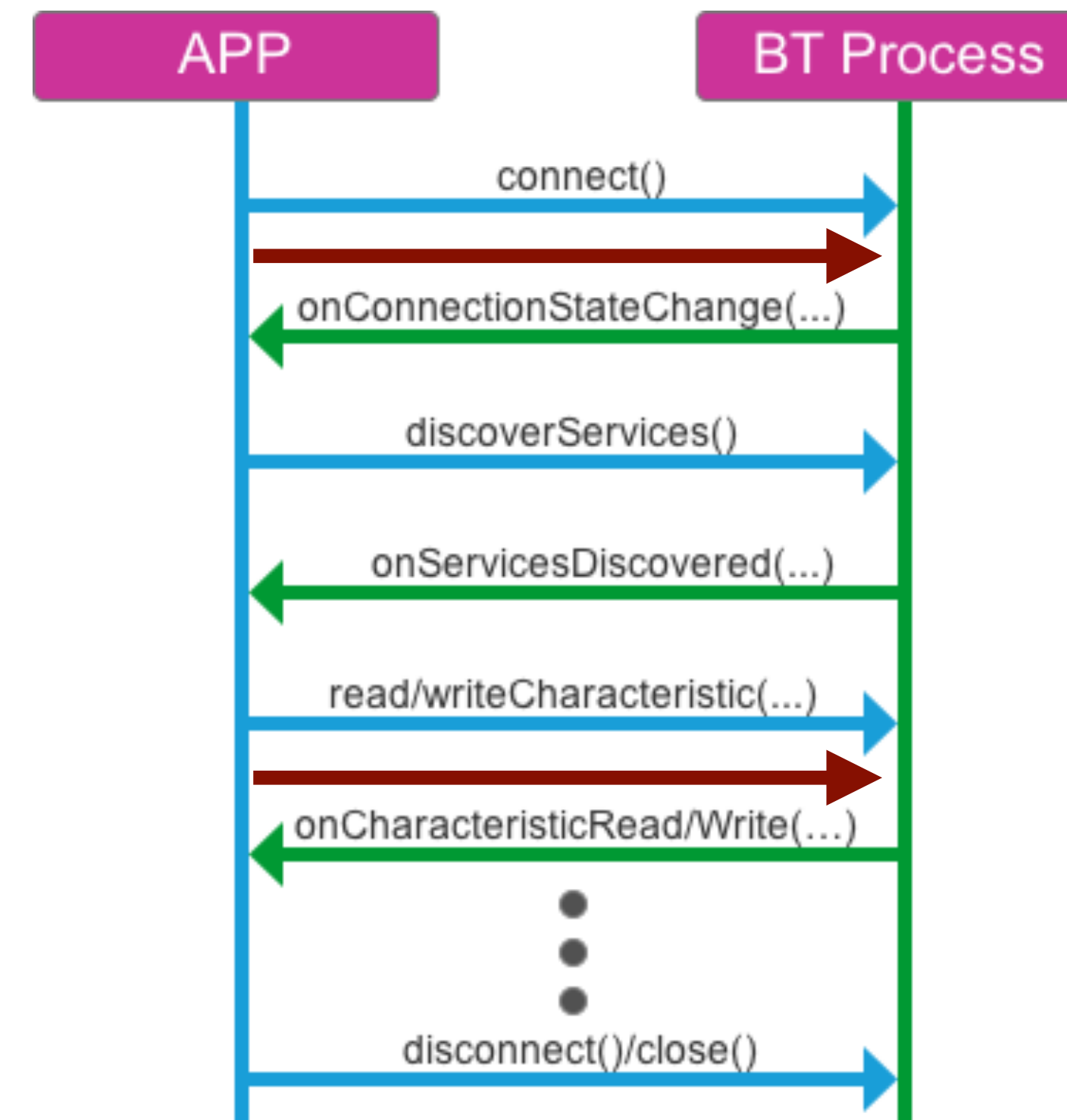
// Disconnect
mGatt.disconnect();

// Close connection
mGatt.close();
```



# GATT Connection

- BLE operation sequence
  - Asynchronous
    - Request/Response
  - In sequence
    - Only 1 request ongoing



# Connection Best Practice

- One connecting request
  - Connection manager
  - Connection request queue
- One R/W request
  - wait/notify
  - Asynchronous → Synchronous

# BLE App. — Peripheral

```
// BLE peripheral related classes
import android.bluetooth.BluetoothGattCharacteristic;
import android.bluetooth.BluetoothGattServer;
import android.bluetooth.BluetoothGattServerCallback;
import android.bluetooth.BluetoothGattService;
import android.bluetooth.BluetoothManager;

private BluetoothManager mBluetoothManager = ...;
private BluetoothAdapter mBluetoothAdapter = ...;
private BluetoothGattServerCallback mGattServerCallback = ...;
private BluetoothGattServer mGattServer =
    mBluetoothManager.openGattServer(mContext, mGattServerCallback);

// Start server
mGattServer = mBluetoothManager.openGattServer(mContext, mGattServerCallback);

// Stop
mGattServer.clearServices();
mGattServer.close();
```

# Outline

- What's Bluetooth Low Energy (BLE)
- BLE on Android
- BLE Applications
- Debug tools
- Best practice



# Debug tools

- BLE Test apps
  - nRF Connect
- Battery usage
  - Battery-historian
- Bluetooth log
  - Logcat
  - BT HCI log
- BT sniffer

The image displays three overlapping screenshots of mobile applications used for debugging Bluetooth Low Energy (BLE) devices.

The top-left screenshot shows the **Battery Historian** app interface. It features a sidebar with a menu icon and a main area displaying a list of battery usage events. The top status bar shows the time as 23:25 and battery level at 0.00K/s.

The top-right screenshot shows a text editor displaying the **btsnoop\_hci.log** file. The log contains a table of Bluetooth HCI events, including a filter set to **btile**. The table columns are No., Time, Source, Destination, Protocol, Length, and Info. The log shows several events, including a frame 1338 and a frame 1339, both of which are 60 bytes long and contain ADV\_IND data.

The bottom screenshot shows a detailed view of a Bluetooth Low Energy (BLE) advertisement packet. The packet is 60 bytes long and contains the following information:

- Access Address: 0x8e89bed6
- Packet Header: 0x2240 (PDU Type: ADV\_IND, TxAdd=false, RxAdd=false)
- Advertising Address: e4:c6:c7:31:95:11 (e4:c6:c7:31:95:11)
- Advertising Data
  - Appearance: Generic Tag (Length: 3, Type: Appearance (0x19), Appearance: Generic Tag (0x0200))
  - Flags (Length: 2, Type: Flags (0x01))
    - 000. .... = Reserved: 0x00
    - ...0 .... = Simultaneous LE and BR/EDR to Same Device Capable (Host): false (0x00)
    - .... 0... = Simultaneous LE and BR/EDR to Same Device Capable (Controller): false (0x00)
    - .... .1.. = BR/EDR Not Supported: true (0x01)
    - .... ..1. = LE General Discoverable Mode: true (0x01)
    - .... ...0 = LE Limited Discoverable Mode: false (0x00)
  - TX Power Level (Length: 2, Type: TX Power Level (0x0a))
    - Power Level (dBm): 0
  - 128-bit Service Class UUIDs (Length: 17, Type: 128-bit service class UUIDs (0x07))
    - Custom UUID: 9ecadc240ee5a9e093f3a3b50100406e
    - CRC: 0xdf2f9f

# Outline

- What's Bluetooth Low Energy (BLE)
- BLE on Android
- BLE Applications
- Debug tools
- Best practice



# Best practices

- Less resources
  - Less connections
  - Less scan
  - Less clientIf
  - Less connected time
- Less advertise
- Less server
- Less parallel request
- Less work in callback

# Best practices

- More testing
  - Various BT vendors
  - Various ROMs
  - Various peripherals
- More ...

# Read More

- Bluetooth core specification v4.1
- <https://www.bluetooth.com/>
- <http://www.race604.com/tag/bluetooth/>
- [http://noahklugman.com/talks/ble\\_presentation.pdf](http://noahklugman.com/talks/ble_presentation.pdf)
- <https://developer.android.com/guide/topics/connectivity/bluetooth-le.html>

# Thanks