BTLE

Bluetooth Low Energy®, a.k.a. Bluetooth Smart®

* simple low energy data transfer
* send simple bits of data fast
* don´t sent alot of data
* easy binding

BTLE

To help consumers identify compatibility and ensure connectivity with products and applications incorporating Bluetooth® Core Specification version 4.0 (or higher), the Bluetooth SIG has developed the Bluetooth Smart and Bluetooth Smart Ready trademarks.[5]

[5] bluetooth.org/en-us/bluetooth-brand/how-to-use-smart-marks

BTLE basics

Services

Services are collections of characteristics and relationships to other services that encapsulate the behavior of part of a device.[2]

[2] developer.bluetooth.org/gatt/services/Pages/ServicesHome.aspx

BTLE basics

Characteristics

Characteristics are defined attribute types that contain a single logical value.[3]

[3] [developer.bluetooth.org/gatt/characteristics/Pages/CharacteristicsHome.aspx)[https://developer.bluetooth.org/gatt/characteristics/Pages/CharacteristicsHome.aspx]

BTLE basics

Descriptors

"Descriptors are defined attributes that describe a characteristic value."[4]

[4] developer.bluetooth.org/gatt/descriptors/Pages/DescriptorsHomePage.aspx

Speed and Cadence

- official Profile all
- Cycling Speed and Cadence
  - org.bluetooth.service.cycling*speed*and_cadence mandatory
  - org.bluetooth.service.device_information optional

It´s all nicely documented.

BTLE on Android

- BluetoothManager
- BluetoothGattCallback
  - onConnectionStateChange
  - onReadRemoteRssi
  - onServicesDiscovered
  - onCharacteristicChanged

Connect.java

BTLE on Android

1. Subscribe to **Notify** Characteristic
   org.bluetooth.characteristic.csc_measurement
2. extract values
3. display them
4. read RSSI (monitor the connection)

Android: Scan and connect

```
final BluetoothAdapter adapter = bluetooth.getAdapter();
UUID[] serviceUUIDs = new UUID[]{CSC_SERVICE_UUID};
adapter.startLeScan(serviceUUIDs, new BluetoothAdapter.LeScanCallback() {
    @Override
    public void onLeScan(BluetoothDevice device, int rssi, byte[] scanRecord) {
        device.connectGatt(Connect.this, autoConnectCheckBox.isChecked(), bluetoothGattCallback);
    }
});
```

Android: Scan for Services

BluetoothGattCallback:

```
@Override
public void onConnectionStateChange(BluetoothGatt gatt, int status, int state) {
    super.onConnectionStateChange(gatt, status, state);
    switch (state) {
        case BluetoothGatt.STATE_CONNECTED: {
            showText("STATE_CONNECTED", Style.INFO);
            setConnectedGatt(gatt);
            gatt.discoverServices();
            break;
        }
    }
}
```

Android: Register for Updates

Register for Updates of the org.bluetooth.characteristic.csc_measurement

BluetoothGattCallback:

```
@Override
public void onServicesDiscovered(BluetoothGatt gatt, int status) {
    super.onServicesDiscovered(gatt, status);
    BluetoothGattCharacteristic valueCharacteristic = gatt.getService(CSC_SERVICE_UUID).getCharacteristic(CSC_CHARACTERISTIC_UUID);
    boolean notificationSet = gatt.setCharacteristicNotification(valueCharacteristic, true);
    BluetoothGattDescriptor descriptor = valueCharacteristic.getDescriptor(BTLE_NOTIFICATION_DESCRIPTOR_UUID);
    descriptor.setValue(BluetoothGattDescriptor.ENABLE_NOTIFICATION_VALUE);
    boolean writeDescriptorSuccess = gatt.writeDescriptor(descriptor);
}
```

Android: Monitor the RSSI

BluetoothGattCallback:

```
@Override
public void onReadRemoteRssi(BluetoothGatt gatt, int rssi, int status) {
    super.onReadRemoteRssi(gatt, rssi, status);
    listener.updateRssiDisplay(rssi);
}
@Override
public void onCharacteristicChanged(BluetoothGatt gatt, BluetoothGattCharacteristic characteristic) {
    super.onCharacteristicChanged(gatt, characteristic);
    gatt.readRemoteRssi();
}
```

Also when scanning:

```
final BluetoothAdapter adapter = bluetooth.getAdapter();
UUID[] serviceUUIDs = new UUID[]{CSC_SERVICE_UUID};
adapter.startLeScan(serviceUUIDs, new BluetoothAdapter.LeScanCallback() {
    @Override
    public void onLeScan(BluetoothDevice device, int rssi, byte[] scanRecord) {
        listener.updateRssiDisplay(rssi);
    }
});
```


Main Activity
Finish Activity
Save Battery
heart beat: 12
bpm (-61db)

Android Read/Write

Async interface:

* one BluetoothGattCallback per device
 * async execution of commands with callbacks

```
onCharacteristicRead(BluetoothGatt gatt,
BluetoothGattCharacteristic characteristic, int
status)
```

Android Read and Write, read/write multiple values

- read, write, read tricky since the `BluetoothGattCharacteristic` contains the value
  - Helper class needed
  - `BluetoothGattCommand` and `BluetoothGattCommandQueue`

Android Read and Write, read/ write multiple values

still work in progress[9]

```
final BluetoothAdapter adapter = bluetooth.getAdapter();
UUID[] serviceUUIDs = new UUID[]{CSC_SERVICE_UUID};

final GattCommandServiceGroup service = new GattCommandServiceGroup(BTUUID.Service.device_information);
service.addCharacteristicOperation(GattCommand.CommandOperation.OPERATION_READ, BTUUID.Characteristic.manufacturer_name_string);
service.addCharacteristicOperation(GattCommand.CommandOperation.OPERATION_READ, BTUUID.Characteristic.model_number_string);
service.addCharacteristicOperation(GattCommand.CommandOperation.OPERATION_READ, BTUUID.Characteristic.firmware_revision_string);
service.addCharacteristicOperation(GattCommand.CommandOperation.OPERATION_READ, BTUUID.Characteristic.hardware_revision_string);
service.addCharacteristicOperation(GattCommand.CommandOperation.OPERATION_READ, BTUUID.Characteristic.serial_number_string);
final GattCommandQueue queue = new GattCommandQueue();
queue.add(service);
queue.setGattCommandQueueCallback(<your callback>)
queue.executeWhenConnected();
[...]
```

[9] Connect.java#L264

Android Wearables

Of course all this works directly on Android Wear

- minimal sample included.
- same code
- no host app needed
- wear app => main app, phone app not needed

Android

Meet the AndroidSimpleBikeComputer

```
git clone https://github.com/deadfalkon/android-simple-bike-computer.git
git clone https://github.com/deadfalkon/simple-bike-computer-presentation.git
```