PHYSICAL COMPUTING 4: ANDROID + ARDUINO BLE

CSE 590 Ubiquitous Computing | Lecture 6 | May 3

Jon Froehlich • **Liang He (TA)**

DESIGN USE BUILD

MAKEABILITY LAB

PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

UNIVERSITY of WASHINGTON

SCHEDULE TODAY: 6:30-9:20

06:30-06:55: Discussion of required reading led by Joe Wandyez

06:55-07:00: Optional discussion led by Abhay Rijhwani (link)

07:00-08:10: Physical Computing 3: Sensors

08:10-08:15: Break

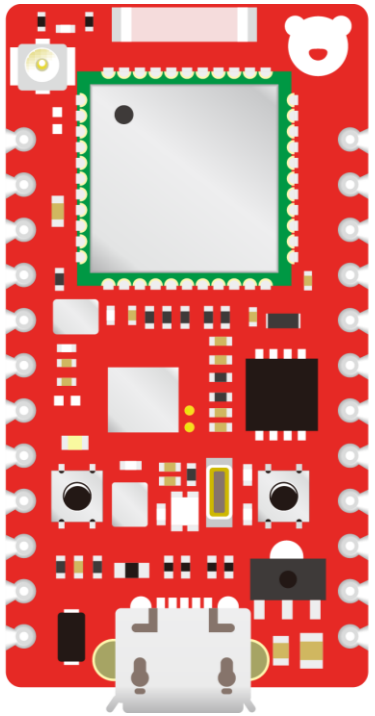
08:15-09:20: Physical Computing 4: Android + Arduino via BLE

ARDUINO + ANDROID BLE: LEARNING GOALS

How to setup **basic unidirectional communication** between Android -> Arduino using BLE

How to setup basic **bidirectional communication** using BLE

ARDUINO + ANDROID BLE: OVERVIEW



1 BLE initialization
Configuration
Set advertising data (name)

2 Register callbacks (receive & send) and services (UUIDs)
Start advertising

3 Receive data from Android and process the data

4 Send data to Android

RedBear Duo (Peripheral)

1 Scan all BLE-enabled devices

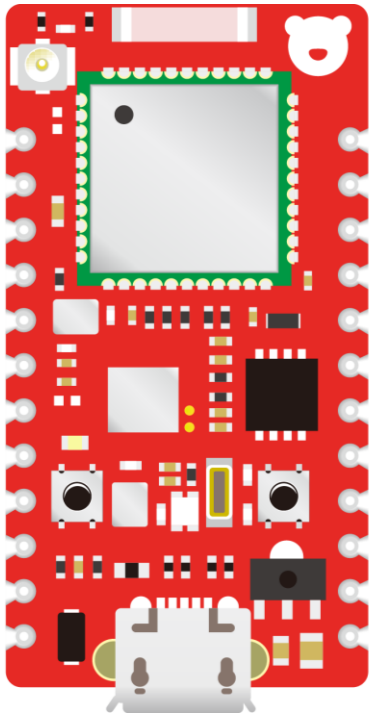
2 Find target device by **UUID** and **NAME** and establish the connection

3 Send data to Duo

4 Receive data from Duo and process the data

Android App (Central)

ARDUINO + ANDROID BLE: OVERVIEW (CODE LEVEL)



ble.init();
configureBLE();
ble.setAdvertisementData()

1 BLE initialization
Configuration
Set advertising data (name)

2 Register callbacks (receive & send) and services (UUIDs)
Start advertising

ble.onDataWriteCallback()
ble.addService
ble.startAdvertising

3 Receive data from Android and process the data

bleWriteCallback()

4 Send data to Android

send_notify()

RedBear Duo (Peripheral)

scanLeDevice()

1 Scan all BLE-enabled devices

BluetoothAdapter.LeScanCallback()

2 Find target device by **UUID** and **NAME** and establish the connection

3 Send data to Duo

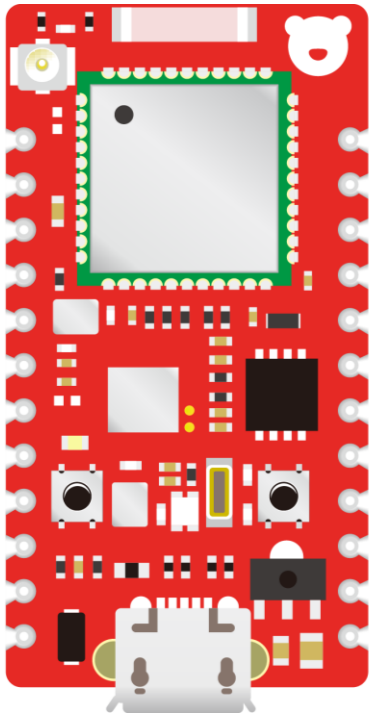
mBluetoothLeService.writeCharacteristic()

4 Receive data from Duo and process the data

intent.getByteArrayExtra()
Customized functions (e.g., readAnalogInValue()

Android App (Central)

ARDUINO + ANDROID BLE: OVERVIEW (CODE LEVEL)



RedBear Duo (Peripheral)

UUID (universally Unique identifier)
Device Name
The Length of the Name
Message (3 bytes)

TX UUID & callback (bleWriteCallback)

RX UUID & callback (send_notify)

Android App (Central)

ANDROID/ARDUINO BLE COMMUNICATION

SKELETON CODE

AndroidBLEBasic (one-way) and RedBearDuoBLEAdvanced (two-way)

Open the notes doc: https://goo.gl/3A9Q9M

DUO SKETCH

```
#define BLE_SHORT_NAME_LEN 0x08 // must be in the range of [0x01, 0x09]
#define BLE_SHORT_NAME 'B','L','E','D','e','m','o'  // define each char but the number of char should be BLE_SHORT_NAME_LEN-1
/* Define the pins on the Duo board
 * TODO: change the pins here for your applications
 */
#define PWM_PIN                 D0
#define ANALOG_IN_PIN           A0

// UUID is used to find the device by other BLE-abled devices
static uint8_t service1_uuid[16]    = { 0x71,0x3d,0x00,0x00,0x50,0x3e,0x4c,0x75,0xba,0x94,0x31,0x48,0xf1,0x8d,0x94,0x1e };
static uint8_t service1_tx_uuid[16] = { 0x71,0x3d,0x00,0x03,0x50,0x3e,0x4c,0x75,0xba,0x94,0x31,0x48,0xf1,0x8d,0x94,0x1e };
static uint8_t service1_rx_uuid[16] = { 0x71,0x3d,0x00,0x02,0x50,0x3e,0x4c,0x75,0xba,0x94,0x31,0x48,0xf1,0x8d,0x94,0x1e };

int bleWriteCallback(uint16_t value_handle, uint8_t *buffer, uint16_t size)
static void  send_notify(btstack_timer_source_t *ts)
```

Android App –> Duo

Duo -> Android App

ANDROID/ARDUINO BLE COMMUNICATION

SKELETON CODE

AndroidBLEBasic (one-way) and RedBearDuoBLEAdvanced (two-way)

Open the notes doc: https://goo.gl/3A9Q9M

ANDROID PROGRAM

```
// Define the device name and the length of the name
// Note the device name and the length should be consistent with the ones defined in the Duo sketch
private String mTargetDeviceName = "BLEDemo";
private int mNameLen = 0x08;
```

```
@Override
public void onProgressChanged(SeekBar seekBar, int progress,
                              boolean fromUser) {
    byte[] buf = new byte[] { (byte) 0x02, (byte) 0x00, (byte) 0x00 };

    buf[1] = (byte) mPWMSeekBar.getProgress();

    mCharacteristicTx.setValue(buf);
    mBluetoothLeService.writeCharacteristic(mCharacteristicTx);
}
```

```
// Process the Gatt and get data if there is data coming from Duo board. Created by the RedBear Team
private final BroadcastReceiver mGattUpdateReceiver = (context, intent) -> {
        final String action = intent.getAction();

        if (RBLService.ACTION_GATT_DISCONNECTED.equals(action)) {
            Toast.makeText(getApplicationContext(), text: "Disconnected",
                    Toast.LENGTH_SHORT).show();
            setButtonDisable();
        } else if (RBLService.ACTION_GATT_SERVICES_DISCOVERED
                .equals(action)) {
            Toast.makeText(getApplicationContext(), text: "Connected",
                    Toast.LENGTH_SHORT).show();

            getGattService(mBluetoothLeService.getSupportedGattService());
        } else if (RBLService.ACTION_DATA_AVAILABLE.equals(action)) {
            mData = intent.getByteArrayExtra(RBLService.EXTRA_DATA);

            readAnalogInValue(mData);
        } else if (RBLService.ACTION_GATT_RSSI.equals(action)) {
            displayData(intent.getStringExtra(RBLService.EXTRA_DATA));
        }
};
```

EXERCISE 1: TURN ON AN LED VIA ANDROID APP

For this exercise, you will use AndroidBLEBasic and RedBearDuoBLEBasic

Open both code sets (in Android Studio and Arduino IDE respectively)

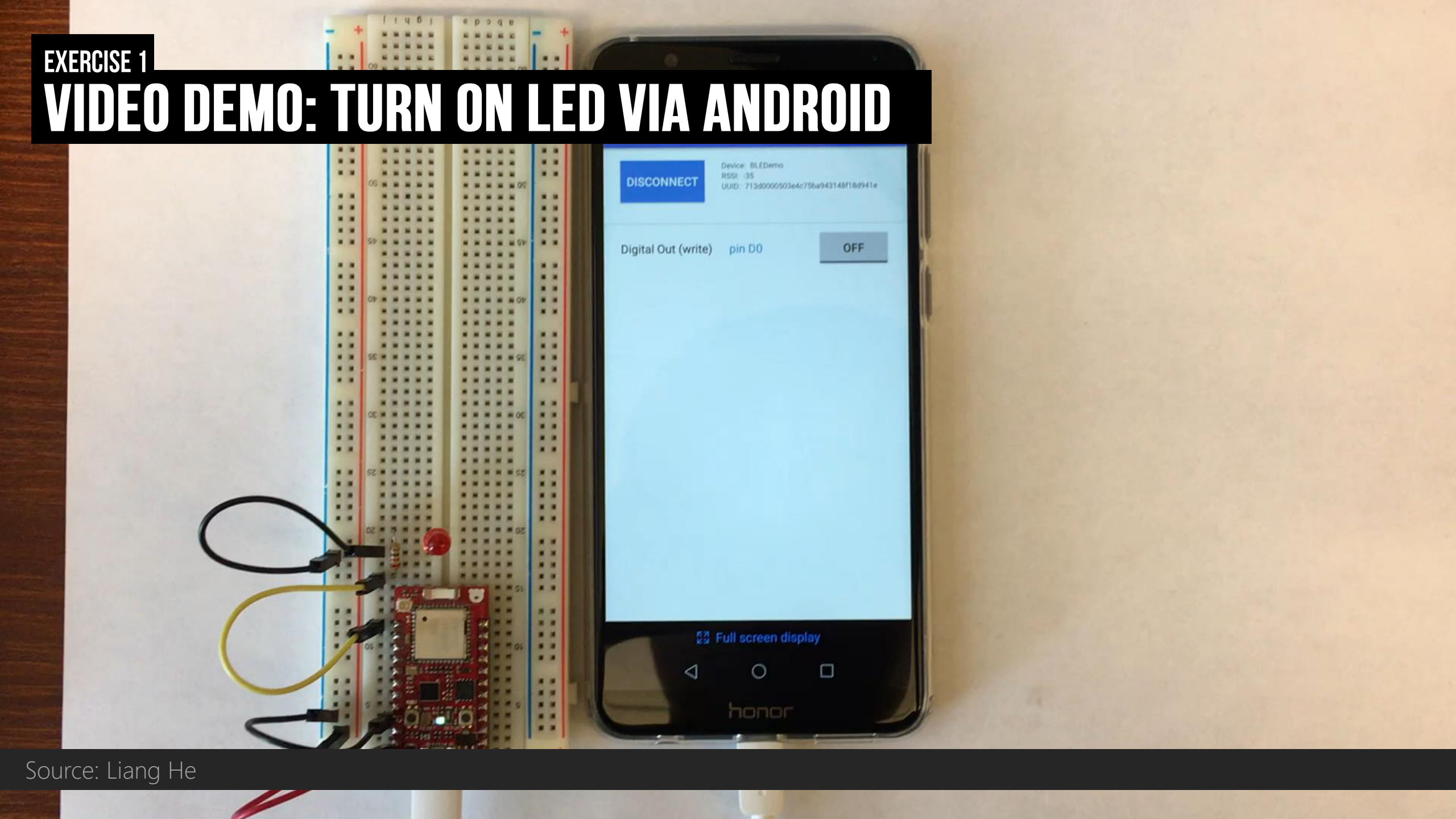
You have to change the device name

Change the pin number to the pin that you want to use

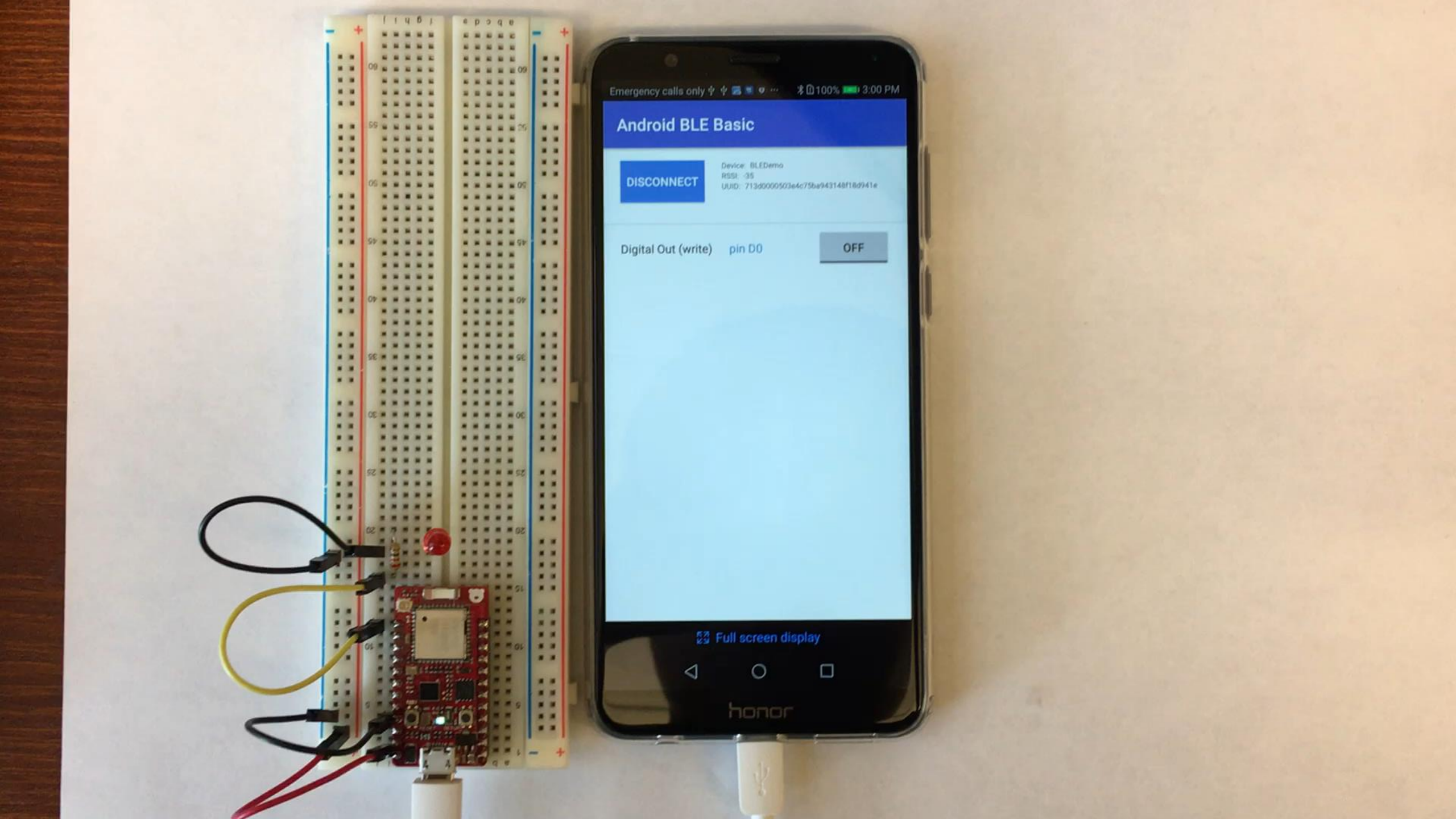
Experiment and play!

EXERCISE 1

VIDEO DEMO: TURN ON LED VIA ANDROID



Source: Liang He


Emergency calls only
100%
3:00 PM
Android BLE Basic
DISCONNECT
Device: BLEDemo
RSSI: -35
UUID: 713d0000503e4c75ba943148f18d941e
Digital Out (write)
pin D0
OFF
Full screen display
honor

EXERCISE 2: FADE AN LED VIA ANDROID APP

Again, use AndroidBLEBasic and RedBearDuoBLEBasic

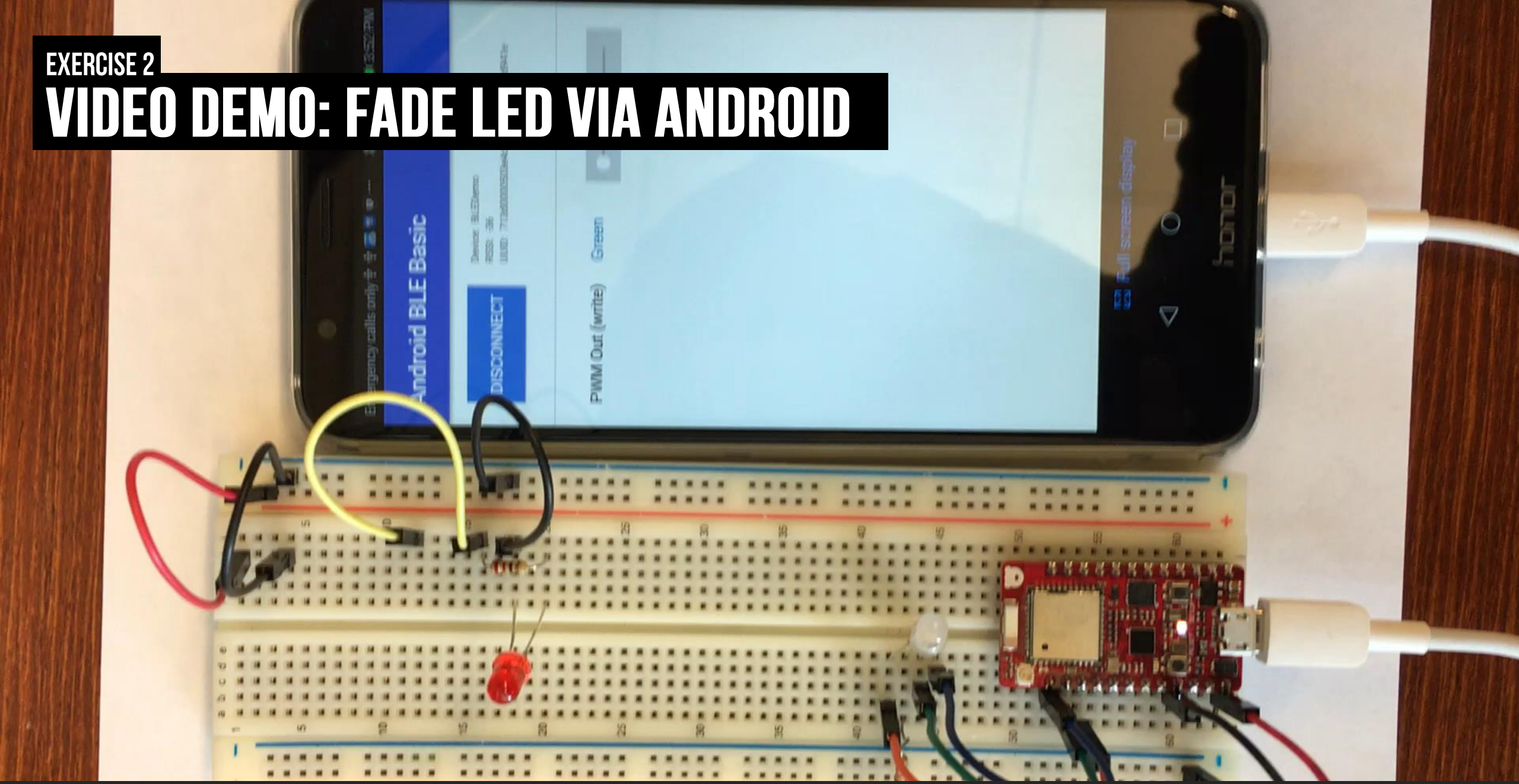
For this, make sure your circuit has the LED hooked up to a PWM pin

Modify Arduino code to use analogWrite (refer to the RedBearDuoBLEAdvanced)

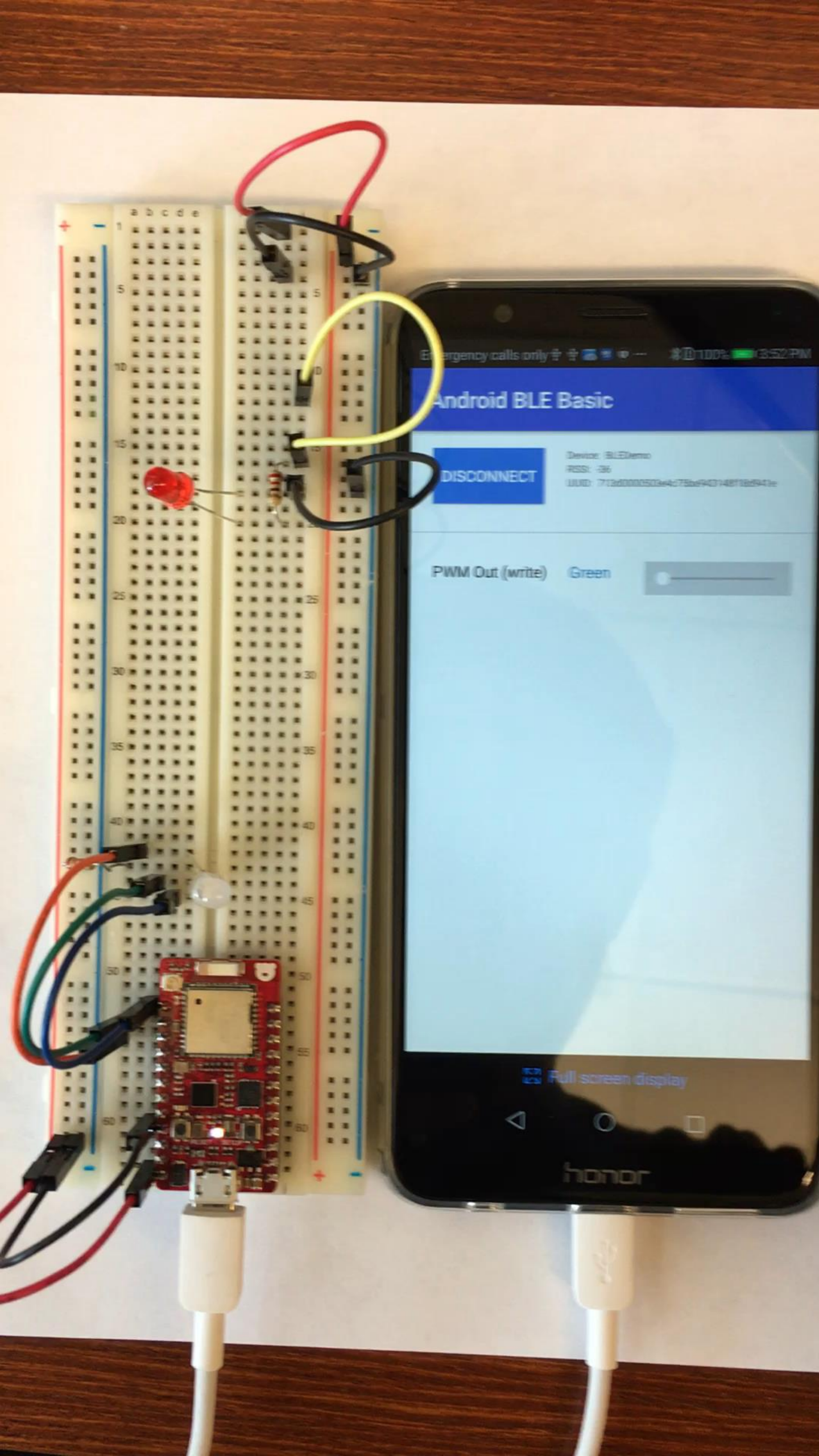
Update Android app to use a SeekBar or some other continuous widget

EXERCISE 2

VIDEO DEMO: FADE LED VIA ANDROID



Source: Liang He


Android BLE Basic
DISCONNECT
PWM Out (write)
Green
Full screen display
honor

EXERCISE 3: 2-WAY COM BETWEEN ANDROID & ARDUINO

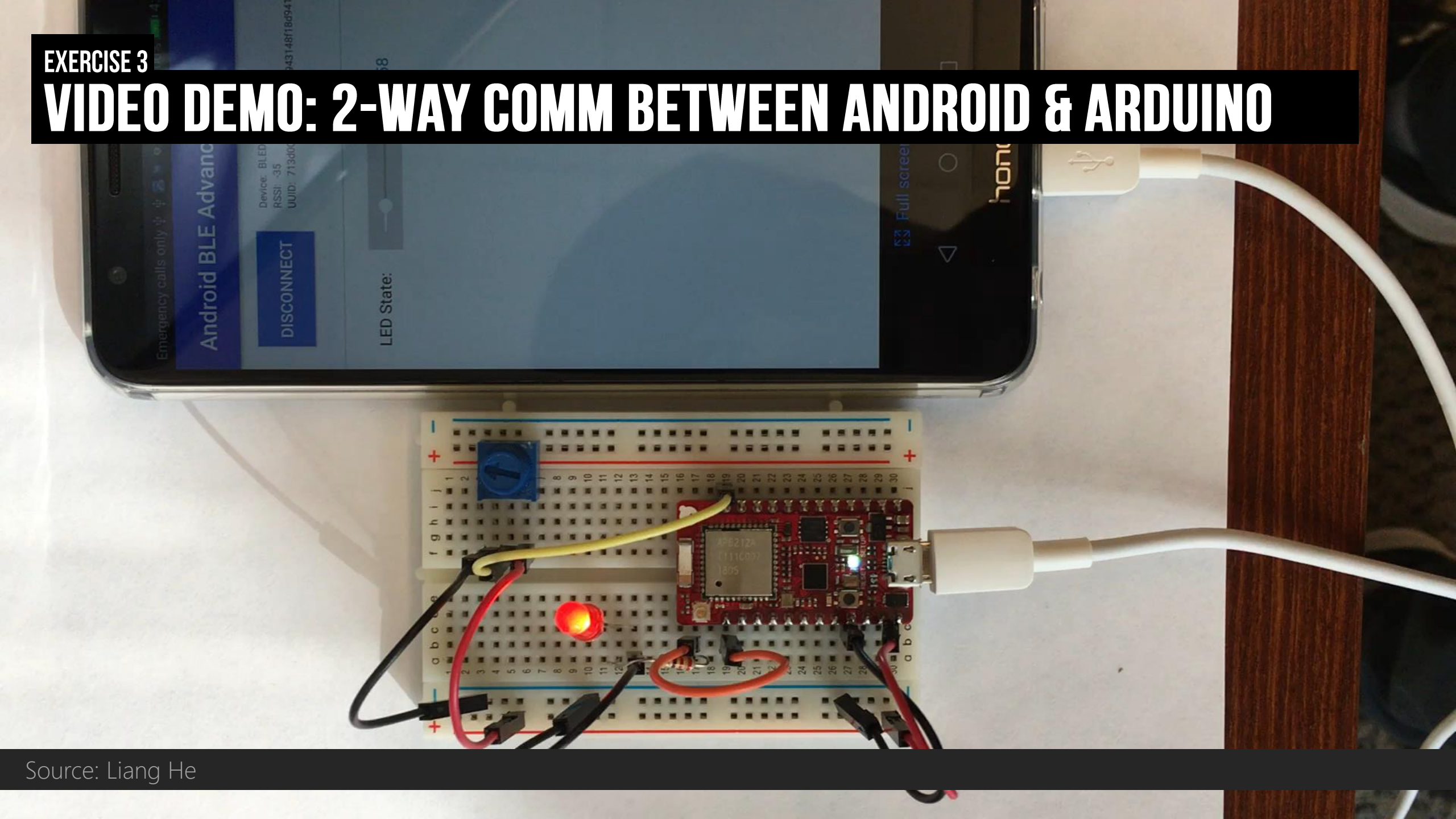
For this, start with AndroidBLEAdvanced and RedBearDuoBLEAdvanced

Fade the LED from Android

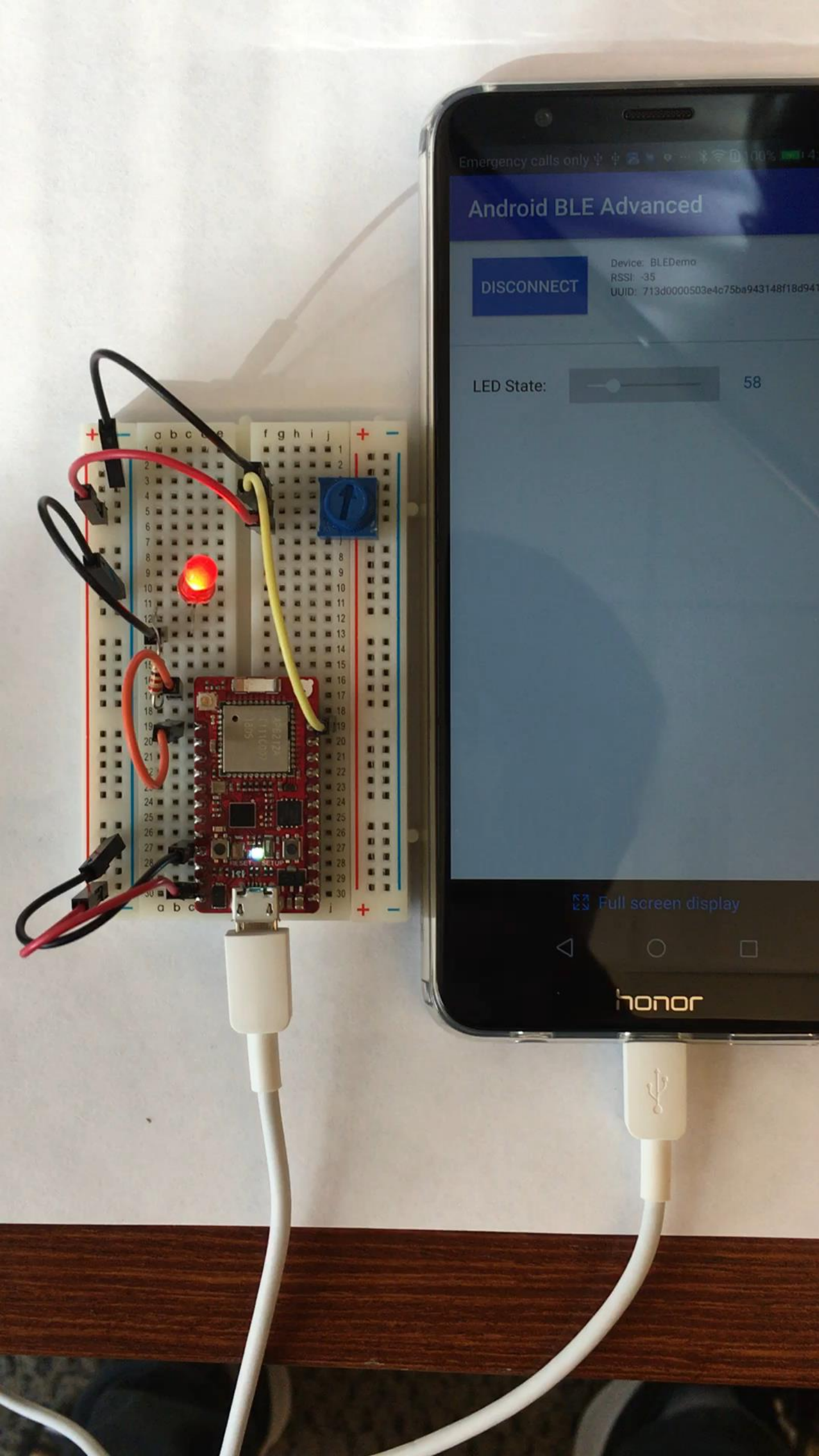
Fade the LED from Arduino (and update Android to properly reflect state)

EXERCISE 3

VIDEO DEMO: 2-WAY COMM BETWEEN ANDROID & ARDUINO



Source: Liang He


Emergency calls only
Android BLE Advanced
DISCONNECT
Device: BLEDemo
RSSI: -35
UUID: 713d0000503e4c75ba943148f18d941
LED State:
58
Full screen display
honor