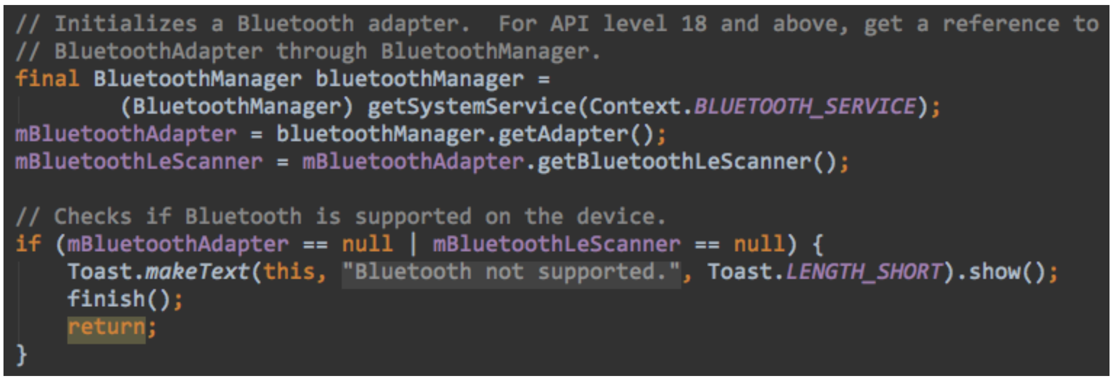
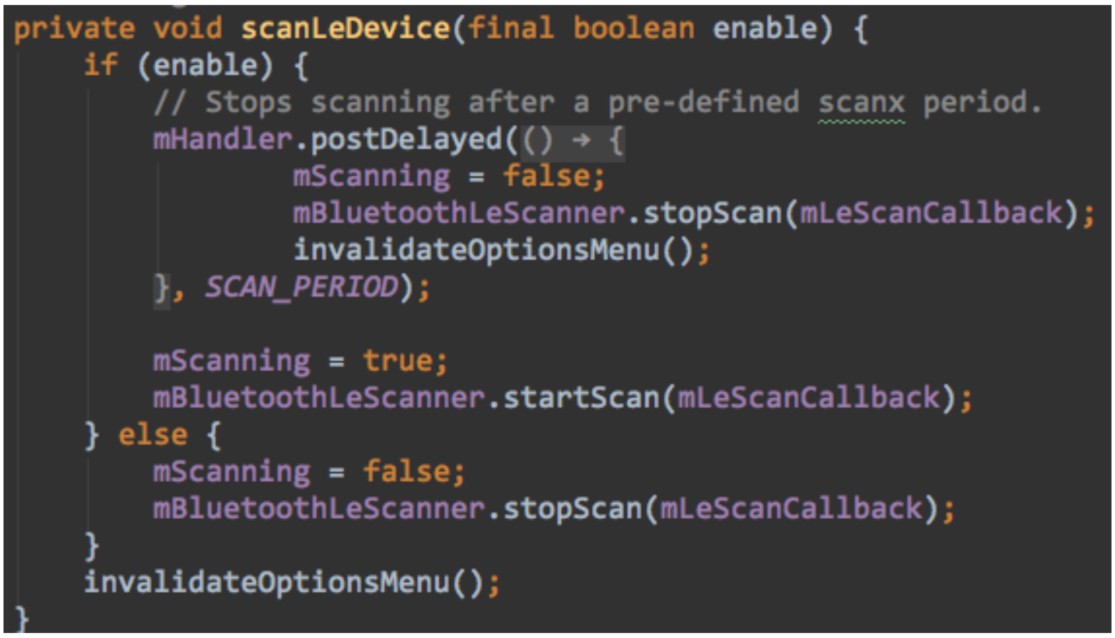
開啟藍芽

DeviceScanAcitivity onCreate();



```java
// Initializes a Bluetooth adapter.  For API level 18 and above, get a reference to
// BluetoothAdapter through BluetoothManager.
final BluetoothManager bluetoothManager =
        (BluetoothManager) getSystemService(Context.BLUETOOTH_SERVICE);
mBluetoothAdapter = bluetoothManager.getAdapter();
mBluetoothLeScanner = mBluetoothAdapter.getBluetoothLeScanner();

// Checks if Bluetooth is supported on the device.
if (mBluetoothAdapter == null | mBluetoothLeScanner == null) {
    Toast.makeText(this, "Bluetooth not supported.", Toast.LENGTH_SHORT).show();
    finish();
    return;
}
```

掃描裝置（只支援 4.0 裝置）

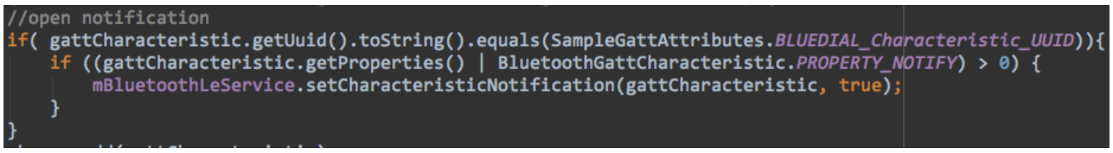


```java
private void scanLeDevice(final boolean enable) {
    if (enable) {
        // Stops scanning after a pre-defined scanx period.
        mHandler.postDelayed(() → {
                mScanning = false;
                mBluetoothLeScanner.stopScan(mLeScanCallback);
                invalidateOptionsMenu();
        }, SCAN_PERIOD);

        mScanning = true;
        mBluetoothLeScanner.startScan(mLeScanCallback);
    } else {
        mScanning = false;
        mBluetoothLeScanner.stopScan(mLeScanCallback);
    }
    invalidateOptionsMenu();
}
```

選擇裝置後會進入 DeviceControlActivity

1. 開啟資料更新通知

   DeviceControlActivity: displayGattServices()

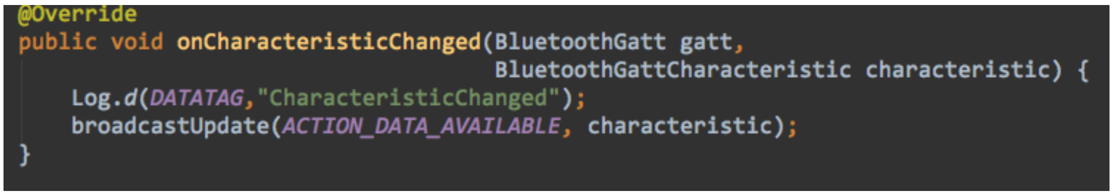
```java
//open notification
if( gattCharacteristic.getUuid().toString().equals(SampleGattAttributes.BLUEDIAL_Characteristic_UUID)){
    if ((gattCharacteristic.getProperties() | BluetoothGattCharacteristic.PROPERTY_NOTIFY) > 0) {
        mBluetoothLeService.setCharacteristicNotification(gattCharacteristic, true);
    }
}
```

   如果 UUID 與資料流 UUID 吻合，打開資料更新服務

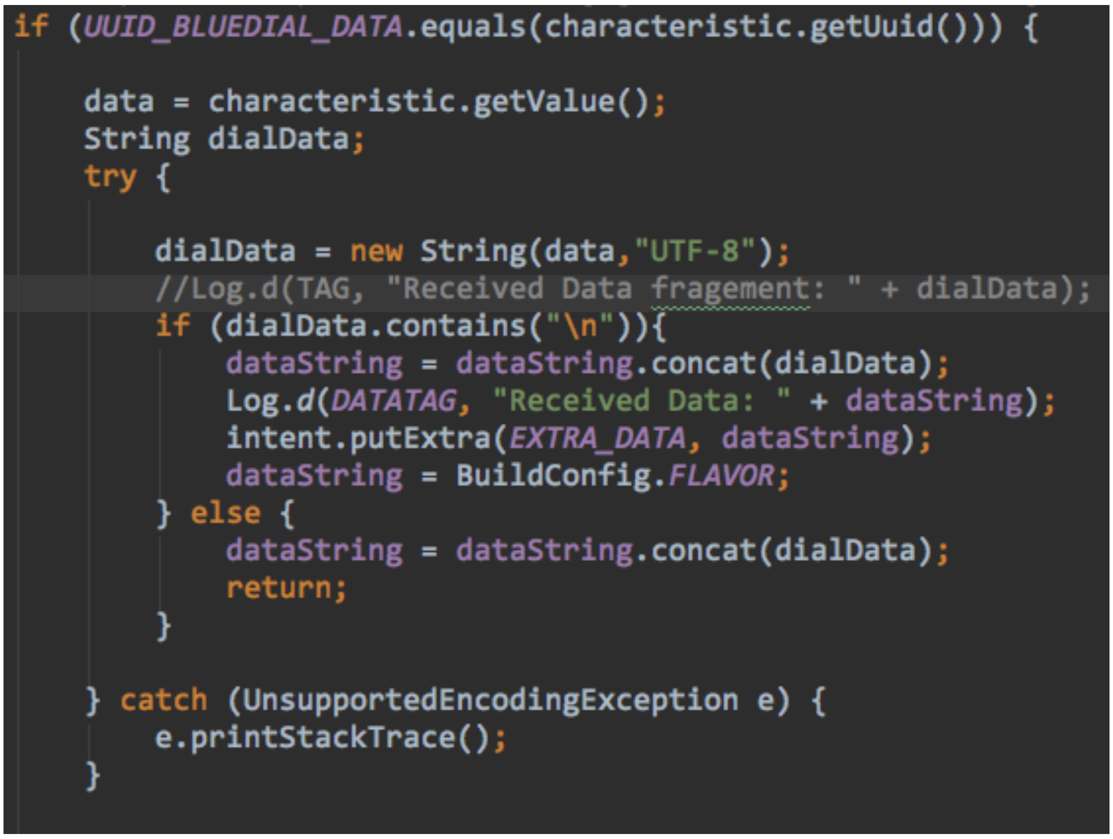
   呼叫 BluetoothLeService:setCharacteristicNotification() //實作開啟通知

2. 等待資料進入後即可開始上傳

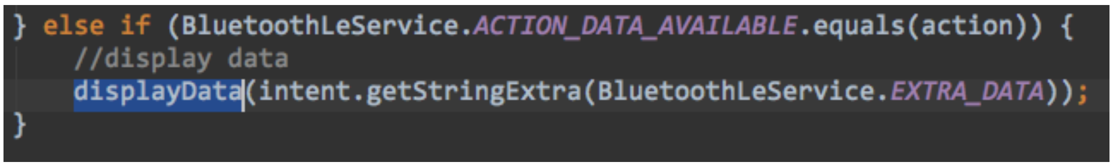
   1) 資料更新會傳回 BluetoothLeService 的 BluetoothGattCallback，CallBack 實作資料更新通知，並呼叫 broadcastUpdate


```
@Override
public void onCharacteristicChanged(BluetoothGatt gatt,
                                    BluetoothGattCharacteristic characteristic) {
    Log.d(DATATAG,"CharacteristicChanged");
    broadcastUpdate(ACTION_DATA_AVAILABLE, characteristic);
}
```

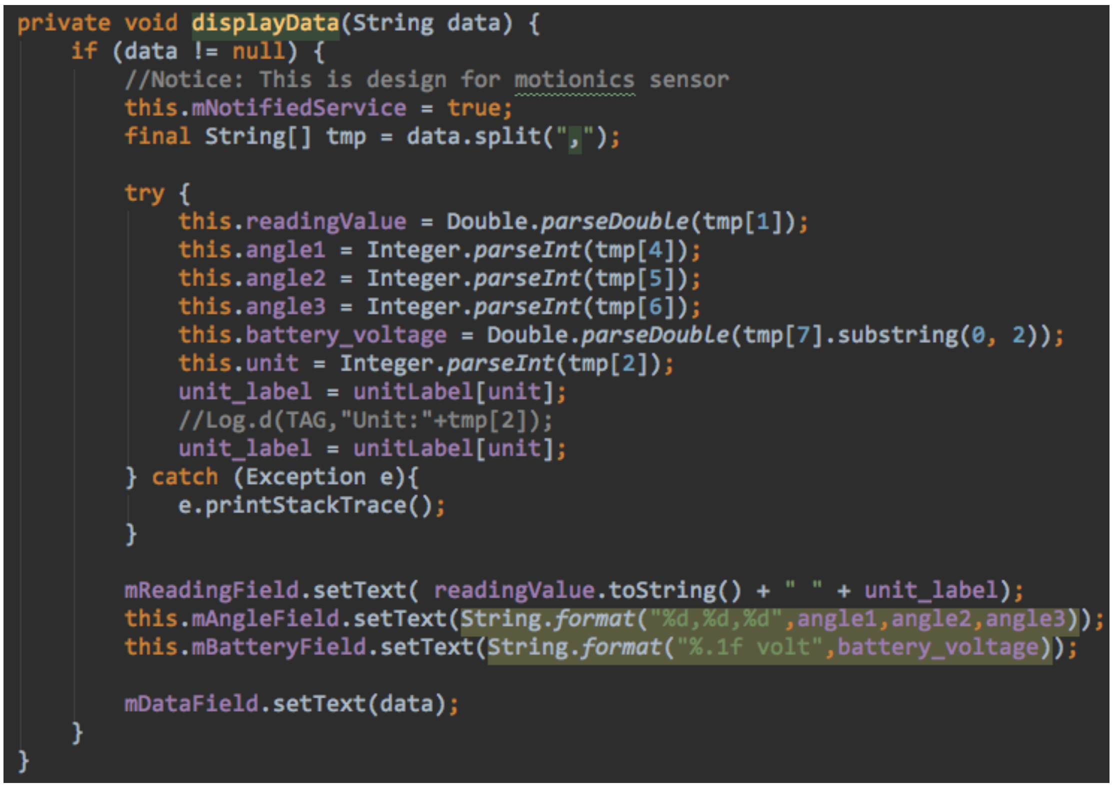
2) broadcastUpdate 中實作合併資料，並由 broadcast 的方式傳回 DeviceControlAcitivity 顯示資料。


```
if (UUID_BLUEDIAL_DATA.equals(characteristic.getUuid())) {

    data = characteristic.getValue();
    String dialData;
    try {

        dialData = new String(data,"UTF-8");
        //Log.d(TAG, "Received Data fragement: " + dialData);
        if (dialData.contains("\n")){
            dataString = dataString.concat(dialData);
            Log.d(DATATAG, "Received Data: " + dataString);
            intent.putExtra(EXTRA_DATA, dataString);
            dataString = BuildConfig.FLAVOR;
        } else {
            dataString = dataString.concat(dialData);
            return;
        }

    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
    }
```

3) DeviceControlAcitivity 經由 BroadcastReceiver mGattUpdateReceiver, 接收 Broadcast，並且呼叫 displayData()，顯示對應資料


```
} else if (BluetoothLeService.ACTION_DATA_AVAILABLE.equals(action)) {
    //display data
    displayData(intent.getStringExtra(BluetoothLeService.EXTRA_DATA));
}
```

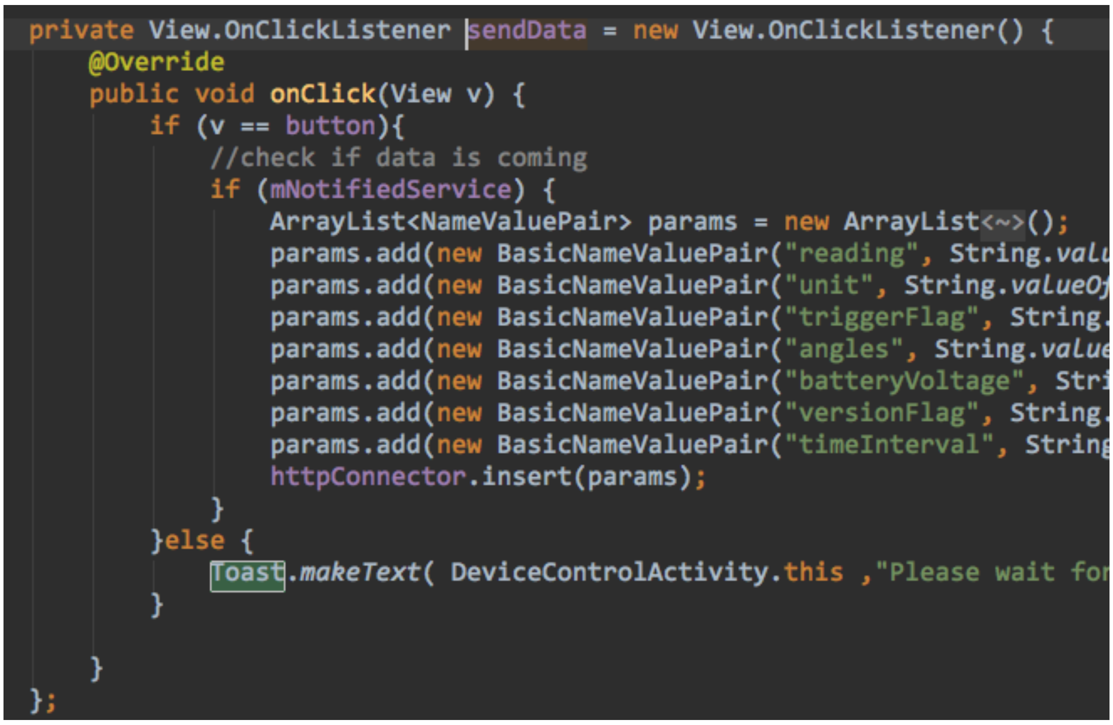
4) displayData,拆解資料並且更新 UI



```
private void displayData(String data) {
    if (data != null) {
        //Notice: This is design for motionics sensor
        this.mNotifiedService = true;
        final String[] tmp = data.split(",");

        try {
            this.readingValue = Double.parseDouble(tmp[1]);
            this.angle1 = Integer.parseInt(tmp[4]);
            this.angle2 = Integer.parseInt(tmp[5]);
            this.angle3 = Integer.parseInt(tmp[6]);
            this.battery_voltage = Double.parseDouble(tmp[7].substring(0, 2));
            this.unit = Integer.parseInt(tmp[2]);
            unit_label = unitLabel[unit];
            //Log.d(TAG,"Unit:"+tmp[2]);
            unit_label = unitLabel[unit];
        } catch (Exception e){
            e.printStackTrace();
        }

        mReadingField.setText( readingValue.toString() + " " + unit_label);
        this.mAngleField.setText(String.format("%d,%d,%d",angle1,angle2,angle3));
        this.mBatteryField.setText(String.format("%.1f volt",battery_voltage));

        mDataField.setText(data);
    }
}
```

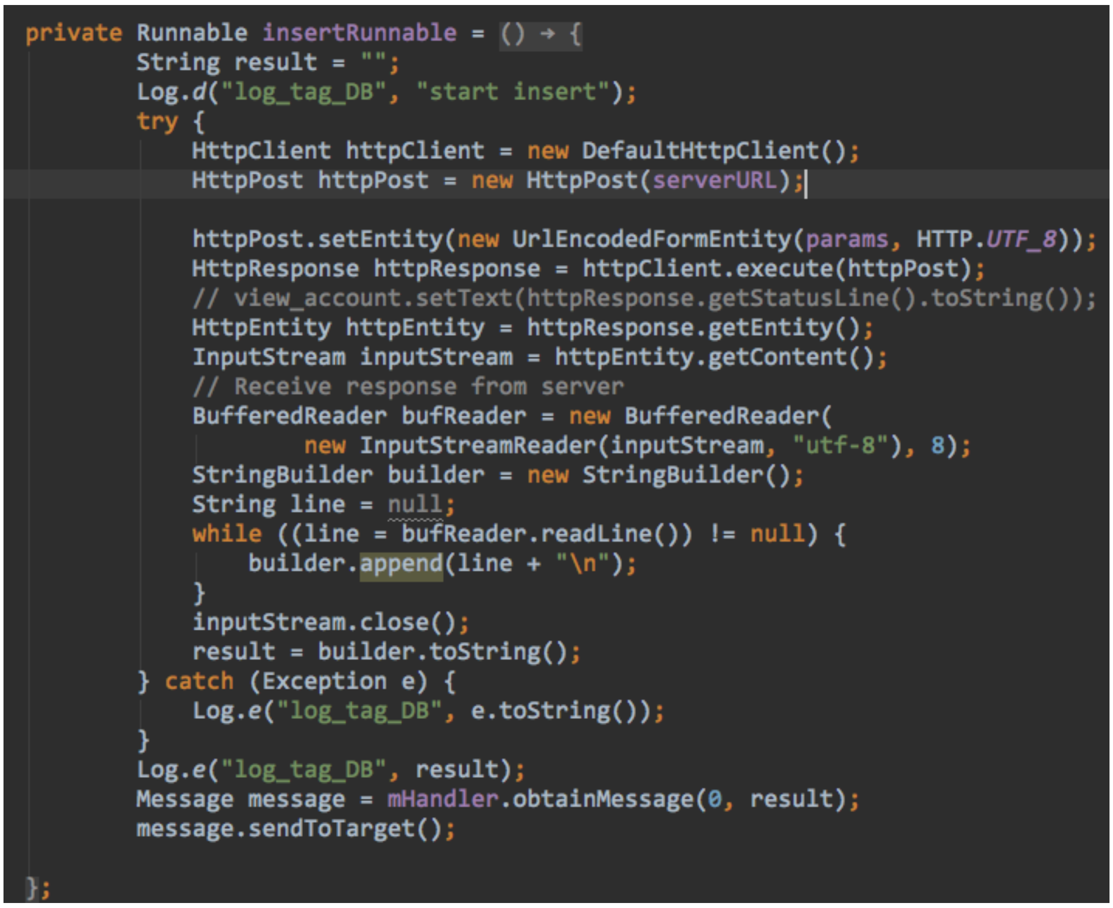
3. 上傳資料庫

   1) DeviceControlAcitivity 實作 Upload 按鈕的 onClickListener(),最後經由 HttpConnector 以 POST 方式傳回資料庫(RESTful API)



```
private View.OnClickListener sendData = new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (v == button){
            //check if data is coming
            if (mNotifiedService) {
                ArrayList<NameValuePair> params = new ArrayList<~>();
                params.add(new BasicNameValuePair("reading", String.valu
                params.add(new BasicNameValuePair("unit", String.valueOf
                params.add(new BasicNameValuePair("triggerFlag", String.
                params.add(new BasicNameValuePair("angles", String.value
                params.add(new BasicNameValuePair("batteryVoltage", Stri
                params.add(new BasicNameValuePair("versionFlag", String.
                params.add(new BasicNameValuePair("timeInterval", String
                httpConnector.insert(params);
            }
        }else {
            Toast.makeText( DeviceControlActivity.this ,"Please wait for
        }

    }
};
```

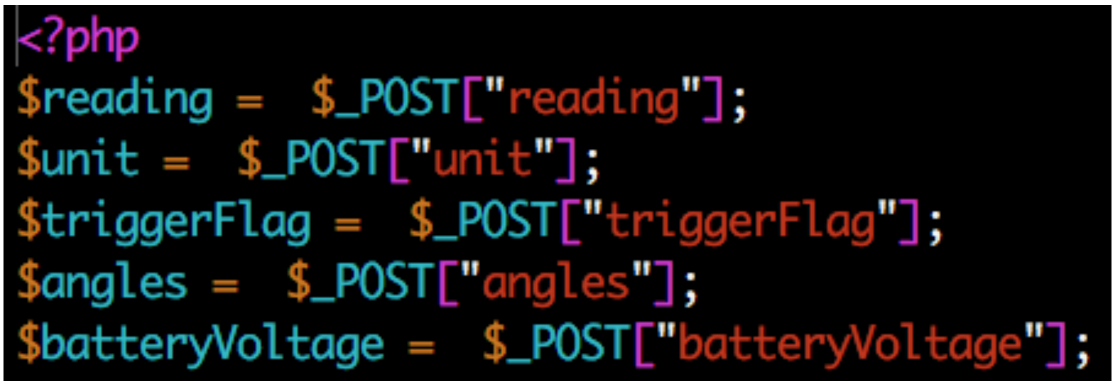
   2) HttpConnector 中實作 Runnable Thread 建立 http post，並且接收資料狀態



```java
private Runnable insertRunnable = () → {
        String result = "";
        Log.d("log_tag_DB", "start insert");
        try {
            HttpClient httpClient = new DefaultHttpClient();
            HttpPost httpPost = new HttpPost(serverURL);

            httpPost.setEntity(new UrlEncodedFormEntity(params, HTTP.UTF_8));
            HttpResponse httpResponse = httpClient.execute(httpPost);
            // view_account.setText(httpResponse.getStatusLine().toString());
            HttpEntity httpEntity = httpResponse.getEntity();
            InputStream inputStream = httpEntity.getContent();
            // Receive response from server
            BufferedReader bufReader = new BufferedReader(
                    new InputStreamReader(inputStream, "utf-8"), 8);
            StringBuilder builder = new StringBuilder();
            String line = null;
            while ((line = bufReader.readLine()) != null) {
                builder.append(line + "\n");
            }
            inputStream.close();
            result = builder.toString();
        } catch (Exception e) {
            Log.e("log_tag_DB", e.toString());
        }
        Log.e("log_tag_DB", result);
        Message message = mHandler.obtainMessage(0, result);
        message.sendToTarget();

};
```

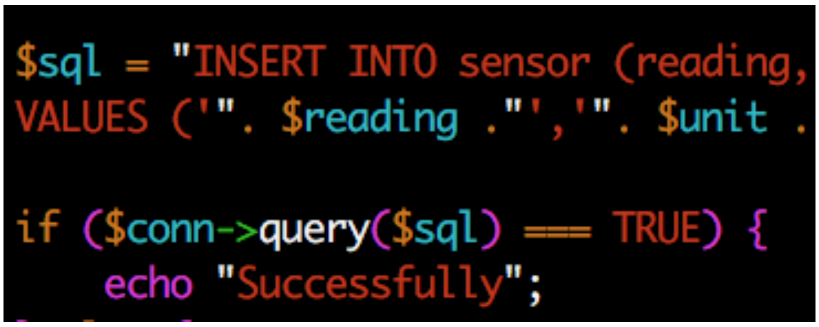
3) Server side:

由 php 接收 post 並寫入資料庫



```php
<?php
$reading =  $_POST["reading"];
$unit =  $_POST["unit"];
$triggerFlag =  $_POST["triggerFlag"];
$angles =  $_POST["angles"];
$batteryVoltage =  $_POST["batteryVoltage"];
```

4) 寫入資料庫



```php
$sql = "INSERT INTO sensor (reading,
VALUES ('". $reading ."','". $unit .

if ($conn->query($sql) === TRUE) {
    echo "Successfully";
```