Report

Group 11
吳侑學、林哲賢

這次的報告我們總共使用了 4 種傳感器

Accelerometer	accelerometer.js
Camera	camera.js
Climate	climate.js
RFID	rfid.js

一、 功能 function 架構

1. Client Browser HTTP request

```
set_handler('GET /acc.html',do_output_acc);
set_handler('GET /climate.html',do_output_cli);
set_handler('GET /rfid.html',do_output_rfid);
set_handler('GET /camera.html',do_output_camera);
```

我們新增了上面的 http request，分別對應四種傳感器，讓 client 去 GET 到每個傳感器的 html 頁面

```
set_handler('POST /data_acc',do_data_acc);
set_handler('POST /data_cli',do_data_cli);
set_handler('POST /data_rfid',do_data_rfid);
set_handler('GET /picture.jpg',do_output_pic);
```

除此之外，上面四個 http request 是為了按下按鈕(或更新頁面)後，在各個傳感器對應的頁面上，顯示出 data，來源分別是

accel.db

climate.db

rfid.db

picture.jpg

2. Device HTTP request

```
set_handler('POST /photo', do_camera);
set_handler('POST /acc', do_accel);
set_handler('POST /rfid', do_rfid);
set_handler('POST /climate', do_climate);
```

上面四個 handler 能夠處理來自傳感器端的 request，把 data 儲存在 server，並根據不同的 data 回應不同的 response，如下:

a. Accelerometer

```
var check = false;
for(var i=0;i<accel_data.length;i++){
 var cal = accel_data[i].x*accel_data[i].x + accel_data[i].y*accel_data[i].y + accel_data[i].z*accel_data[i].z
 var cal = Math.sqrt(cal);
 console.log(cal);
 if(cal>1.1){
  check = true;
  console.log("move");
  break;
 }
}
if(check){
 send_response(new Buffer("legal"), {'Content-Type': 'text/plain;charset=UTF-8'});
}
else{
 send_response(new Buffer("OK"), {'Content-Type': 'text/plain;charset=UTF-8'});
}
```

除了將 data 存到 accel.db 當中之外，do_accel 這個 function 還會計算絕對加速度，如果 cal>1.1，就會 response: legal，device 端收到這個 response 就會呼叫 do_audio 這個 function，並撥出”no.mp3”音效檔；反之，則 response: OK，這樣一來就不會撥出音效檔

```
var do_audio = function (input){
    audio.setVolume(.8);
    if (input==='legal'){
     var audioFile = fs.readFileSync('no.mp3');
     audio.play(audioFile, function(err) {});
    }
```

(這裡我們假想的應用模式是放在門上，若感測到加速度，就會在 server 留下紀錄，並撥出”no.mp3”，跟進入的人說「you shall not pass」)

b. Camera

```
var do_camera = function(send_response, request_body, request_headers, time){
  var imageBuffer = new Buffer(request_body);
  require('fs').writeFile("picture.jpg", imageBuffer , function(err){
    if (err) throw err;
  });
  send_response(new Buffer("legal"), {'Content-Type': 'text/plain;charset=UTF-8'});
};
```

這裡會將收到的 request body 存成 picture.jpg，如果沒有 error 的話會 response: legal。而 device 端接受到這個 response 就會撥出 ”shutter.mp3”，撥出快門的聲音

```
var do_audio = function (input){

    audio.setVolume(.8);
    if (input==='legal'){
     var audioFile = fs.readFileSync('shutter.mp3');
     audio.play(audioFile, function(err) {});
    }
```

c. Climate

```
for(var i=0;i<climate_data.length;i++){
  if(climate_data[i]["Degrees"]>91){
    check = 1;
    break;
  }
  else if(climate_data[i]["Degrees"]<89){
    check = 2
    break;
  }
}
if(check===1){
  send_response(new Buffer("hot"), {'Content-Type': 'text/plain;charset=UTF-8'});
}
else if(check===2){
  send_response(new Buffer("cold"), {'Content-Type': 'text/plain;charset=UTF-8'});
}
else{
  send_response(new Buffer("OK"), {'Content-Type': 'text/plain;charset=UTF-8'});
}
```

這裡會將收到的 data 存成 climate.db，並根據溫度的高低 response: hot, cold 或 OK。
而 device 端則會根據 response 撥出「hot.mp3」或「cold.mp3」(或不撥)

```
var do_audio = function (input){
    audio.setVolume(.8);
    if (input==='hot'){
     var audioFile = fs.readFileSync('hot.mp3');
     audio.play(audioFile, function(err) {
     });
    }
    if (input==='cold'){
     var audioFile = fs.readFileSync('cold.mp3');
     audio.play(audioFile, function(err) {
     });
    }
}
```

d. RFID

```
var do_rfid = function(send_response, request_body, request_headers, time){
  var rfid_data = JSON.parse(request_body.toString());

  require('fs').readFile('rfid.db', function(err,data){
    if (err) throw err;
    else{

      if(data.length ===0){
        var rfidOBJ = new Array;
        rfidOBJ[0] = rfid_data;
        send_response(new Buffer("legal"), {'Content-Type': 'text/plain;charset=UTF-8'});
      }
      else{
        var rfidOBJ = JSON.parse(data.toString());
        rfidOBJ[rfidOBJ.length] = rfid_data;

        send_response(new Buffer("legal"), {'Content-Type': 'text/plain;charset=UTF-8'});
      }
      require('fs').writeFile('rfid.db', JSON.stringify(rfidOBJ), function(err,data){
          if(err) throw err;});
    }
  });
};
```

這裡會將 UID 存成 rfid.db，紀錄有誰刷過卡，並回應 legal 的 response，在 device 端會撥出「familymart.mp3」歡迎使用者進門

```
var do_audio = function (input){
    audio.setVolume(.8);
    console.log("aaa");
    console.log(input);
    console.log(typeof(input));
    if (input==='legal'){

     var audioFile = fs.readFileSync('familymart.mp3');
         audio.play(audioFile, function(err) {});
    }
}
```

二、 網頁圖表化

網頁的圖表化主要是在 d3.js 上，將 accelerometer 和 temperature humidity 等等 data 做成折線圖觀察其趨勢線，利用上次實作的留言板的介面，處理網頁端和 server 端的 request 和 response 來取得 data，相較之下並沒有遇到太大的困難，只是需要熟悉 d3 這個 library 的使用方法，花了較多時間

三、 心得與困難

這次的實驗由於遇到期中所以相較之下比較沒花那麼多的力氣，還有很多 d3 的功能其實沒有很深入的去研究，其實這個 library 十分強大，對於許多資料的視覺化相當有幫助，圖表甚至可以做成像 3D 的，利用三軸加速器所測量到的數據，去真正模擬物體本身的移動狀況，只是礙於時間因素沒有辦法實作出來，還是有點小小遺憾，不過我想這次主要的工作還是在熟悉 tessel 版上的應用，和實踐將 tessel 所收集的資料傳回 server 的功能是最主要的 effort，其實使用後發現 tessel 並不是非常 robust，常常會有一些比較難以理解的 bug，可能也是因為我們對於這種嵌入式系統的架構並沒有非常熟悉，所以對於一些 thread 和 process 的問題產生的 bug 毫無頭緒，最後也是查了一些資料才理解為什麼我們這樣做並不會像我們想像的 run 出結果，這應該也是一個在使用嵌入式系統上常遇見的問題吧，我們平常都是在 PC 的 OS 上在寫程式和 debug，當把這些移到板子上時，並不一定像我們平常在電腦上所見的跑，這可能就需要更多經驗的累積，而且在不同板子上可能又會遇到截然不同的問題，像上學期生醫實驗所使用的 Arduino 就與現在差異極大，不過我想在克服一些難題後，這門課的所學應該能將我們心目中想完成的 prototype 真的實踐出來，並且再利用一開始所學的 http 技巧，利用 internet 做成一個 IOT 的 device，這也是這門課有趣的地方。