

# 嵌入式 final project

Say " Hey siri " with Arduino

105062139 林楷宸

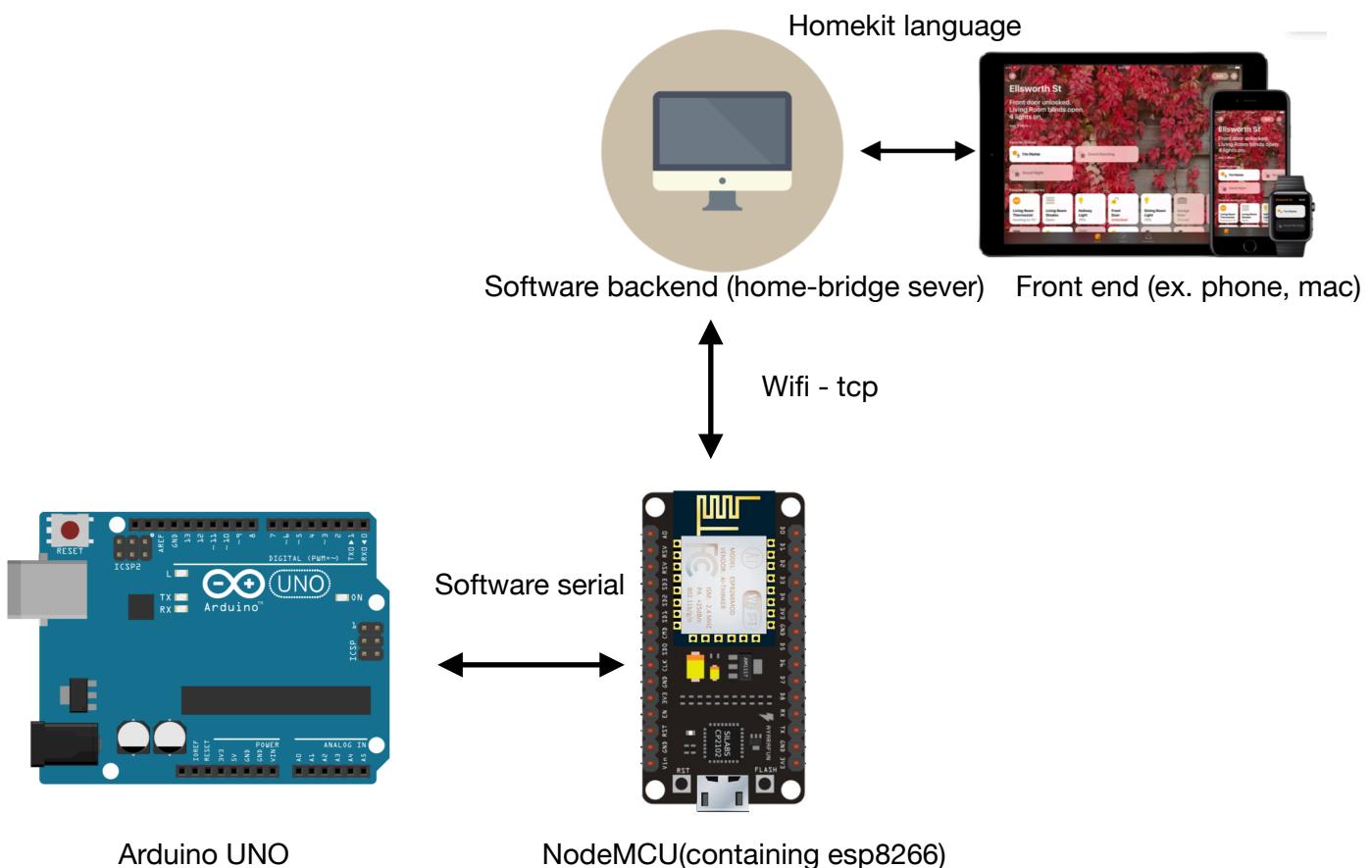
## Goal

科技日新月異，並影響了我們的生活，如今科技所帶來的便利性，已成為我們生活不可或缺的一部份。是否想過，忙碌了一天，回家只想要靜靜的躺在沙發上，一聲「Hey, siri」，便能解決所有事情，不必為了開個燈而不情願的從沙發爬起，天氣冷時，一聲「Hey Siri，幫我開暖氣」，亦或是，在回家的途中，叫 siri 幫你開冷氣，回到家便便能有個舒適的環境。

智能家居已是個火熱的議題。然而現今的設備價格並不親民，一個「智慧電燈」可能就要數百塊，甚至數千元。而不同廠商的設備又不能共用，這使智能家居在推廣上困難重重。

因此，這次我的目標是，利用 arduino 搭配 siri，在有限的價格與設備下，實作出相同的成果。

## Hardware Flow Chart

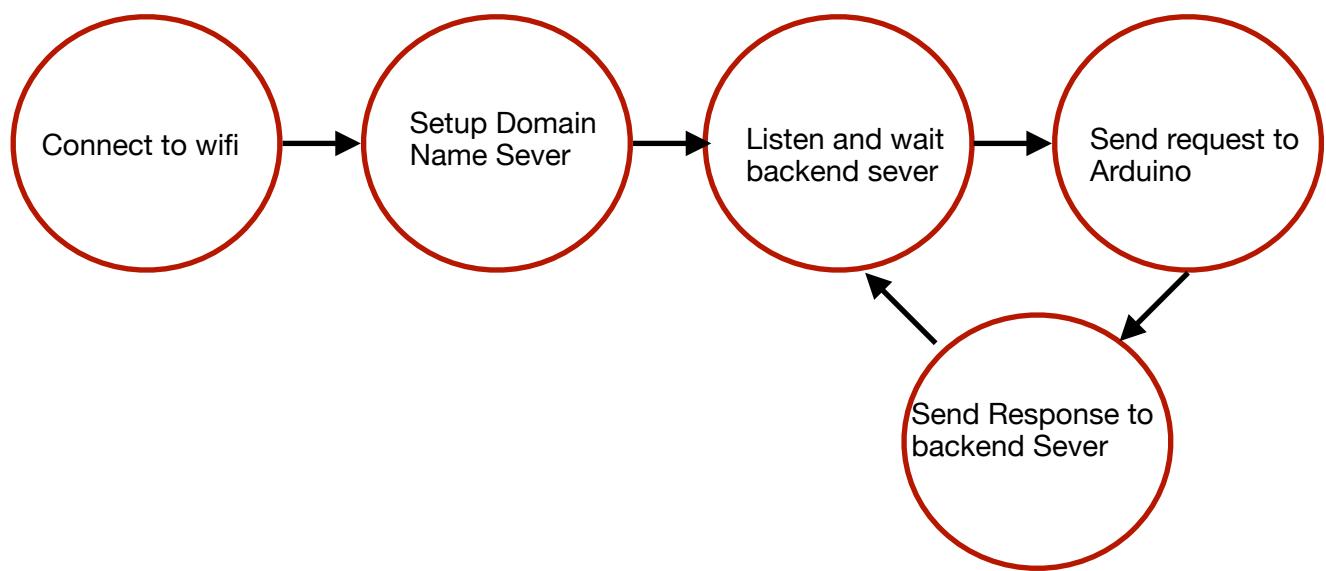


Arduino UNO

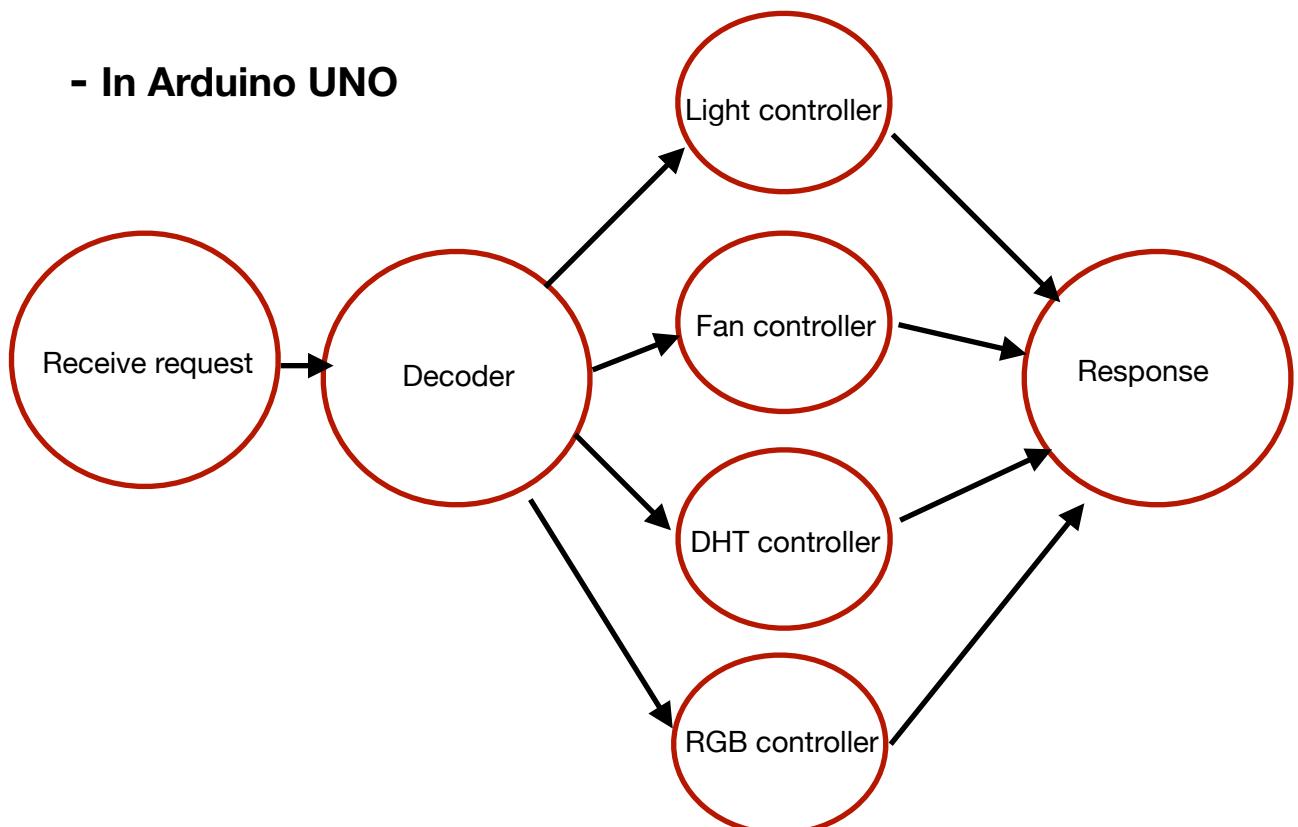
NodeMCU(containing esp8266)

## Program Flow Chart

### - In NodeMCU



### - In Arduino UNO



## Components

### - Front End

採用 Apple iPhone 內建的 Home - Kit App 當作前端，好處是消費者的COST小，不用為了產品多多下載一個 APP。



Home Kit 介面，搭載於所有 Apple 產品上，任何時候都能使用。

### - Software Backend

採用 HomeBridge sever，為一個 node.js 的後端，以此為基礎，新增 module 跟修改 code，由於我們是採用 HTTP 為 backend 與硬體的溝通語言，因此必須自定義 GET request，撰寫 config 檔讓 sever 能夠自行發送 config。

右圖為 config 的格式，必須

自定義大量的配件跟 http

request

kaichenHome.local 為自定義

Domain Name，須借由

NodeMCU 設定 Domain

Name Sever

```

58      {
59          "status": "http://kaichenHome.local/RGB/brightness/status",
60          "url": "http://kaichenHome.local/RGB/brightness/%s"
61      },
62
63      "color":
64      {
65          "status": "http://kaichenHome.local/RGB/color/status",
66          "url": "http://kaichenHome.local/RGB/color/%s",
67          "brightness": true
68      },
69
70      {
71          "accessory": "HTTP-RGB",
72          "name": "臥室的燈",
73          "service": "Light",
74          "switch":
75          {
76              "status": "http://kaichenHome.local/led/1/status",
77              "powerOn": "http://kaichenHome.local/led/1/on",
78              "powerOff": "http://kaichenHome.local/led/1/off"
79          }
80      },
81
82      {
83          "accessory": "HTTP-RGB",
84          "name": "浴室的燈",
85          "service": "Light",
86          "switch":
87          {
88              "status": "http://kaichenHome.local/led/2/status",
89              "powerOn": "http://kaichenHome.local/led/2/on",
90              "powerOff": "http://kaichenHome.local/led/2/off"
91      },
92
93      {
94          "accessory": "HTTP-RGB",
95          "name": "廚房的燈",
96          "service": "Light",
97          "switch":
98          {
99              "status": "http://kaichenHome.local/led/3/status",
100             "powerOn": "http://kaichenHome.local/led/3/on",
101             "powerOff": "http://kaichenHome.local/led/3/off"
102         }
103     }

```

接著，將這些 http request 廣播到區網裡，由 Wifi 模組接收並傳給 Arduino 進行 decode。

將來可以將這個 sever 架在樹莓派上，便能當作家管使用，不必再多開一台電腦當作 sever 使用。

## - NodeMCU

NodeMCU 是一塊擁有 esp8266 (wifi 模組) 的開發版，可使用 Arduino editor 來開發，因此語言跟 Arduino 一樣，但需要其他的 library 來開發。

2019年1月14日 星期一

Connect to wifi：由於他必須接收 Wifi 的封包，因此需要擁有 wifi client 的權限，利用 WiFiServer 來設定

```
bool TryConnecting(char* ssid, char* password)
{
    String connectingMesg = "try to connect to " + String(ssid);
    Serial.println(connectingMesg);

    WiFi.hostname(HostName);
    WiFi.mode(WIFI_STA);
    WiFi.begin(ssid, password);

    // waiting response
    int times = 0;
    while (WiFi.status() != WL_CONNECTED && times < 15)
    {
        Serial.print(".");
        delay(500);
        times++;
    }

    Serial.println("");
    if (times < 15) return true;
    else         return false;
}
```

由於我們需要從 pool 裡面拿 ip，因此 ip address 並非固定，因此需要自己架一個 Domain Name Sever，讓 backend 可以使用自定義 Domain Name 來傳封包，

```

76 // set domain name
77 while (!MDNS.begin("kaichenHome"))
78 {
79     Serial.println("Error setting up MDNS responder!");
80     delay(500);
81 }
82
83 Serial.println("mDNS responder started");
84
85 // start server
86 server.begin();
87 Serial.println("Server started");
88
89 // add service to MDNS-SD
90 MDNS.addService("http", "tcp", 80);
91 }
92

```

### 利用 <ESP8266mDNS.h> 來架 DNS

最後兩塊板子（Arudino UNO 跟 NodeMCU）的溝通，由於兩塊板子並非 slave 跟 master 的關係，因此不能用上課教的方式（<Wire.h>）來進行通訊，此外不能用 Serial 來通訊（要 debug），這裡採用一個新的 library，<SoftwareSerial.h> 利用 port 8 跟 port 9 來進行通訊（一個是 RX 一個是 TX）。

然而，各種通訊的 library 都有 buffer 的限制，但是我們要回傳的 response 可能是一個 html 或是 json，勢必會超過 buffer，因此這裡設計了一個 hand shaking protocol 來進行溝通。（詳情看 difficulties）

## - Arduino

Decoder：將 GET request 解碼，並分到各個 controller 進行控制。

```
void decoder(String req)
{
    if      (req.indexOf("/led") != -1)      ledControl(req);
    else if (req.indexOf("/dht") != -1)       dhtControl(req);
    else if (req.indexOf("/RGB") != -1)       RGBControl(req);
    else if (req.indexOf("/fan") != -1)       fanControl(req);
    else                                     Serial.println("error request");
}
```

Light Controller：需要區分三種 request：on, off, status，並回傳一個 text/html 的 response。

Dht Controller：利用 <SimpleDHT.h> 來測量溫濕度，最後回傳一個 json string 回去，利用 timer 來控制讀取溫濕度。

RGB Controller：需要 handle on, off, status, brightness/set, brightness/status, color/set, color/status。並回傳一個 text/html 的 response。

必須將 decimal string 轉成 hex integer，抑或是 hex string 傳成 decimal integer 等等。

Response Event：基於前面提到回傳的格式不同跟 hand shaking protocol，因此，需要設計一個功能強大且齊全的 response function。

右圖為 response 的 function，將要回傳的值進行包裝，轉成封包的格式，並加入各種格式的 title。

Start sending 的部分是自己設計的一個 hand shaking protocol。

```
// 回傳訊號
void requestEvent(String res, int type)
{
    String response = "";
    String sending = "";

    if (type == text)
    {
        response += "text/html\r\n\r\n";
        response += res;
    } else if (type == json)
    {
        response += "application/json\r\nCache-Control: no-store\r\n\r\n";
        response += "{";
        response += res;
        response += "}";
    }

    Serial.print("response:");
    Serial.println(response);
    Serial.flush();

    // start sending
    while (response.length() > 20)
    {
        sending = response.substring(0, 20);
        response = response.substring(20);

        mySerial.print(sending);
        mySerial.flush();
        while ( !mySerial.available()) delay(20);
    }

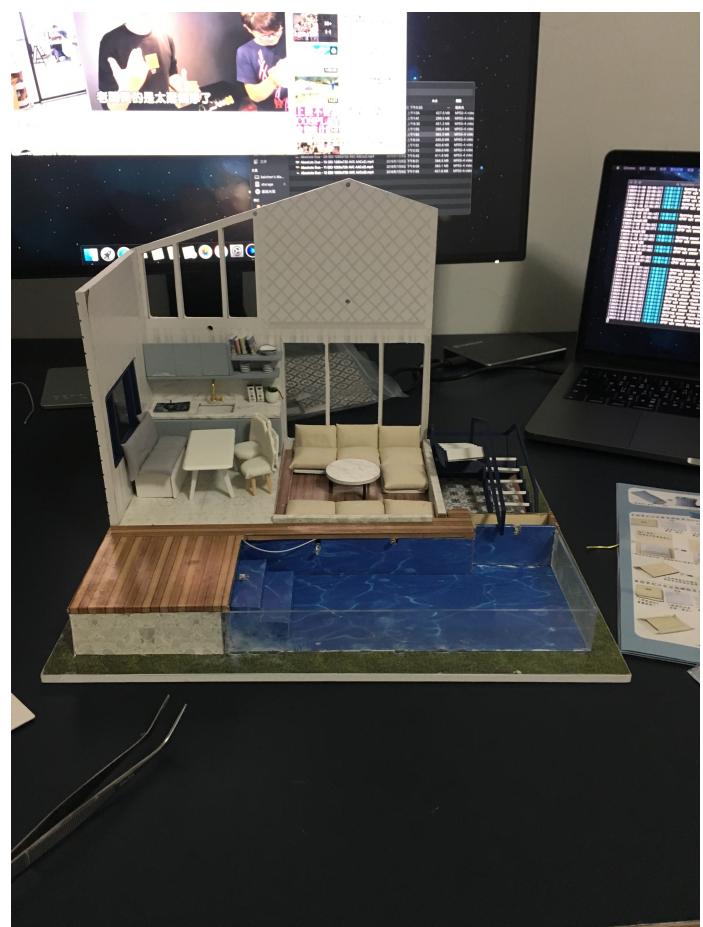
    mySerial.print(response);
    mySerial.flush();

    // send end signal
    delay(100);
    mySerial.print("fine");
    mySerial.flush();
}
```

2019年1月14日 星期一

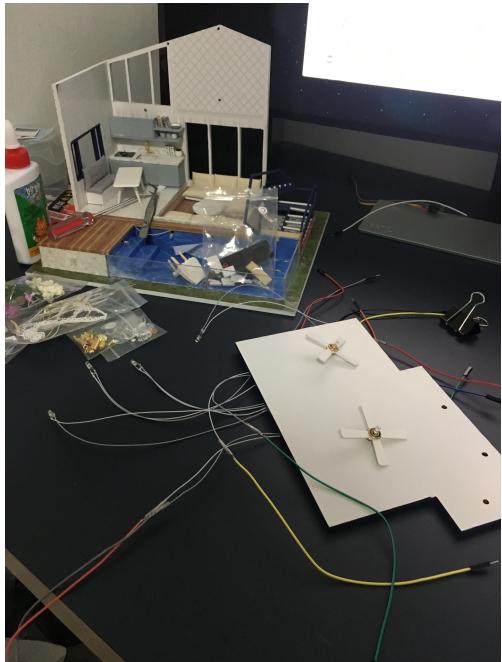
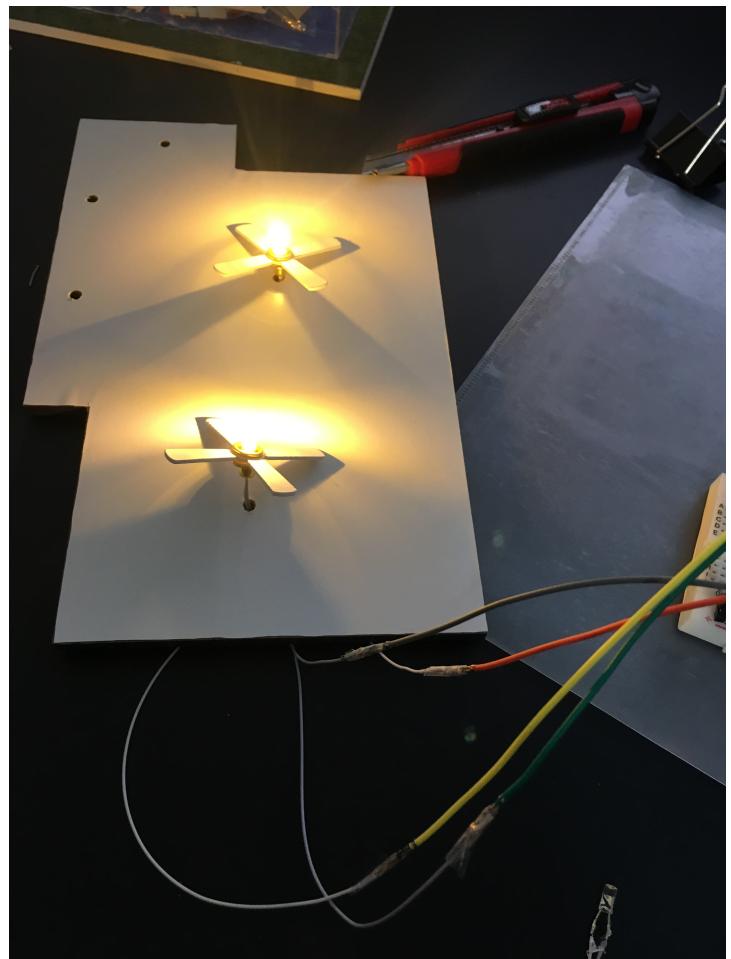
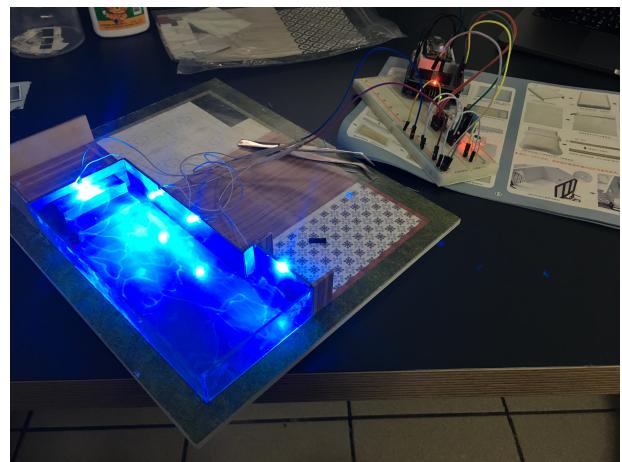
## - 蓋房子

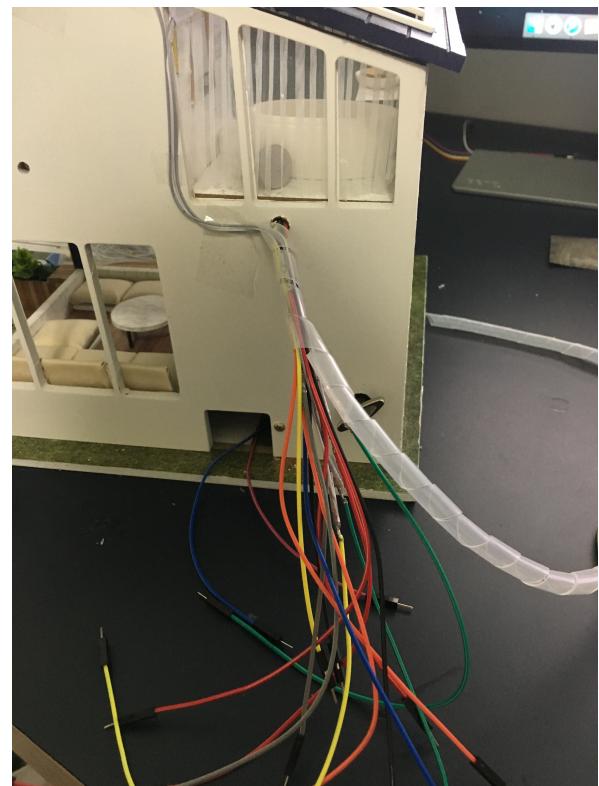
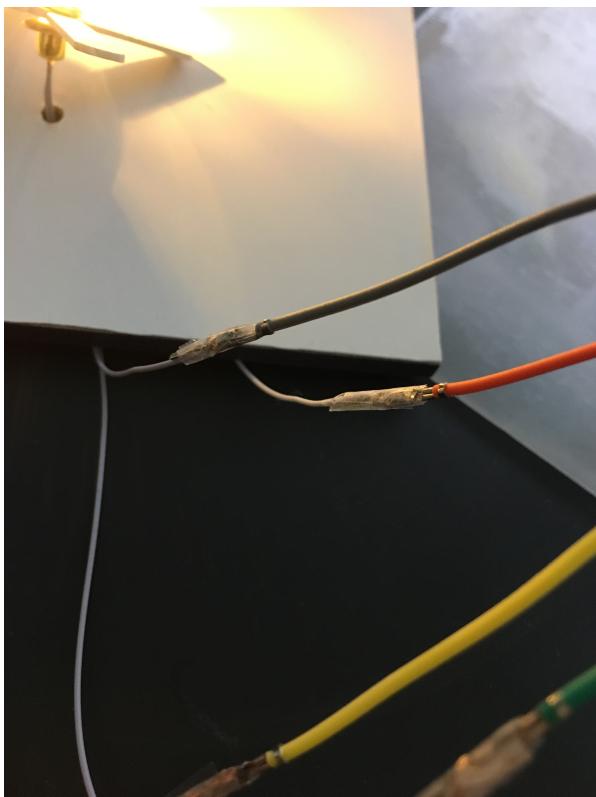
此次 project 最難的部分之一。

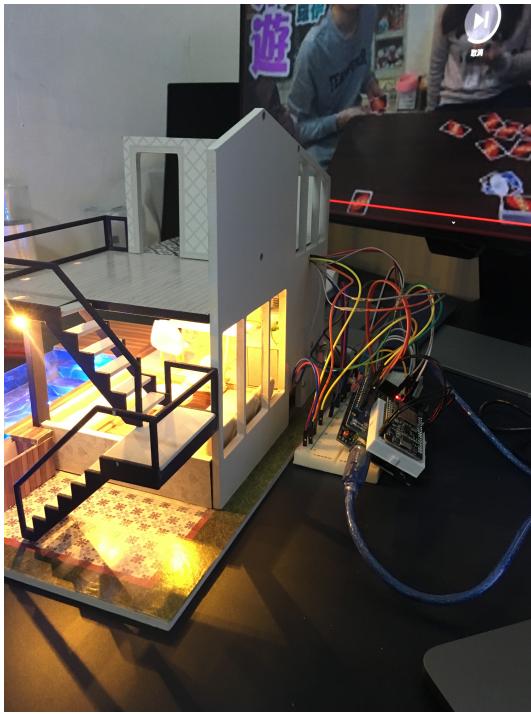




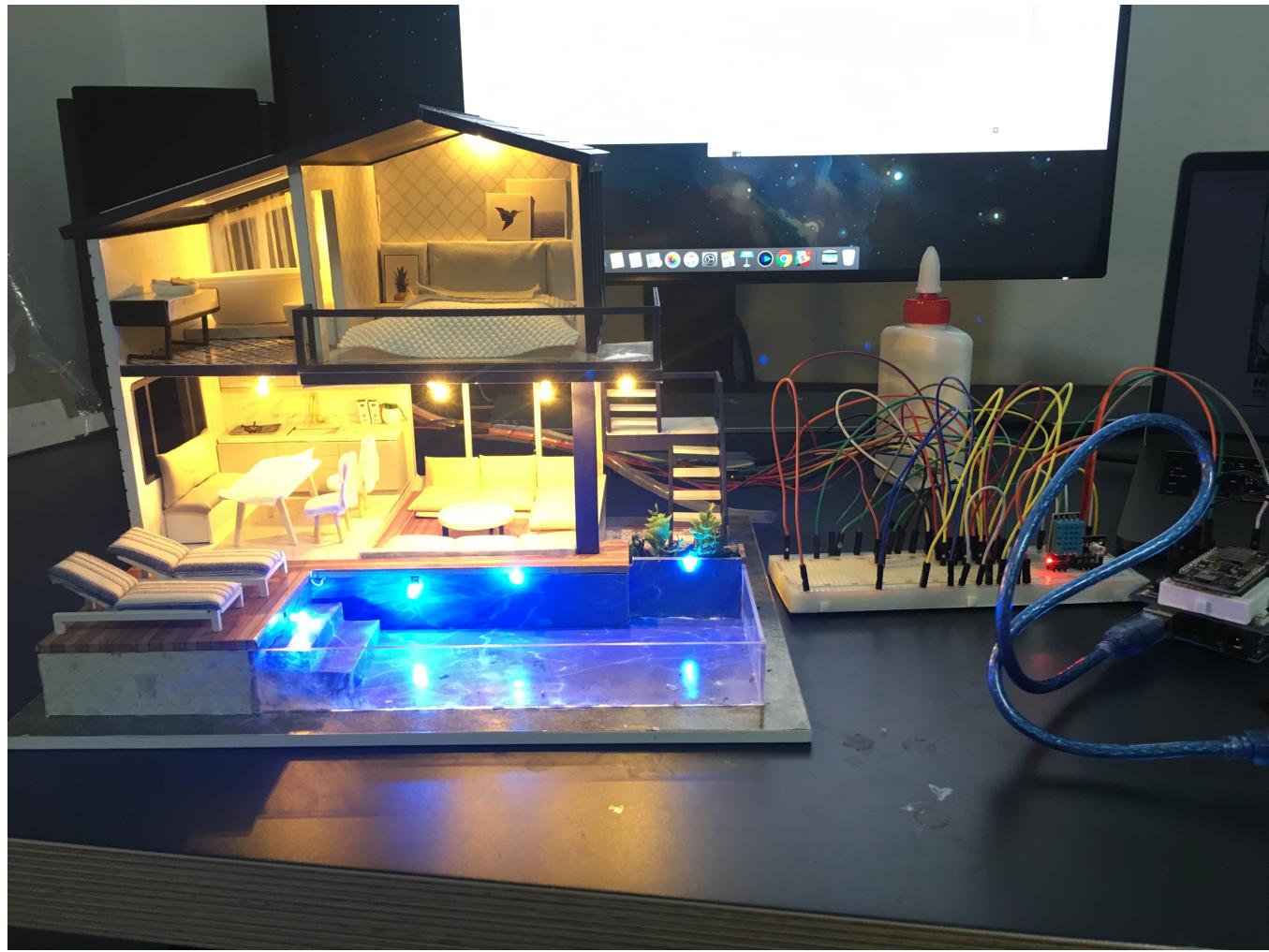
2019年1月14日 星期一







2019年1月14日 星期一



## difficulties

### - 兩塊板子溝通

由於兩塊板子的傳輸量很大，且並沒有 master 跟 slave 的關係又不能用 serial，因此才選擇使用 software serial 來進行溝通，然而，兩塊板子的傳輸又牽扯到 string 沒辦法直接傳，print 跟 write 的意義不同，以及怎麼定義 5V 到 0V 兩塊板子的容忍度不同，因此中間花了很多力氣再處理這一塊。

### - Hand shaking protocol

然而，即使能夠溝通，對於大量傳輸又是另一個問題，每個 library 都有 buffer 上限沒辦法大量傳輸，因此，只好自己開發一個 hand shaking protocol 來解決問題。

### - 建模

這一次花最多時間的不外乎就是 sever 的搭建跟 domai sever 以及蓋房子。從一堆木板、紙、皮革，到自己牽線，藏線，配線，花了整整半個月以上，手上不知道沾了多少罐三秒膠。所有的東西都是 tiny 等級的，因此對於手殘的我，每個物件的製作都很煎熬。

### - And the Others

遇到太多問題了.....。

## Feedbacks

助教非常仁慈 :)) )。