

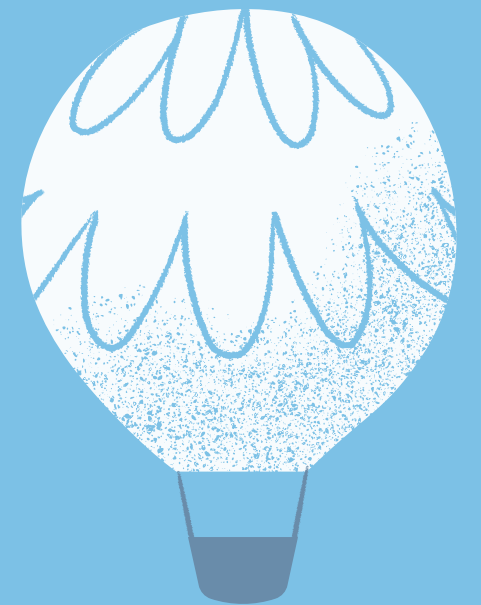


ESP32之提醒帶雨傘應用

組員：

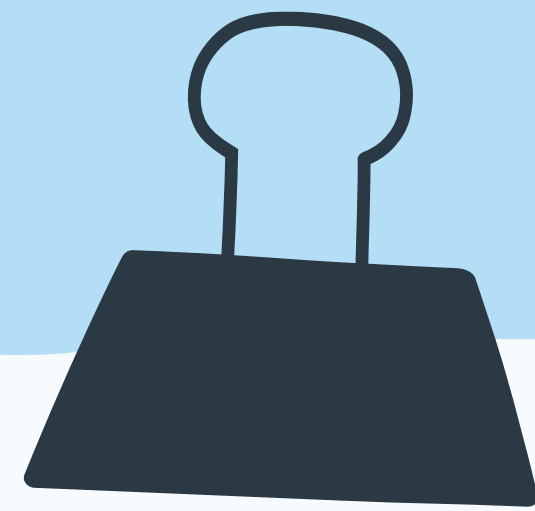
110062302 黃振瑋

110062306 郭原廷



動機

- 梅雨季
- 忘記看天氣預報
- 忘記帶傘



使用材料 - NodeMCU-32s

- 搭載 esp32 晶片
- esp32 支援 WiFi, Classic Bluetooth, BLE



使用材料 - ili9341

- 2.8 吋, 240 * 320 解析度
- 使用 SPI 通訊界面
- 用來顯示天氣



使用材料 - 有源蜂鳴器

- 通電就會叫
- 用來提醒使用者

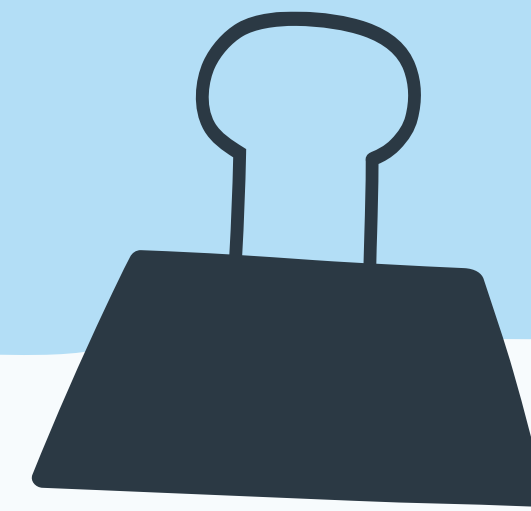


使用材料 - nRF Connect

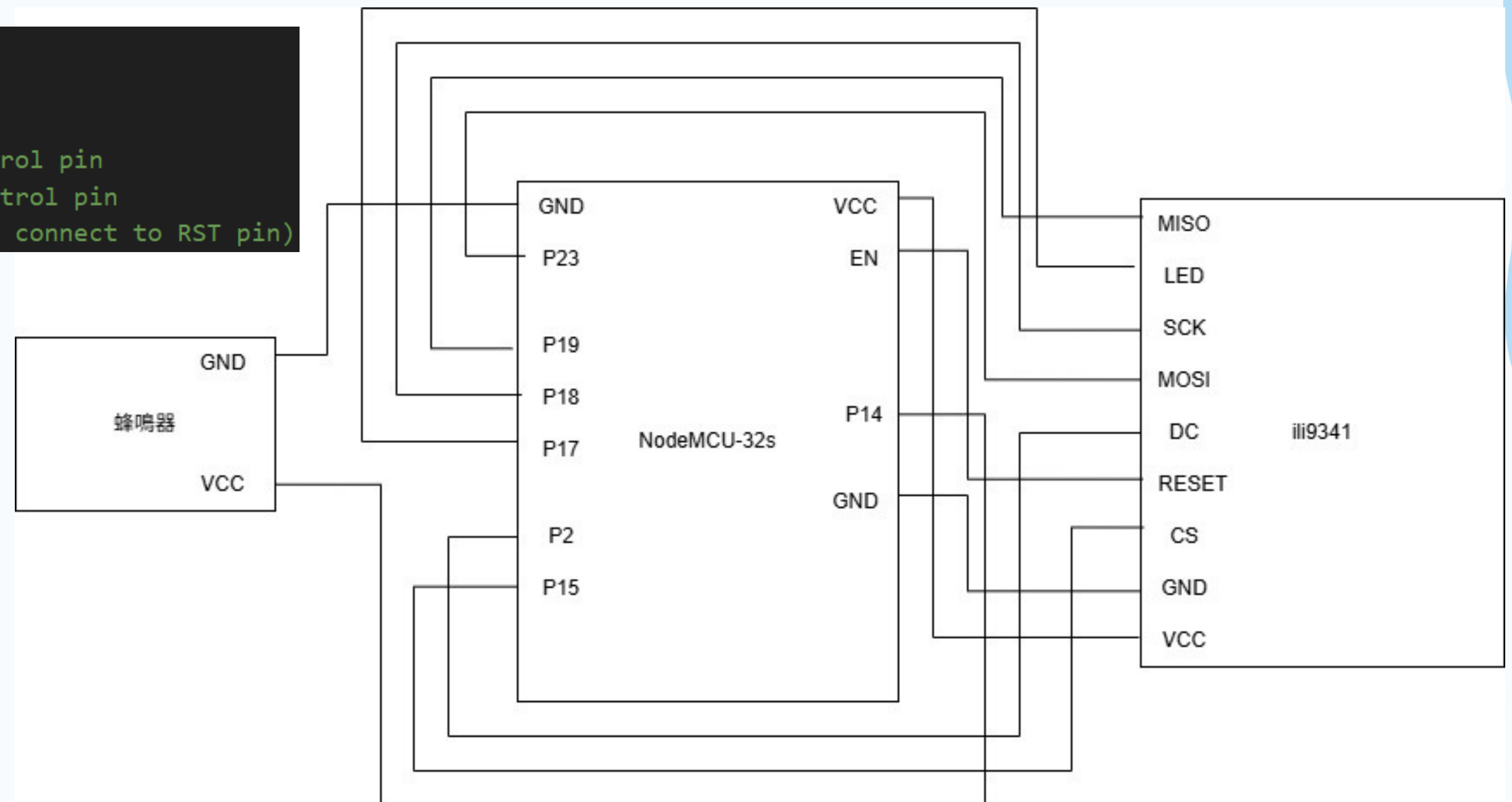
- 手機 app
- 將手機模擬成一個 BLE Beacon 發出廣播



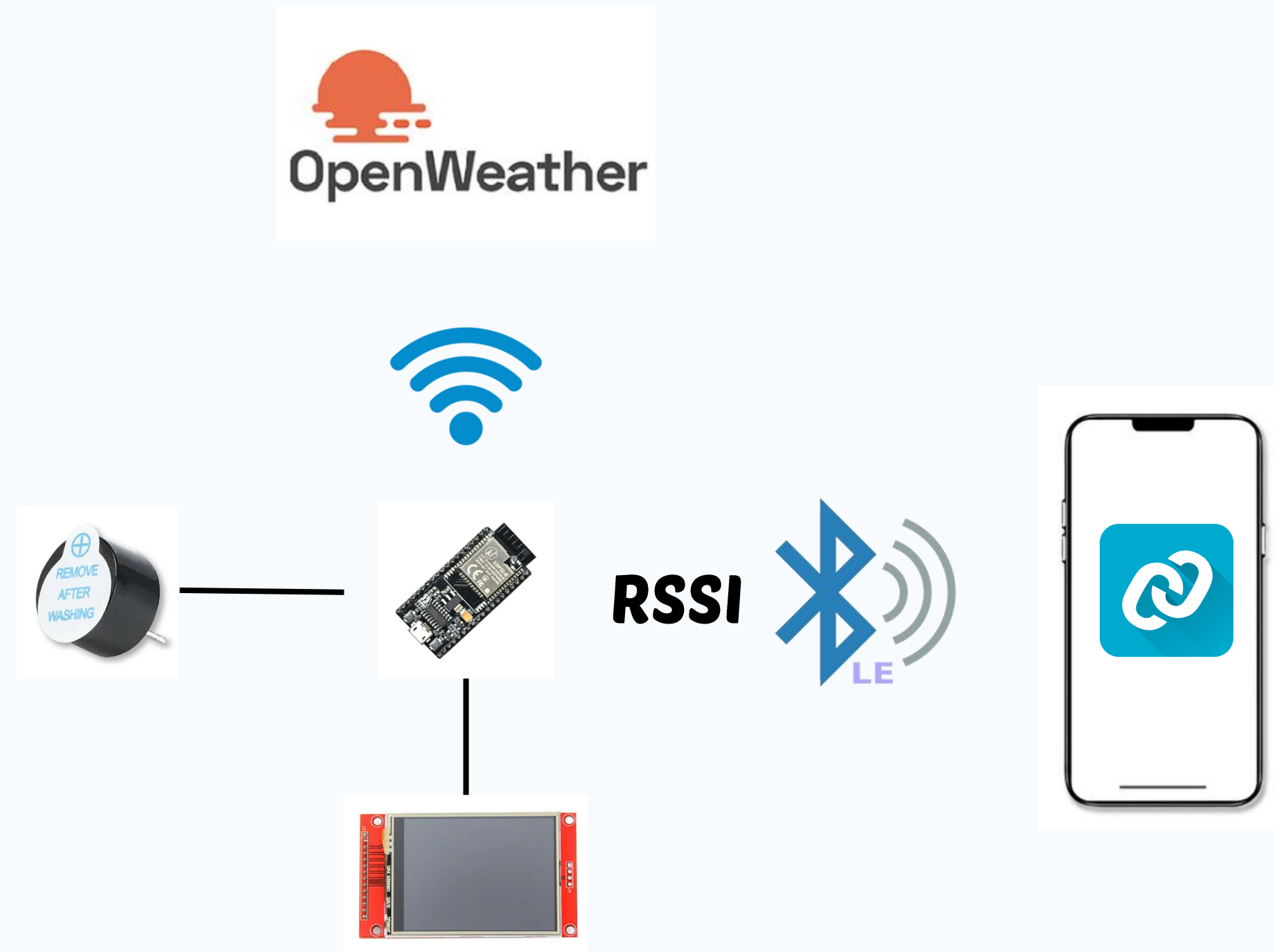
Block diagram



```
#define TFT_MISO 19
#define TFT_MOSI 23
#define TFT_SCLK 18
#define TFT_CS 15 // Chip select control pin
#define TFT_DC 2 // Data Command control pin
#define TFT_RST 4 // Reset pin (could connect to RST pin)
```



架構

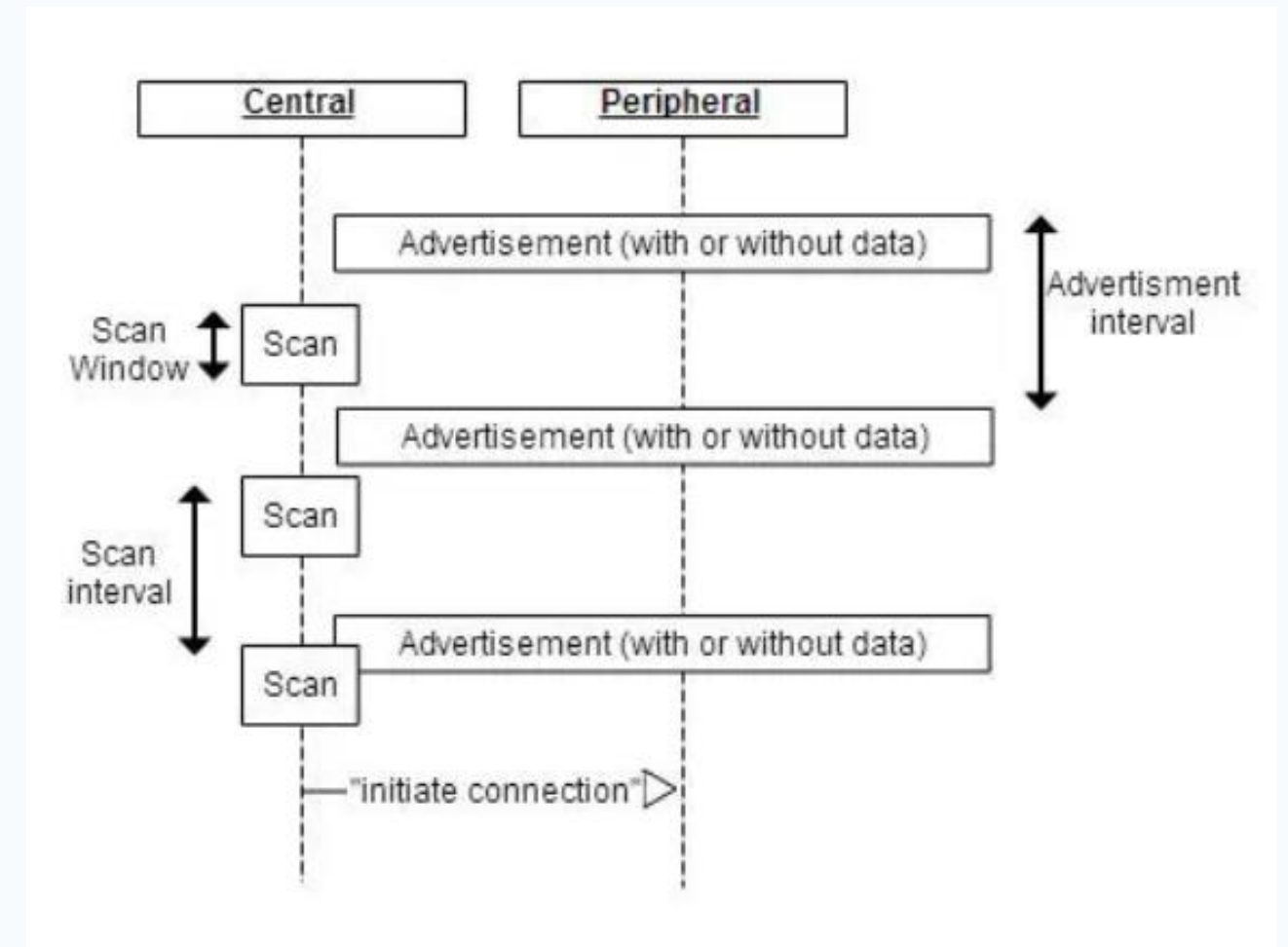


BLE (Bluetooth Low Energy)

- 低功耗
- 輕量資料傳輸
- 2.4 GHz 頻段，每個 channel 2 MHz
- 共 40 channels，0~36 為data channels，37~39 為advertising channels
- FHSS (Frequency Hopping Spread Spectrum)

BLE (Bluetooth Low Energy)

- BLE 設備分為 central 與 peripheral
- peripheral 不斷發出廣播
- central 掃描到後即可與之建立連線
- peripheral 的廣播間隔與central 的掃描時長是電池壽命與反應性的 trade off



BLE (Bluetooth Low Energy)

- esp32 擔任 central，手機擔任 peripheral
- esp32 只負責掃描取得 RSSI，不會建立連線
- 廣播封包包含 name、MAC address、RSSI
- esp32 使用 name、RSSI 判斷使用者是否靠近



實作部分 - 透過wifi 連上 openweather

- ESP32 使用 Wi-Fi 連接 AP
- 建立 HTTPS 連線
- 發送 GET 請求
- 等待 HTTP header 結束
- 開始逐字解析 JSON 資料 (使用library JSON Decoder)
- 根據是否成功解析資料，回傳 true 或 false

```
227 | WiFi.begin(WIFI_SSID, WIFI_PASSWORD);  
  
121 | WiFiClientSecure client;  
122 | client.setInsecure(); // Certificate not checked  
123 |  
124 | const char* host = "api.openweathermap.org";  
125 | port = 443;  
  
146 | client.print(String("GET ") + url + " HTTP/1.1\r\n" + "Host: "  
147 | + host + "\r\n" + "Connection: close\r\n\r\n");
```

```
3 | JSON_Decoder parser;  
4 | parser.setListener(this);
```

```
172 | while ( client.available() > 0 || client.connected())  
173 | {  
174 |     while(client.available() > 0)  
175 |     {  
176 |         c = client.read();  
177 |         parser.parse(c);
```


實作部分 - 使用BLE確認使用者是否靠近

- 宣告ESP32藍牙 BLE 掃描器的物件
- 初始化 BLE 功能並命名裝置
- 取得 BLE 掃描控制物件的指標，並儲存到 pBLEScan
- 每1秒會掃描一次附近裝置的RSSI (阻塞式)
- 呼叫Call back function

```
100 BLEScan* pBLEScan;
```

```
174 BLEDevice::init("ESP32_RSSI_SCANNER");  
175 pBLEScan = BLEDevice::getScan();  
176 pBLEScan->setAdvertisedDeviceCallbacks(new MyAdvertisedDeviceCallbacks());
```

```
pBLEScan->clearResults();  
pBLEScan->start(1, true);
```

實作部分 - 使用BLE確認使用者是否靠近(續)

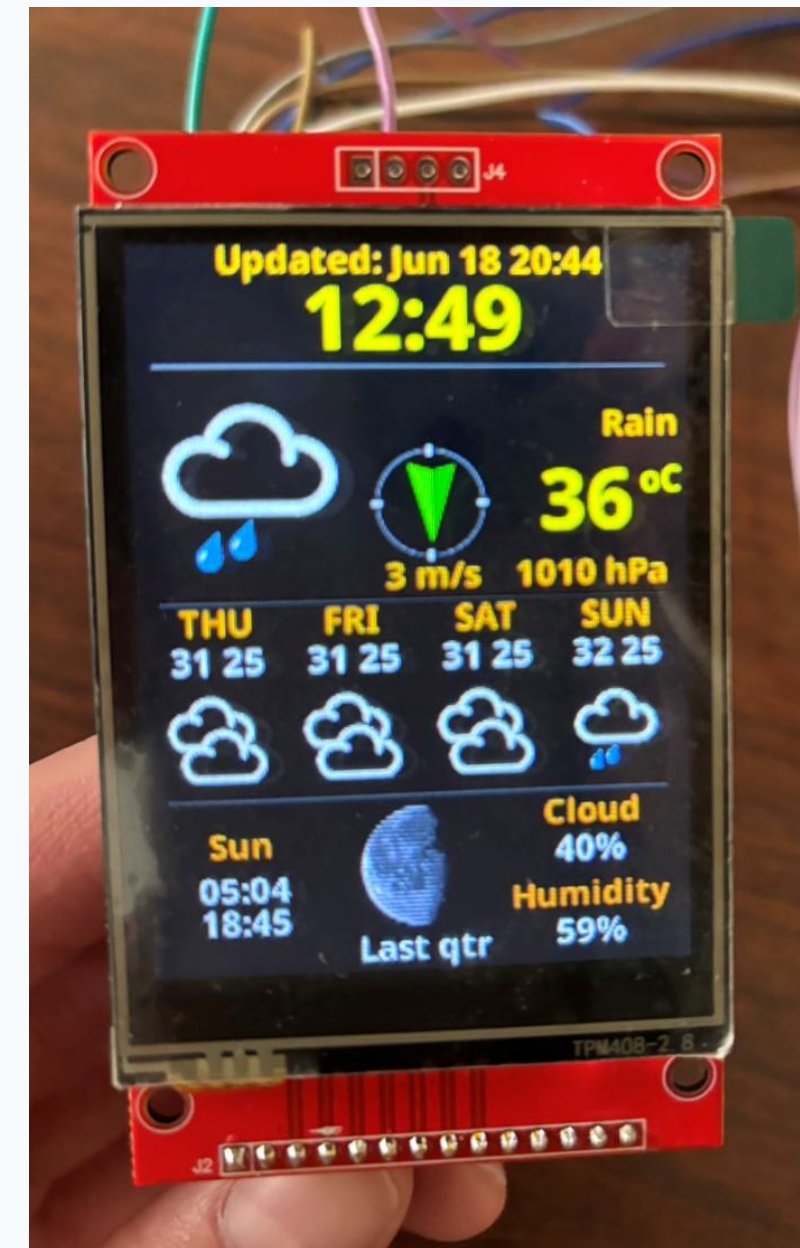
- 呼叫**Call back function**

```
8 void MyAdvertisedDeviceCallbacks::onResult(BLEAdvertisedDevice advertisedDevice) {
9     String mac = advertisedDevice.getAddress().toString().c_str();
10    int rssi = advertisedDevice.getRSSI();
11    String device_name = advertisedDevice.getName().c_str();
12    if (device_name.length() == 0) device_name = "xxx";
13
14    if (device_name.equalsIgnoreCase("realme 8 5G")) {
15        avg += rssi;
16        if (sample_count == 2) {
17            avg /= NUM_SAMPLE_TIMES;
18            if (avg >= rssiThreshold && (getMeteoconIcon(forecast->id[0], true) == "thunderstorm" || getMeteoconIcon(forecast->id[0], true) == "rain"))
19                digitalWrite(14, HIGH);
20            } else {
21                digitalWrite(14, LOW);
22            }
23            avg = 0;
24        }
25        Serial.printf("MAC: %s, RSSI: %d, Name: %s\n", mac.c_str(), rssi, device_name.c_str());
26        sample_count = (sample_count + 1) % NUM_SAMPLE_TIMES;
27        last_sample = millis();
28    }
```

實作部分 - 使用BLE確認使用者是否靠近(續)

```
614 const char* getMeteoconIcon(uint16_t id, bool today)
615 {
616     if ( today && id/100 == 8 && (forecast->dt[0] < forecast->sunrise || forecast->dt[0] > forecast->sunset)) id += 1000;
617
618     if (id/100 == 2) return "thunderstorm";
619     if (id/100 == 3) return "drizzle";
620     if (id/100 == 4) return "unknown";
621     if (id == 500) return "lightRain";
622     else if (id == 511) return "sleet";
623     else if (id/100 == 5) return "rain";
624     if (id >= 611 && id <= 616) return "sleet";
625     else if (id/100 == 6) return "snow";
626     if (id/100 == 7) return "fog";
627     if (id == 800) return "clear-day";
628     if (id == 801) return "partly-cloudy-day";
629     if (id == 802) return "cloudy";
630     if (id == 803) return "cloudy";
631     if (id == 804) return "cloudy";
632     if (id == 1800) return "clear-night";
633     if (id == 1801) return "partly-cloudy-night";
634     if (id == 1802) return "cloudy";
635     if (id == 1803) return "cloudy";
636     if (id == 1804) return "cloudy";
637
638     return "unknown";
639 }
```

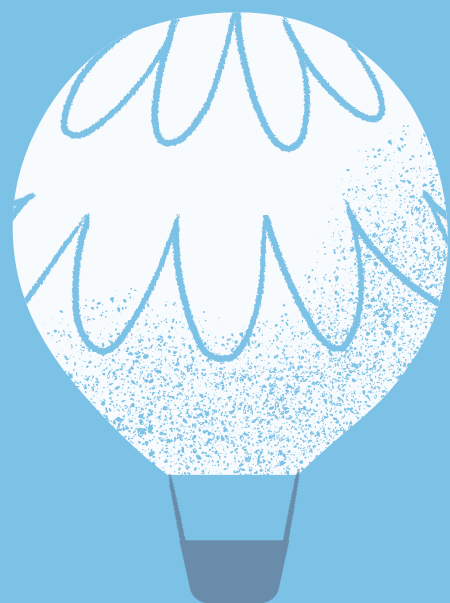
Demo



結論 & 未來展望

- 透過此裝置，可以有效避免使用者忘記帶傘出門
- 目前使用 **Arduino** 的 **BLEDevice.h** 來實作，該介面提供的掃描窗口大小只能設定成整數秒，導致幾秒的延遲，希望未來能使用較進階的開發環境





Thank You

