做一個小項目: 物聯網屏蔽

https://randomnerdtutorials.com/esp32-iot-shield-pcb-dashboard/

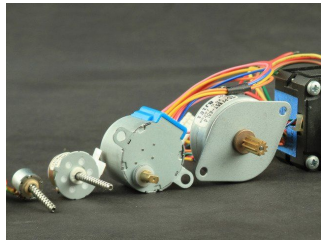
附加軟件庫: DHT Sensors, ESPUI, ESP32servo

什麼是物聯網屏蔽?

- 有輸出 - LED/蜂鳴器/

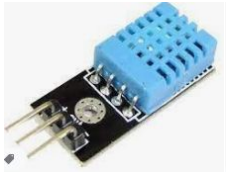
  直流馬達/伺服馬達/步進電機



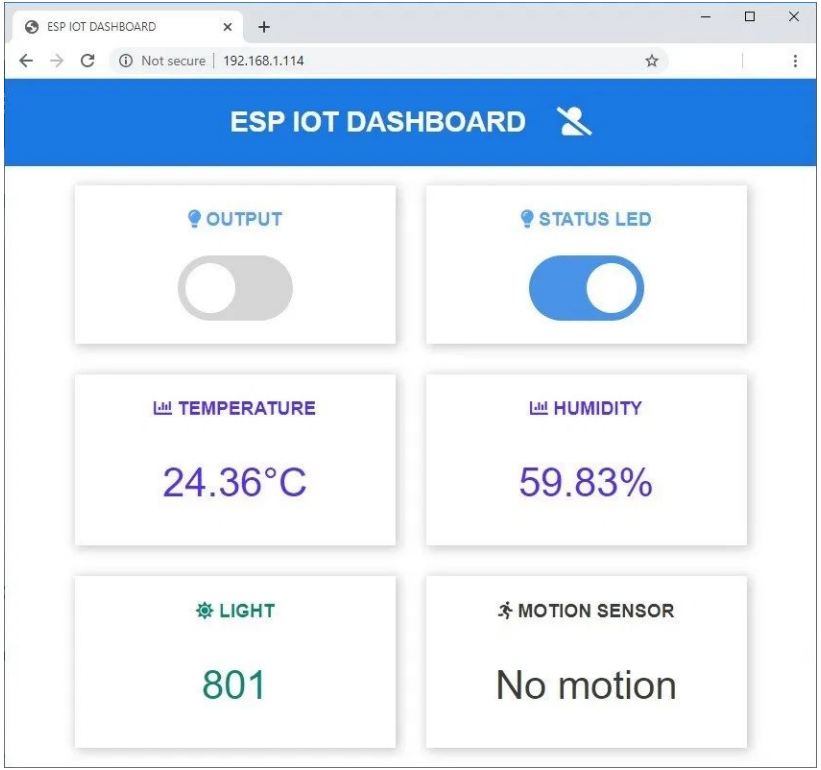
- 有輸入 - 按鈕/觸摸輸入


- 感應器 - 濕度,溫度計/光量計




什麼是物聯網屏蔽?

- 界面 - 網頁儀表板



物聯網屏蔽怎麼做?

- 使用Arduino開發環境為ESP32編寫軟件
  - 輸入/輸出 -
    - 輸入 - 按鈕/輕觸開關/直流馬達/伺服馬達/步進電機/無刷電機
    - 輸出 - LED/
  - 感應器控制
  - 網頁界面
- 電子設計
  - 什麼是電路原理圖
  - 印刷電路板製作
  - 電子設計自動化工具
  - 如何焊接

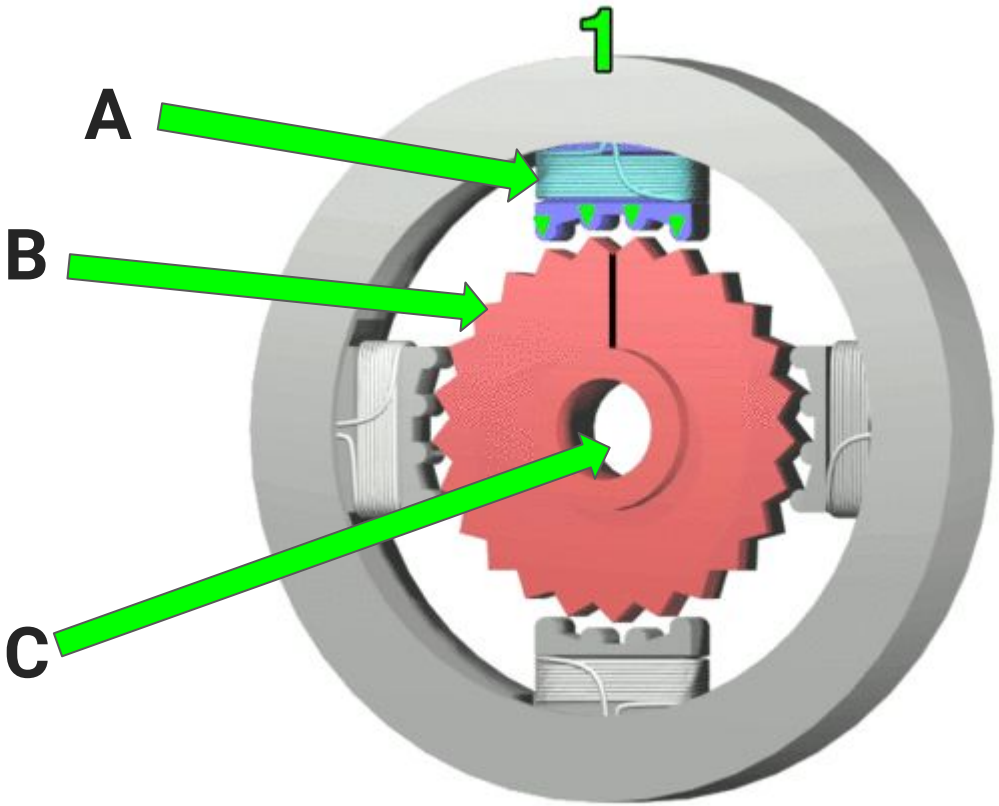
電機類型

- 直流馬達 □
- 伺服馬達 □
- 步進電機 □
- 無刷電機

步進電機

步進電機的構造:

A. 線圈
B. 轉子
C. 軸柄



步進電機

特性:

- 採用開迴路控制（Open-loop control);不需要運轉量感測器（sensor）或編碼器
- 藉由切換流向線圈中的電流，以一定角度逐步轉動
- 靜止時具有相當的保持力
- 每一步級的角度誤差小，而且沒有累積誤差
- 靜止時，步進馬達有很高的保持轉矩(Holding Torque)，可保持在 停止的位置，不需使用煞車迴路就不會自由轉動
- 旋轉角度0 到360 度

步進電機

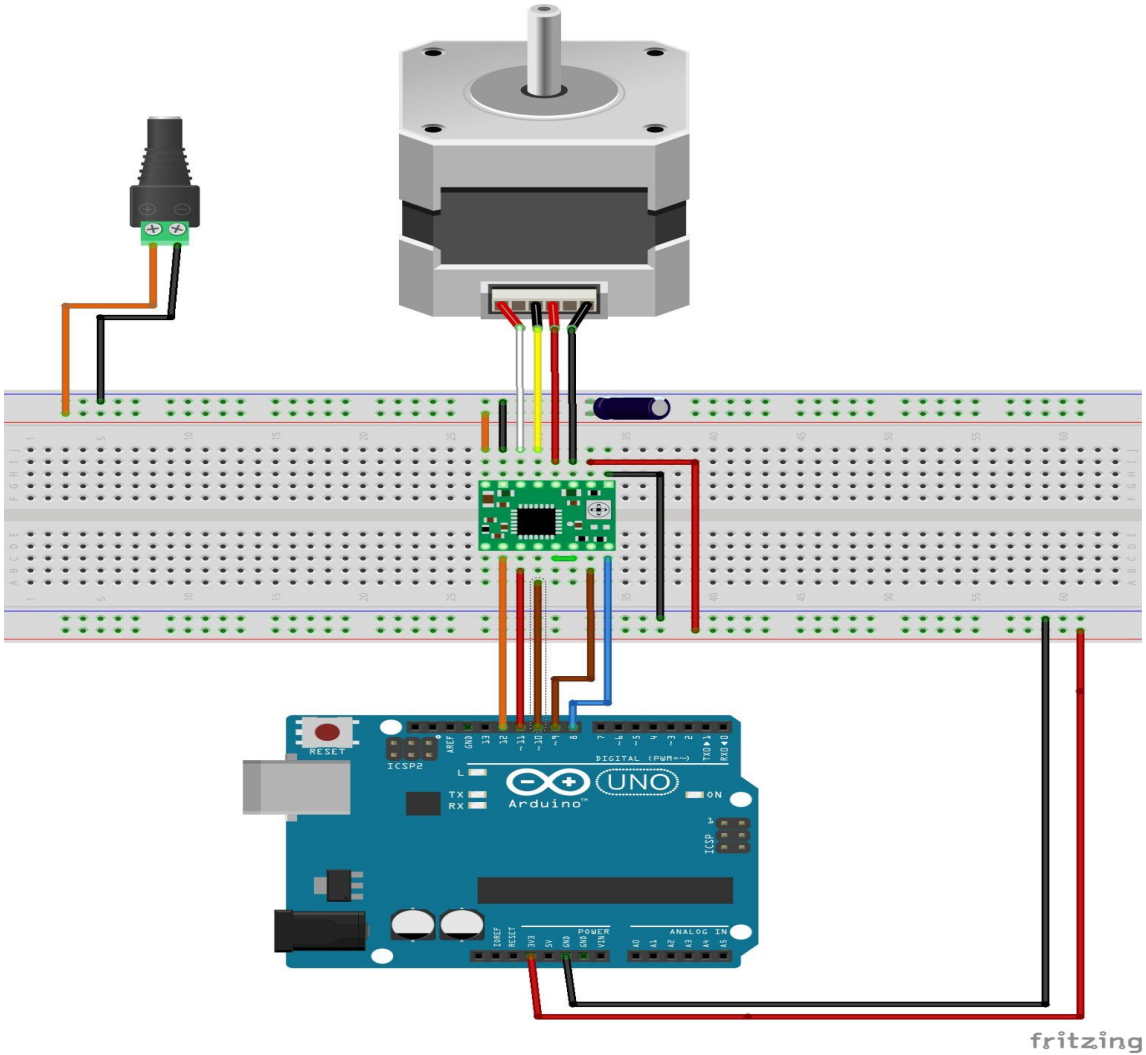
步進電機的工作原理

步進電機

步進電機驅動器, A4988:

- 工作電壓為 8V 至 35
- 每相(線圈)可提供高達約 1A 的電流
- 五種步進模式, 包括 全步、半步、1/4步、1/8步、1/16步(MS1-MS3)
- 只需要兩個引腳來驅動; DIR(方向,1或0)和STEP(步進,定寬脈衝)
- 睡眠功能(SLEEP 引腳);讓A4899進入低能耗睡眠状态

範例 步進電機


RESET
ICSP2
UNO
Arduino
DIGITAL (PWM~)
POWER
ANALOG IN
fritzing

程式碼解

```
#include <Arduino.h>

// Motor steps per revolution. Most steppers are 200 steps or 1.8 degrees/step
#define MOTOR_STEPS 20
#define RPM 30
int Step_size = 1;
int Max_Step = 100;
int   inbyte = 0 ;

#define DIR 12
#define STEP 13
#define SLEEP 13 // optional (just delete SLEEP from everywhere if not used)

#include "A4988.h"
#define MS1 19
#define MS2 23
#define MS3 5
A4988 stepper(MOTOR_STEPS, DIR, STEP, SLEEP, MS1, MS2, MS3);
```

程式碼解

```
void setup() {
    Serial.begin(115200);
    /*
     * Set target motor RPM.
     */
    stepper.begin(RPM); //設置步進器的速度
    // if using enable/disable on ENABLE pin (active LOW) instead of SLEEP uncomment next line
    stepper.setEnableActiveState(LOW);
    stepper.enable(); //啟用步進驅動程序

}
```

程式碼解

```
void menuRun() // 設置步進器步長的簡單菜單
{
  // ConOut.print("Choose Step Size");
  if (Serial.available())   inbyte = Serial.read(); //Serial input available
  switch (inbyte)
  {

    //====(Serial Menu)======
    case 'f': //Full Step
      Step_size=1;
      Serial.println(Max_Step*Step_size);
      break;

    case 'h': //Half Step
      Step_size=2;
      Serial.println(Max_Step*Step_size);
      break;

    case 'q': // 1/4 Step
      Step_size=4;
      Serial.println(Max_Step*Step_size);
      break;
```

程式碼解

```
case 'e': // 1/8 Step
      Step_size=8;
      Serial.println(Max_Step*Step_size);
      break;

    case 's': // 1/16 Step
      Step_size=16;
      Serial.println(Max_Step*Step_size);
      break;

  } //end switch (inbyte)

  inbyte = 0 ;

}
```

程式碼解

```
void loop() {

    menuRun();

    stepper.setMicrostep(Step_size);  // 設置步進電機的步長

    stepper.move(Max_Step*Step_size); // Forward step
    delay(500);
    stepper.move(-Max_Step*Step_size); // Backward step

}
```

ESP32 Wifi 設置

在 *void setup* 前:

```
const char *ssid = "ESPUI"; // 您要連接的 ssid
const char *apssid = “ESPUI_AP” // 設置為一個AP用的 SSID
IPAddress apIP(192, 168, 4, 1); // 設置為一個AP用的 IP 地址
const char *password = "espui"; // 你的wifi密碼
const char *hostname = "espui";
```

在 *void setup* 里:

```
void setup {
…… // 其他設置
WiFi.setHostname(hostname); // 設置主機名
WiFi.begin(ssid, password); // 使用您的 ssid 和密碼連接到您的 wifi
Serial.print("\n\nTry to connect to existing network");
```

ESP32 Wifi 設置:如果無法連接怎麼辦?

如果無法連接怎麼辦? 如果無法連接超過5秒, 退卻設置為一個AP:

```
{
    uint8_t timeout = 10;
    // Wait for connection, 5s timeout //如果無法連接超過5秒
    do {
      delay(500);
      Serial.print(".");
      timeout--;
    } while (timeout && WiFi.status() != WL_CONNECTED);
```

ESP32 Wifi 設置

```
// not connected -> create hotspot 退卻設置為一個AP
  if (WiFi.status() != WL_CONNECTED) {
    Serial.print("\n\nCreating hotspot");
    WiFi.mode(WIFI_AP); // 設置為一個AP
    WiFi.softAPConfig(apIP, apIP, IPAddress(255, 255, 255, 0)); // 設置AP的IP 地址
    WiFi.softAP(apssid); // 設置AP的SSID
    timeout = 5;
    do {
      delay(500);
      Serial.print(".");
      timeout--;
    } while (timeout);
  }
}
```

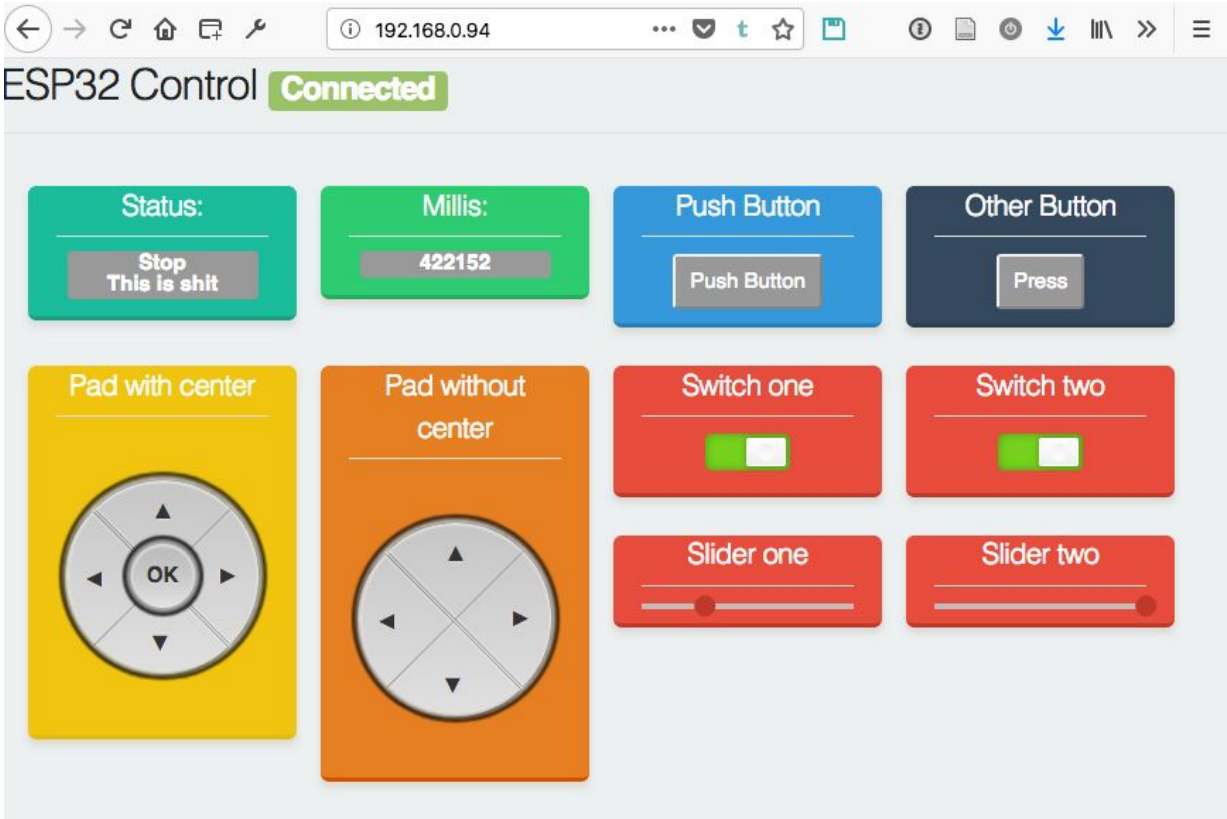
ESP32 Wifi 設置: 在串口上顯示wifi設置

```
dnsServer.start(DNS_PORT, "*", apIP);


  Serial.println("\n\nWiFi parameters:");
  Serial.print("Mode: ");
  Serial.println(WiFi.getMode() == WIFI_AP ? "Station" : "Client");
  Serial.print("IP address: ");
  Serial.println(WiFi.getMode() == WIFI_AP ? WiFi.softAPIP() : WiFi.localIP());
```

網頁界面

ESPUI網址



網頁界面

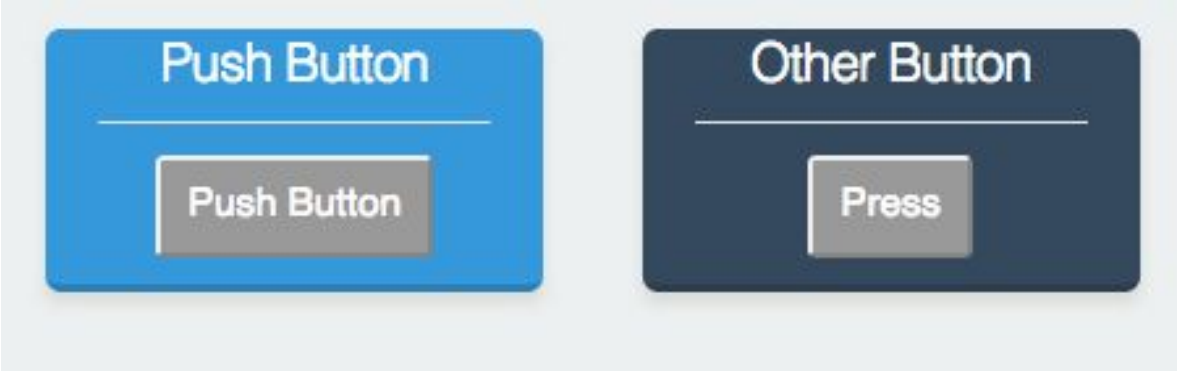
用戶界面元素:

- 標籤(Label)
- 按鈕(Button)
- 開關(Switch)
- 控制面板(Control Pad)
- 帶中心按鈕的控制板(Control Pad with Centre)
- 滑塊(Slider)
- 文本輸入(Text Input)
- 數字輸入(Number Input)
- 圖表(Graph)
- 選項選擇(Option select)

網頁界面

- 標籤(Label)

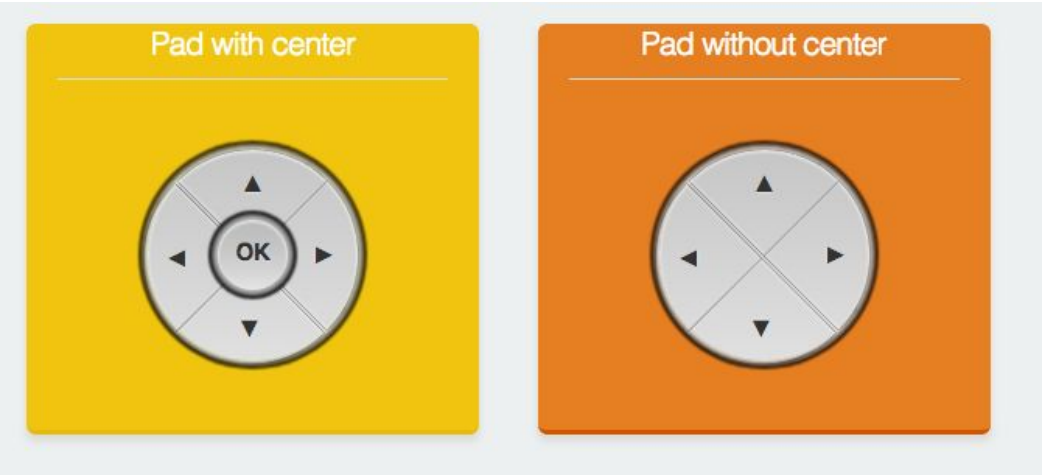
- 按鈕(Button)



網頁界面

- 開關(Switch)

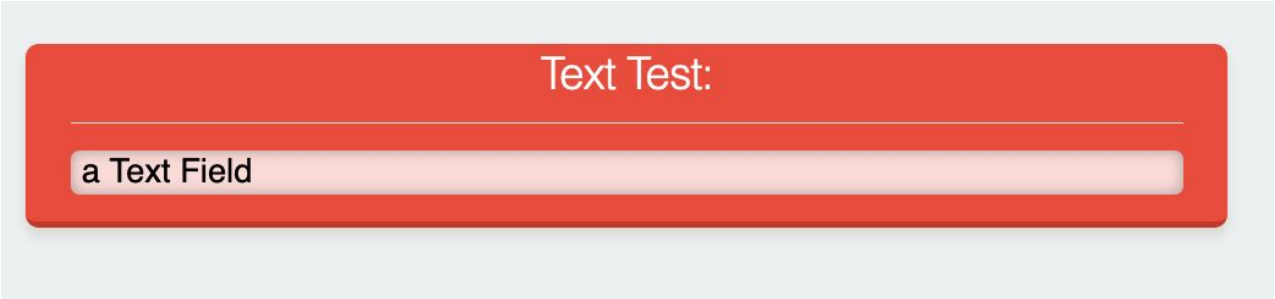
- 帶中心按鈕的控制板(Control Pad with Centre)/控制面板(Control Pad)



網頁界面

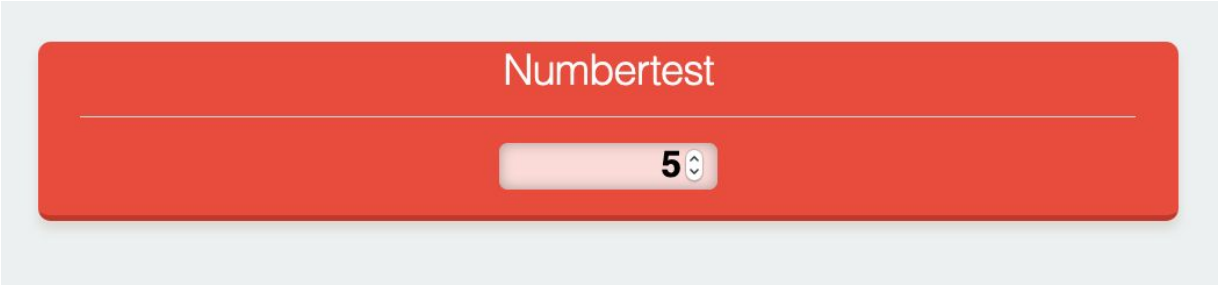
- 滑塊(Slider)

- 文本輸入(Text Input)

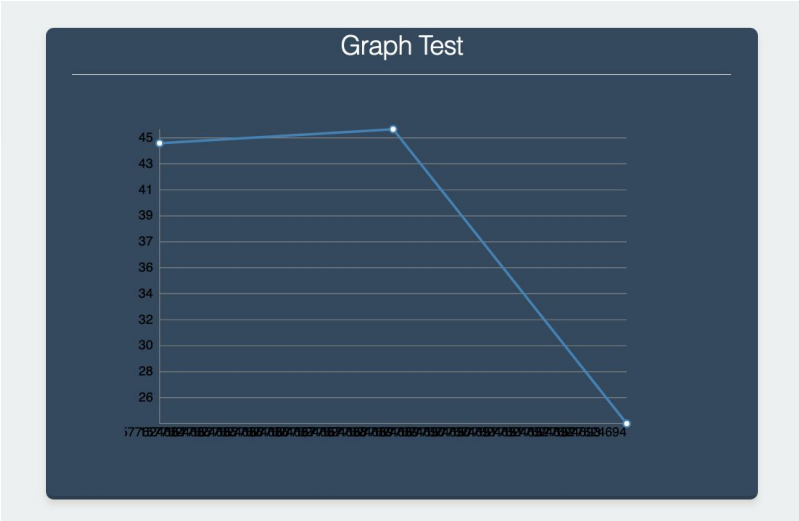


網頁界面

- 數字輸入(Number Input)

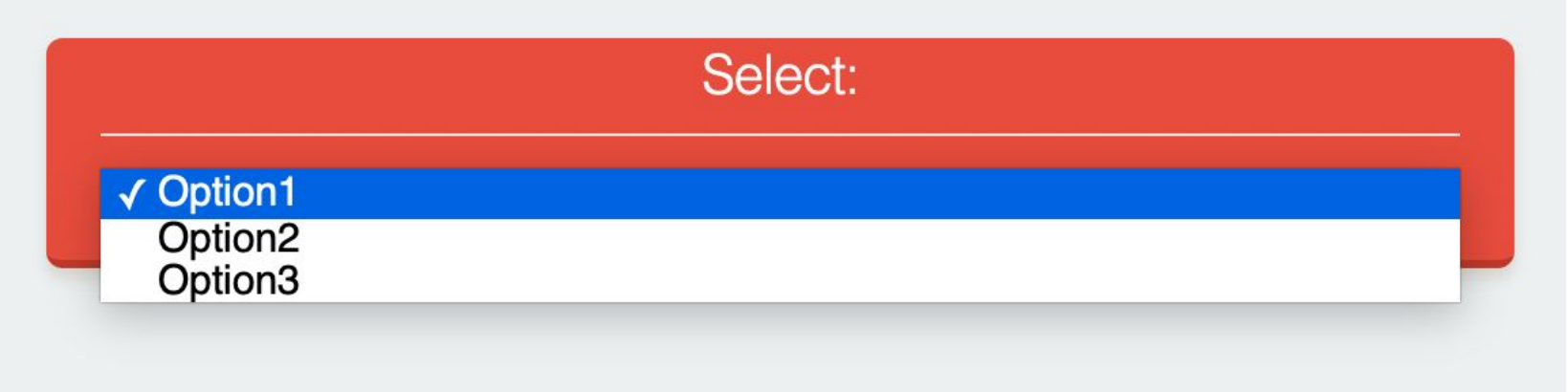


- 圖表(Graph)



網頁界面

- 選項選擇(Option select)



程式碼解 - 簡單按鈕

```
#include <ESPUI.h>
void buttonCallback(Control *sender, int type) {                // 按鈕程序
  switch (type) {
  case B_DOWN: // Button Event
    Serial.println("Button DOWN"); //Button Action
    break;
 case B_UP: // Button Event
    Serial.println("Button UP"); //Button Action
    break;
  }
}
```

程式碼解 - 簡單按鈕

```
void setup(void) {
  Serial.begin(115200);   //Set 串行端口 至 115200 baud)
<Setup WiFi>
  ESPUI.setVerbosity(Verbosity::Verbose);   //ESPUI 將向串口報告輸入狀態
  ESPUI.button("Push Button", &buttonCallback, ControlColor::Peterriver, "Push Button");
  ESPUI.begin("ESP32 Control"); // 開始 ESPUI
  }
```

按鈕標籤

按鈕程序

按鈕顏色

按鈕上標籤

網頁界面標籤

程式碼解 - 簡單滑塊

```
#include <ESPUI.h>
void slider(Control *sender, int type) {
   // Like all Control Values in ESPUI slider values are Strings. To use them as int simply do this:
  int sliderValueWithOffset = sender->value.toInt() + 100; // 滑塊值 sliderValueWithOffset 0-100
}
In void setup {
…..
ESPUI.slider("Slider one", &slider, ControlColor::Alizarin, 30);
```

其他按鈕事件

- 按鈕(Button)
  - B_DOWN
  - B_UP
- 開關(Switch)
  - S_ACTIVE
  - S_INACTIVE

其他按鈕事件

- 控制面板(Control Pad)/帶中心按鈕的控制板(Control Pad with Centre)
  - P_LEFT_DOWN
  - P_LEFT_UP
  - P_RIGHT_DOWN
  - P_RIGHT_UP
  - P_FOR_DOWN
  - P_FOR_UP
  - P_BACK_DOWN
  - P_BACK_UP
  - P_CENTER_DOWN
  - P_CENTER_UP