//可以只看loop就好

#include <TridentTD\_LineNotify.h>

#include <dht11.h>

// ArduTalk

#define DefaultIoTtalkServerIP "140.113.199.200"

#define DM\_name "NodeMCU"

#define DF\_list {"D0~","D1~","D2~","D5","D6","D7","D8","A0"}

#define nODF 10 // The max number of ODFs which the DA can pull.

#include <ESP8266WiFi.h>

//#include <WiFiClient.h>

#include <ESP8266WebServer.h>

#include <ESP8266WiFiMulti.h>

#include "ESP8266HTTPClient2.h"

#include <EEPROM.h>

char IoTtalkServerIP[100] = "";

String result;

String url = "";

String passwordkey ="";

HTTPClient http;

String remove\_ws(const String& str )

{

String str\_no\_ws ;

for( char c : str ) if( !std::isspace(c) ) str\_no\_ws += c ;

return str\_no\_ws ;

}

void clr\_eeprom(int sw=0){

if (!sw){

Serial.println("Count down 3 seconds to clear EEPROM.");

digitalWrite(2,LOW);

delay(3000);

}

if( (digitalRead(0) == LOW) || (sw == 1) ){

for(int addr=0; addr<50; addr++) EEPROM.write(addr,0); // clear eeprom

EEPROM.commit();

Serial.println("Clear EEPROM and reboot.");

digitalWrite(2,HIGH);

ESP.reset();

}

}

void save\_netInfo(char \*wifiSSID, char \*wifiPASS, char \*ServerIP){ //stoage format: [SSID,PASS,ServerIP]

char \*netInfo[3] = {wifiSSID, wifiPASS, ServerIP};

int addr=0,i=0,j=0;

EEPROM.write (addr++,'['); // the code is equal to (EEPROM.write (addr,'['); addr=addr+1;)

for (j=0;j<3;j++){

i=0;

while(netInfo[j][i] != '\0') EEPROM.write(addr++,netInfo[j][i++]);

if(j<2) EEPROM.write(addr++,',');

}

EEPROM.write (addr++,']');

EEPROM.commit();

}

int read\_netInfo(char \*wifiSSID, char \*wifiPASS, char \*ServerIP){ // storage format: [SSID,PASS,ServerIP]

char \*netInfo[3] = {wifiSSID, wifiPASS, ServerIP};

String readdata="";

int addr=0;

char temp = EEPROM.read(addr++);

if(temp == '['){

for (int i=0; i<3; i++){

readdata ="";

while(1){

temp = EEPROM.read(addr++);

if (temp == ',' || temp == ']') break;

readdata += temp;

}

readdata.toCharArray(netInfo[i],100);

}

if (String(ServerIP).length () < 7){

Serial.println("ServerIP loading failed.");

return 2;

}

else{

Serial.println("Load setting successfully.");

return 0;

}

}

else{

Serial.println("no data in eeprom");

return 1;

}

}

String scan\_network(void){

int AP\_N,i; //AP\_N: AP number

String AP\_List="<select name=\"SSID\" style=\"width: 150px; font-size:16px; color:blue; \" required>" ;// make ap\_name in a string

AP\_List += "<option value=\"\" disabled selected>Select AP</option>";

WiFi.disconnect();

delay(100);

AP\_N = WiFi.scanNetworks();

if(AP\_N>0) for (i=0;i<AP\_N;i++) AP\_List += "<option value=\""+WiFi.SSID(i)+"\">" + WiFi.SSID(i) + "</option>";

else AP\_List = "<option value=\"\">NO AP</option>";

AP\_List +="</select><br><br>";

return(AP\_List);

}

ESP8266WebServer server ( 80 );

void handleRoot(int retry){

String temp = "<html><title>Wi-Fi Setting</title>";

temp += "<head><meta http-equiv=\"Content-Type\" content=\"text/html; charset=utf-8\"/>";

temp += "<meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0\"></head><body bgcolor=\"#F2F2F2\">";

if (retry) temp += "<font color=\"#FF0000\">Please fill all fields.</font>";

temp += "<form action=\"setup\"><div>";

temp += "<center>SSID:<br>";

temp += scan\_network();

temp += "Password:<br>";

temp += "<input type=\"password\" name=\"Password\" vplaceholder=\"Password\" style=\"width: 150px; font-size:16px; color:blue; \">";

temp += "<br><br>IoTtalk Server IP <br>";

temp += "<input type=\"serverIP\" name=\"serverIP\" value=\"";

temp += DefaultIoTtalkServerIP;

temp += "\" style=\"width: 150px; font-size:16px; color:blue;\" required>";

temp += "<br><br><input style=\"-webkit-border-radius: 11; -moz-border-radius: 11";

temp += "border-radius: 0px;";

temp += "text-shadow: 1px 1px 3px #666666;";

temp += "font-family: Arial;";

temp += "color: #ffffff;";

temp += "font-size: 18px;";

temp += "background: #AAAAAA;";

temp += "padding: 10px 20px 7px 20px;";

temp += "text-decoration: none;\"" ;

temp += "type=\"submit\" value=\"Submit\" on\_click=\"javascript:alert('TEST');\"></center>";

temp += "</div></form><br>";

temp += "</body></html>";

server.send ( 200, "text/html", temp );

}

void handleNotFound() {

server.send( 404, "text/html", "Page not found.");

}

void saveInfoAndConnectToWiFi() {

Serial.println("Get network information.");

char \_SSID\_[100]="";

char \_PASS\_[100]="";

if (server.arg(0) != "" && server.arg(2) != ""){//arg[0]-> SSID, arg[1]-> password (both string)

server.arg(0).toCharArray(\_SSID\_,100);

server.arg(1).toCharArray(\_PASS\_,100);

server.arg(2).toCharArray(IoTtalkServerIP,100);

server.send(200, "text/html", "<html><body><center><span style=\" font-size:72px; color:blue; margin:100px; \"> Setup successfully. </span></center></body></html>");

server.stop();

save\_netInfo(\_SSID\_, \_PASS\_, IoTtalkServerIP);

connect\_to\_wifi(\_SSID\_, \_PASS\_);

}

else {

handleRoot(1);

}

}

void start\_web\_server(void){

server.on ( "/", [](){handleRoot(0);} );

server.on ( "/setup", saveInfoAndConnectToWiFi);

server.onNotFound ( handleNotFound );

server.begin();

}

void wifi\_setting(void){

String softapname = "MCU-";

uint8\_t MAC\_array[6];

WiFi.macAddress(MAC\_array);

for (int i=0;i<6;i++){

if( MAC\_array[i]<0x10 ) softapname+="0";

softapname+= String(MAC\_array[i],HEX); //Append the mac address to url string

}

IPAddress ip(192,168,0,1);

IPAddress gateway(192,168,0,1);

IPAddress subnet(255,255,255,0);

WiFi.mode(WIFI\_AP\_STA);

WiFi.disconnect();

WiFi.softAPConfig(ip,gateway,subnet);

WiFi.softAP(&softapname[0]);

start\_web\_server();

Serial.println ( "Switch to AP mode and start web server." );

}

uint8\_t wifimode = 1; //1:AP , 0: STA

void connect\_to\_wifi(char \*wifiSSID, char \*wifiPASS){

long connecttimeout = millis();

WiFi.softAPdisconnect(true);

Serial.println("-----Connect to Wi-Fi-----");

WiFi.begin(wifiSSID, wifiPASS);

while (WiFi.status() != WL\_CONNECTED && (millis() - connecttimeout < 10000) ) {

delay(1000);

Serial.print(".");

}

if(WiFi.status() == WL\_CONNECTED){

Serial.println ( "Connected!\n");

digitalWrite(2,LOW);

wifimode = 0;

}

else if (millis() - connecttimeout > 10000){

Serial.println("Connect fail");

wifi\_setting();

}

}

int iottalk\_register(void){

url = "http://" + String(IoTtalkServerIP) + ":9999/";

String df\_list[] = DF\_list;

int n\_of\_DF = sizeof(df\_list)/sizeof(df\_list[0]); // the number of DFs in the DF\_list

String DFlist = "";

for (int i=0; i<n\_of\_DF; i++){

DFlist += "\"" + df\_list[i] + "\"";

if (i<n\_of\_DF-1) DFlist += ",";

}

uint8\_t MAC\_array[6];

WiFi.macAddress(MAC\_array);//get esp12f mac address

for (int i=0;i<6;i++){

if( MAC\_array[i]<0x10 ) url+="0";

url+= String(MAC\_array[i],HEX); //Append the mac address to url string

}

//send the register packet

Serial.println("[HTTP] POST..." + url);

String profile="{\"profile\": {\"d\_name\": \"";

//profile += "MCU.";

for (int i=3;i<6;i++){

if( MAC\_array[i]<0x10 ) profile+="0";

profile += String(MAC\_array[i],HEX);

}

profile += "\", \"dm\_name\": \"";

profile += DM\_name;

profile += "\", \"is\_sim\": false, \"df\_list\": [";

profile += DFlist;

profile += "]}}";

http.begin(url);

http.addHeader("Content-Type","application/json");

int httpCode = http.POST(profile);

Serial.println("[HTTP] Register... code: " + (String)httpCode );

Serial.println(http.getString());

//http.end();

url +="/";

return httpCode;

}

String df\_name\_list[nODF];

String df\_timestamp[nODF];

void init\_ODFtimestamp(){

for (int i=0; i<nODF; i++) df\_timestamp[i] = "";

for (int i=0; i<nODF; i++) df\_name\_list[i] = "";

}

int DFindex(char \*df\_name){

for (int i=0; i<nODF; i++){

if (String(df\_name) == df\_name\_list[i]) return i;

else if (df\_name\_list[i] == ""){

df\_name\_list[i] = String(df\_name);

return i;

}

}

return nODF+1; // df\_timestamp is full

}

int push(char \*df\_name, String value){

http.begin( url + String(df\_name));

http.addHeader("Content-Type","application/json");

String data = "{\"data\":[" + value + "]}";

//Serial.println("data: "+data);

int httpCode = http.PUT(data);

if (httpCode != 200) Serial.println("[HTTP] PUSH \"" + String(df\_name) + "\"... code: " + (String)httpCode + ", retry to register.");

while (httpCode != 200){

digitalWrite(4, LOW);

digitalWrite(2, HIGH);

httpCode = iottalk\_register();

if (httpCode == 200){

http.PUT(data);

// if (switchState) digitalWrite(4,HIGH);

}

else delay(3000);

}

http.end();

return httpCode;

}

String pull(char \*df\_name){

http.begin( url + String(df\_name) );

Serial.println(url + String(df\_name));

http.addHeader("Content-Type","application/json");

int httpCode = http.GET(); //http state code

if (httpCode != 200) Serial.println("[HTTP] "+url + String(df\_name)+" PULL \"" + String(df\_name) + "\"... code: " + (String)httpCode + ", retry to register.");

while (httpCode != 200){

digitalWrite(4, LOW);

digitalWrite(2, HIGH);

httpCode = iottalk\_register();

if (httpCode == 200){

http.GET();

//if (switchState) digitalWrite(4,HIGH);

}

else delay(3000);

}

String get\_ret\_str = http.getString(); //After send GET request , store the return string

// Serial.println

Serial.println("output "+String(df\_name)+": \n"+get\_ret\_str);

http.end();

get\_ret\_str = remove\_ws(get\_ret\_str);

int string\_index = 0;

string\_index = get\_ret\_str.indexOf("[",string\_index);

String portion = ""; //This portion is used to fetch the timestamp.

if (get\_ret\_str[string\_index+1] == '[' && get\_ret\_str[string\_index+2] == '\"'){

string\_index += 3;

while (get\_ret\_str[string\_index] != '\"'){

portion += get\_ret\_str[string\_index];

string\_index+=1;

}

if (df\_timestamp[DFindex(df\_name)] != portion){

df\_timestamp[DFindex(df\_name)] = portion;

string\_index = get\_ret\_str.indexOf("[",string\_index);

string\_index += 1;

portion = ""; //This portion is used to fetch the data.

while (get\_ret\_str[string\_index] != ']'){

portion += get\_ret\_str[string\_index];

string\_index+=1;

}

return portion; // return the data.

}

else return "\_\_\_NULL\_DATA\_\_\_";

}

else return "\_\_\_NULL\_DATA\_\_\_";

}

long sensorValue, suspend = 0;

long cycleTimestamp = millis();

void setup() {

pinMode(2, OUTPUT);// D4 : on board led

digitalWrite(2,HIGH);

pinMode(0, INPUT\_PULLUP); // D3, GPIO0: clear eeprom button

pinMode(16, OUTPUT);// D0~

pinMode(5, OUTPUT); // D1~

pinMode(4, OUTPUT); // D2~

pinMode(14, OUTPUT);// D5

pinMode(12, OUTPUT);// D6

pinMode(13, INPUT);// D7

pinMode(15, OUTPUT);// D8

EEPROM.begin(512);

Serial.begin(115200);

char wifissid[100]="";

char wifipass[100]="";

int statesCode = read\_netInfo(wifissid, wifipass, IoTtalkServerIP);

//for (int k=0; k<50; k++) Serial.printf("%c", EEPROM.read(k) ); //inspect EEPROM data for the debug purpose.

if (!statesCode) connect\_to\_wifi(wifissid, wifipass);

else{

Serial.println("Laod setting failed! statesCode: " + String(statesCode)); // StatesCode 1=No data, 2=ServerIP with wrong format

wifi\_setting();

}

while(wifimode){

server.handleClient(); //waitting for connecting to AP ;

delay(10);

}

statesCode = 0;

while (statesCode != 200) {

statesCode = iottalk\_register();

if (statesCode != 200){

Serial.println("Retry to register to the IoTtalk server. Suspend 3 seconds.");

if (digitalRead(0) == LOW) clr\_eeprom();

delay(3000);

}

}

init\_ODFtimestamp();

//digitalWrite(16,LOW);

digitalWrite(5,LOW);

digitalWrite(4,LOW);

digitalWrite(14,LOW);

digitalWrite(12,LOW);

digitalWrite(13,LOW);

digitalWrite(15,LOW);

}

dht11 DhtSensor;

int pinA0;

long LEDflashCycle = millis();

long LEDonCycle = millis();

int LEDhadFlashed = 0;

int t,h;

//int frequency = 1000;

//int ton = 500,toff=500;

int count = 0;

//String temp,humid,soil;

//long i=0;

void loop() {

if (digitalRead(0) == LOW) clr\_eeprom();

if (millis() - cycleTimestamp > 200) {

pinA0 = analogRead(A0);

DhtSensor.read(D7);

int t = DhtSensor.temperature , h = DhtSensor.humidity;

Serial.println(String(t)+"/"+String(h));

Serial.println("Soil\_moisture: "+String(pinA0));

if(pinA0>800){

if(count==0){

digitalWrite(D8,HIGH);//繼電器

digitalWrite(D1,HIGH);delay(200);//RED

digitalWrite(D2,HIGH);delay(200);//GREEN

digitalWrite(D6,HIGH);delay(300);//BLUE

digitalWrite(D8,LOW);

}count=count+1;

}else{

digitalWrite(D1,LOW);

digitalWrite(D2,LOW);

digitalWrite(D6,LOW);

count = 0;

}

//pinA0 = map(pinA0,0,1023,100,0);

/\*tone(D0, frequency);

delay(ton);

noTone(D0);

delay(toff);\*/

push("A0", String(t)+String(h)+String(pinA0));

delay(2000);

/\*result = pull("D0~");

if (result != "\_\_\_NULL\_DATA\_\_\_"){

Serial.println ("D0~: "+result);

if (result.toInt() >= 0 && result.toInt() <= 255) analogWrite(16, result.toInt());

}\*/

result = pull("D1~");

if (result != "\_\_\_NULL\_DATA\_\_\_"){

Serial.println ("D1~: "+result);

if (result.toInt() >= 0 && result.toInt() <= 255) analogWrite(5, result.toInt());

}

result = pull("D2~");

if (result != "\_\_\_NULL\_DATA\_\_\_"){

Serial.println ("D2~: "+result);

if (result.toInt() >= 0 && result.toInt() <= 255) analogWrite(4, result.toInt());

}

/\*result = pull("D5");

if (result != "\_\_\_NULL\_DATA\_\_\_"){

Serial.println ("D5: "+result);

if (result.toInt() > 0 ) digitalWrite(14, 1);

else digitalWrite(14, 0);

}\*/

result = pull("D6");

if (result != "\_\_\_NULL\_DATA\_\_\_"){

Serial.println ("D6: "+result);

if (result.toInt() > 0 ) digitalWrite(12, 1);

else digitalWrite(12, 0);

}

/\*result = pull("D7");

if (result != "\_\_\_NULL\_DATA\_\_\_"){

Serial.println ("D7: "+result);

if (result.toInt() > 0 ) digitalWrite(13, 1);

else digitalWrite(13, 0);

}\*/

result = pull("D8");

if (result != "\_\_\_NULL\_DATA\_\_\_"){

Serial.println ("D8: "+result);

if (result.toInt() > 0 ) digitalWrite(15, 1);

else digitalWrite(15, 0);

}

cycleTimestamp = millis();

}

if (millis() - LEDflashCycle > 2000){

LEDhadFlashed = 0;

LEDflashCycle = millis();

}

if (!LEDhadFlashed){

digitalWrite(2, 0);

LEDhadFlashed = 1;

LEDonCycle = millis();

}

if (millis()-LEDonCycle > 1) digitalWrite(2, 1);

}