

1.본인이 작성한 코드

주요 세팅

#include를 사용하여 ESP8266과 LCD 모니터 관련 라이브러리를 불러왔습니다. 이를 통해 ESP8266 보드와 LCD를 함께 사용할 수 있도록 설정했습니다.

API에 연결하기 위해 SSID(와이파이 이름), Password(비밀번호), API Key, Server 주소, HTTP Port를 설정하였습니다.

ESP8266WebServer server 객체는 웹서버 기능을 위해 추가로 구성하였습니다.(추가)
WiFiClient client 객체는 API 요청을 보내기 위해 사용됩니다.

문자열 인코딩 관련

station 값이 "한대앞"처럼 한글로 되어 있을 경우, LCD나 시리얼 모니터에서 문자 깨짐 현상이 발생합니다.

이를 방지하기 위해 문자열을 URL 인코딩된 값인 %ED%95%9C%EB%8C%80%EC%95%9E 형태로 변환하여 출력하도록 했습니다.

```
1  #include <ESP8266WiFi.h>
2  #include<LiquidCrystal_I2C.h>
3      #include <ESP8266WebServer.h>
4
5  #include <Wire.h>
6  LiquidCrystal_I2C lcd(0x27,16,2); //LCD 클래스 초기화
7
8
9  const char* ssid = "HY-DORM3-403";
10 const char* password = "residence403";
11
12 const char* api_key = "4b6b796843746f6f3639475a64796e";
13 const char* SERVER = "swopenapi.seoul.go.kr";
14 const int httpPort = 80;
15
16 ESP8266WebServer server(80);
17 WiFiClient client;
18 #define BUTTON_UP 4
19 #define BUTTON_DOWN 5
20
21 boolean last_Button = HIGH;
22 boolean c_Button = HIGH;
23
24 boolean last_Button2 = HIGH;
25 boolean c_Button2 = HIGH;
26 bool requestDone1 = false; // type1 버튼 요청 완료 플래그
27 bool requestDone2 = false; // type2 버튼 요청 완료 플래그
28
29 String station = "%ED%95%9C%EB%8C%80%EC%95%9E"; // UTF-8 한글 인코딩 주의
30
31 String payload = "";
32 String trainName,sttsu,msg2,msg3 = "";
33 String filteredPayload = "";
34
```

setup

셋업에서는 시리얼모니터,lcd모니터wifi연결확인을 하였습니다.

server.on 부터는 추가아이디어에 있는 어플과 상호작용하기위해 만든 코드입니다.

type1을 하면 하행 type2를 하면 상행 아무것도 없으면 웹에 fiterpayload인 string 줄을 웹에 보냅니다.(추가)

```
1 void setup() {
2   Serial.begin(115200);
3   delay(100);
4   lcd.init();
5   lcd.backlight();
6
7   pinMode(BUTTON_UP, INPUT_PULLUP);
8   pinMode(BUTTON_DOWN, INPUT_PULLUP);
9
10  Serial.println();
11  Serial.print("Connecting to ");
12  Serial.println(ssid);
13
14  WiFi.begin(ssid, password);
15  while (WiFi.status() != WL_CONNECTED) {
16    delay(500);
17    Serial.print(".");
18  }
19  Serial.println("");
20  Serial.println("WiFi connected");
21  Serial.println("IP address: ");
22  Serial.println(WiFi.localIP());
23  lcd.print("push Button");
24  // /type1 경로 처리
25  server.on("/type1", []() {
26    c_Button= LOW;
27    server.send(200, "text/plain; charset=UTF-8", "하행 요청 수신됨");
28  });
29
30  // /type2 경로 처리
31  server.on("/type2", []() {
32    c_Button2= LOW;
33    server.send(200, "text/plain; charset=UTF-8", "상행 요청 수신됨");
34  });
35  server.on("/", []() {
36    server.send(200, "text/plain; charset=UTF-8", filteredPayload);
37  });
38  server.begin();
39  Serial.println("HTTP server started");
40 }
```

loop

버튼의 상태를 받고 c_button이 low이면 하행 c_button2이 low이면 상행 열차를 불러올수 있게 만들었습니다.

추가로 웹서버에서 핸들링을 할 때 type1 또는 type2가 계속 켜져 있으면 버튼은 계속 low버튼이 되기 때문에 requestDone1을 이용해서 중복을 방지하였습니다.(추가)

```
1 void loop() {
2
3
4   c_Button = digitalRead(BUTTON_DOWN);
5   c_Button2 = digitalRead(BUTTON_UP);
6   server.handleClient();
7
8   // type1 버튼 눌림 감지 (HIGH->LOW) & 중복 방지
9   if (last_Button == HIGH && c_Button == LOW && !requestDone1) {
10      makeRequestAndUpdateMessage("<updnLine>하행</updnLine>");
11      requestDone1 = true;
12   }
13   // 버튼 떼어짐 감지 (LOW->HIGH) 시 플래그 초기화
14   if (last_Button == LOW && c_Button == HIGH) {
15      requestDone1 = false;
16   }
17
18   // type2 버튼 눌림 감지 & 중복 방지
19   if (last_Button2 == HIGH && c_Button2 == LOW && !requestDone2) {
20      makeRequestAndUpdateMessage("<updnLine>상행</updnLine>");
21      requestDone2 = true;
22   }
23   // 버튼 떼어짐 시 플래그 초기화
24   if (last_Button2 == LOW && c_Button2 == HIGH) {
25      requestDone2 = false;
26   }
27
28   last_Button = c_Button;
29   last_Button2 = c_Button2;
30
31   delay(10);
32 }
```

함수

이 함수는 line 정보를 받아 상행인지 하행인지 먼저 판별합니다.

그 후 section에서 trainName, msg2, msg3 등의 정보를 추출하여 시리얼 모니터에 출력합니다.

초기에는 4호선과 신분당선이 동시에 출력되는 문제가 있었기 때문에, 추가 조건으로 subway 값을 확인하여, 값이 1004(4호선)가 아니면 출력하지 않도록 필터링을 추가했습니다.

LCD 모니터에서는 "몇 정거장 남았는지"를 표시하는 것이 주 목적이기 때문에, msg2 값을 중심으로 사용하였습니다.

msg2는 한글 문자열로 되어 있기 때문에, 이를 기반으로 남은 정거장 수를 추출하여 숫자로 변환해 사용하였습니다.

특히 msg2에 따라 다음과 같이 숫자를 대응시켜 표기하였습니다

"한대앞" ==0 , "전역" ==1

예를 들어, LCD 화면에는 "0 > 1 > 3"처럼 표시되며,

이는 번역하면 "한 대앞 진입 > 전역 진입 > 3정거장 전 진입"을 의미합니다.

또한, 어플리케이션과 연동하기 위해 진입한 역 정보는 result 변수에 저장해 두었습니다.

모든 출력이 완료되면 delay(10000);을 실행하여 10초간 대기 후,

LCD에는 "wait"라는 문구가 표시되도록 구현했습니다.

```
1 void makeRequestAndUpdateMessage(String line) {
2   String url = "/api/subway/";
3   url += api_key;
4   url += "/xml/realtimeStationArrival/1/5/";
5   url += station;
6
7   Serial.print("Requesting URL: ");
8   Serial.println(url);
9
10  if (!client.connect(SERVER, httpPort)) {
11    Serial.println("Connection failed");
12    return;
13  }
14
15  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
16               "Host: " + SERVER + "\r\n" +
17               "Connection: close\r\n\r\n");
18
19  // 서버 응답 읽기
20  String payload = "";
21  while (client.connected()) {
22    while (client.available()) {
23      payload = client.readStringUntil('\n');
24    }
25  }
26
27  int index = 0;
28  String result = "";
29  lcd.clear();
30
31  while (true) {
32    String section = extractRow(payload, index);
33    if (section == "") break;
34
35    if (section.indexOf(line) >= 0) {
36      String subwayId = getNumber(section, "<subwayId>", "</subwayId>");
37      if (subwayId != "1004") {
38        index++;
39        continue;
40      }
41
42      String trainName = getNumber(section, "<trainLineNm>", "</trainLineNm>");
43      String msg2 = getNumber(section, "<arv1Msg2>", "</arv1Msg2>");
44      String msg3 = getNumber(section, "<arv1Msg3>", "</arv1Msg3>");
45
46      Serial.println(subwayId);
47      Serial.println(trainName);
48      Serial.println(msg2);
49      Serial.println(msg3);
50    }
51  }
```

```

50
51     if (msg2.length() <= 1) {
52         lcd.print("not train");
53         break;
54     }
55     result += subwayId + " " + trainName + "\n" + msg2 + "\n";
56
57     if (strncmp(msg2.c_str(), "전", 3) == 0) {
58         msg2.setCharAt(1, '1');
59     } else if (strncmp(msg2.c_str(), "한", 3) == 0) {
60         msg2.setCharAt(1, '0');
61     }
62     lcd.print(msg2[1]);
63     lcd.print(" > ");
64 }
65 index++;
66 delay(1000);
67 }
68 delay(10000);
69 lcd.clear();
70 lcd.print("wait");
71
72 filteredPayload = result;
73 }

```

함수2,3

extractRow(String xml, int index)

XML 데이터에서 <row> 태그로 감싼 n번째 데이터만 잘라서 가져오는 함수입니다.

예: <row>1</row><row>2</row> → index가 1이면 <row>2</row> 반환

getNumber(String xml, String startTag, String endTag)

XML 또는 문자열에서 특정 태그 사이 값만 뽑아주는 함수입니다.

예: <statNm>한대앞</statNm> → "한대앞" 반환

```

1  String extractRow(String xml, int index) {
2      int start = 0;
3      for (int i = 0; i <= index; i++) {
4          start = xml.indexOf("<row>", start);
5          if (start == -1) return "";
6          // 다음 탐색을 위해 end 이후로 이동
7          int end = xml.indexOf("</row>", start);
8          if (end == -1) return "";
9          if (i == index) {
10             return xml.substring(start, end + 6);
11         }
12         // 다음 <row> 탐색을 위해 end 다음으로 이동
13         start = end + 6;
14     }
15     return "";
16 }
17 String getNumber(String xml, String startTag, String endTag) {
18     int start = xml.indexOf(startTag);
19     if (start == -1) return "";
20     start += startTag.length();
21     int end = xml.indexOf(endTag, start);
22     if (end == -1) return "";
23     return xml.substring(start, end);
24 }
25

```


2.아두이노회로설명

준비물 : 버튼2개,esp8266,선,lcd모터터

처음은 아두이노의 고장을 방지하기위해 보드는 다르지만 tinkercad웹사이트에서 제작을 하였습니다.

버튼 == : 왼쪽 : D3,D4 오른쪽 : 빵판 -

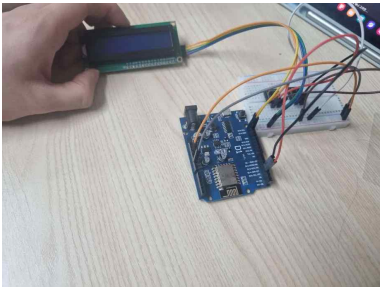
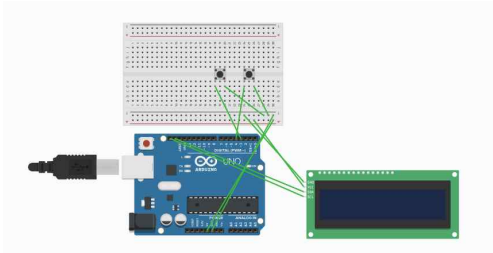
LCD모터터 == SCL : D15 SDA : D14 VCC : 빵판 + GND : 빵판 -

아두이노 보드 : 고장 방지를 위해 3.3V D가 들어간 숫자가 들어가지 않는 반대편 GND

주의할점: esp8266은 아두이노 코드에서는 하나씩 밀려서 코드를 작성해야한다.

ex) D3 -> D4 D4 -> D5

3.아두이노 전체 회로 사진

실제 제작 아두이노 회로	tinkercad에서 제작한 회로 (esp8266은 아님)
	

3.추가 아이디어

사용 도구: 앱 인벤터(App Inventor)

ESP8266 모듈의 Wi-Fi 기능을 보다 효율적으로 활용하기 위해, 앱 인벤터를 이용하여 전용 어플리케이션을 제작하였습니다. 이 어플을 통해 휴대폰에서 데이터를 전송하거나 버튼을 조작할 수 있으며, 그 결과를 LCD와 휴대폰 화면에 동시에 출력할 수 있도록 구상하였습니다.

1. 디자인

앱인벤터 화면 디자인	실제 어플 디자인

2.작동설명

ESP8266의 Wi-Fi가 연결되면, LCD 모니터에 "Button Push"라는 문구가 표시됩니다. 이후 상행선 버튼 또는 하행선 버튼 중 하나를 누르면, LCD에 몇 정거장이 남았는지 표시됩니다. 그 다음, "조회하기" 버튼을 누르면, 검은색 화면에 한글로 남은 정거장 수가 표시되는 어플입니다.

3.앱인벤터 코드

버튼 3개

URL의 type 값으로 type1, type2, 없음(null)을 설정하여, 해당 버튼이 상행버튼, 하행버튼, 또는 조회하기 버튼인지 구분할 수 있도록 했다.

텍스트 처리

웹1에서 텍스트를 받으면, 해당 텍스트를 label1에 표시되도록 구현했다.

