|  |  |
| --- | --- |
| 과목명 | 임베디드 시스템 개론 |
| 담당교수 | 안영신 교수님 |
| 조이름 | 1532 |
| 조원 | 박종호, 오나연, 이현정, 최정환 |
| 제출일 | 2017.06.09 |

REPORT

**임베디드 시스템 개론 최종 보고서**

목차

[1. ’스마트 우편함’ 2](#_Toc484716193)

[(1)개발배경 2](#_Toc484716194)

[(2)아두이노 스케치 2](#_Toc484716195)

[(3)아두이노 통신방법\_WeMosD1+PushBullet 6](#_Toc484716196)

[2. 조 장점과 문제해결 경험 11](#_Toc484716197)

[(1)1532조의 장점 11](#_Toc484716198)

[(2)문제 해결의 경험 11](#_Toc484716199)

[3. Toon\_1532 14](#_Toc484716200)

[1주차 14](#_Toc484716201)

[2주차 15](#_Toc484716202)

[3주차 16](#_Toc484716203)

[4주차 17](#_Toc484716204)

[5주차 18](#_Toc484716206)

[6주차 19](#_Toc484716208)

[7주차 20](#_Toc484716210)

[4. 소감 21](#_Toc484716212)

[박종호 21](#_Toc484716213)

[오나연 21](#_Toc484716214)

[이현정 21](#_Toc484716215)

[최정환 22](#_Toc484716216)

1. ’스마트 우편함’

# (1)개발배경

누구나 우편함에 도착한 공과금 고지서나 중요한 서류들을 잊고 지나친 경험이 있을 것이다. 현대사회는 지식정보산업이 극도로 발달한 사회이기에, 사람들은 스마트폰 혹은 PC로 많은 업무를 처리할 수 있다. 덕분에 우리는 우리의 우편함에 도착한 수 많은 우편물들을 자주 까먹고는 한다. 그러나 종이로 된 서류들은 자주 발송되며, 여전히 중요하다.

아두이노는 간단한 코딩방식과 다양한 부속품들로 다채로운 제품들을 만들어낼 수 있다. 우리 1532조는 이 같은 아두이노의 응용성을 바탕으로 ‘스마트 우편함’을 개발하기로 결정했다. 우리는 이 프로젝트를 통해 지식정보사회를 살아가는 우리 인류가 겪을 수 있는 아주 작지만 중요한 어려움들을 해결할 수 있을 것이라 기대한다.

# (2)아두이노 스케치

|  |
| --- |
| #include *<ESP8266WiFi.h>*  #include *<WiFiClient.h>*  #include *<WiFiClientSecure.h>*  #include *<WiFiServer.h>*  #include *<WiFiUdp.h>*    **const** **char**\* ssid = "ASVMPTOTE"; *// 와이파이 이름*  **const** **char**\* password = "whereareyounow?"; *// 와이파이 비밀번호*  **const** **char**\* accessToken = "o.TSh4FVHVkqFcuD5C0pOxPgC9GMPK6xKd"; *// PushBullet 토큰*  **String** Title = "1532조 스마트 우편함";    WiFiClientSecure client;  #define echo D7 *// 에코 핀*  #define trig D6 *// 트리거 핀*  **int** def = 10; *//기본 값*  **int** temp; *//측정 값을 임시로 저장할 변수*  **void** setup() {  Serial.begin(9600);  WiFi.begin(ssid, password);  **while** (WiFi.status() != WL\_CONNECTED) {  delay(500);  }  pinMode(trig,**OUTPUT**); *//6번핀을 출력 모드로 설정한다*  pinMode(echo,**INPUT**); *//7번핀을 입력 모드로 설정한다*  }    **void** loop() {  temp = distance();  Serial.println(temp);  **if**(def>temp){  sendToPushBullet("우편 도착");  def = temp;  }  **else** **if**(def<temp){  sendToPushBullet("우편 회수");  def = temp;  }  **else**{  def = temp;  }  delay(6000); *//딜레이 6.0초*  }  **int** distance(){  digitalWrite(trig,**HIGH**); *//trig핀에 HIGH 신호를 보낸다*  delayMicroseconds(30); *//10마이크로(1/100,000초) 동안*  digitalWrite(trig, **LOW**); *//trig핀에 LOW신호를 보낸다*  **return** pulseIn(echo,**HIGH**) \*17 /1000; *//echo핀이 다시 HIGH 신호를 받기까지의 시간 \*17/1000 값을 반환*  }  **void** sendToPushBullet(**String** Msg) {  Msg = "{**\"**type**\"**: **\"**note**\"**, **\"**title**\"**: **\"**" + Title + "**\"**, **\"**body**\"**: **\"**" + Msg + "**\"**}**\r\n**";  **if** (client.connect("api.pushbullet.com", 443)) {  client.println("POST /v2/pushes HTTP/1.1");  client.println("Host: api.pushbullet.com");  client.println("Authorization: Bearer " + **String**(accessToken));  client.println("Content-Type: application/json");  client.println("Content-Length: " + **String**(Msg.length()) + "**\r\n**");  client.print(Msg);  client.stop();  }  } |

**● distance() 코드 해설**

**거리 = 시간 X 속도**

초음파를 통해 거리를 구하기 위해서는 시간과 속도를 알아야 한다. 초음파의 속도는 초당 340m이며,

시간을 구하기 위해서는 우리는 pulseIn()이라는 함수를 사용해야 한다. pulseIn(echo,HIGH) 함수는 echo핀이 HIGH 상태에서 LOW 신호를 받고, 다시 HIGH를 받기까지의 시간, 즉 다시 HIGH가 되기까지의 시간을 반환한다. 이는 10ms(마이크로 초) 동안 쏜 초음파를 받기까지의 시간을 구한다는 의미이다. 그리고 pulseIn() 함수로 구한 시간이 편도가 아닌 왕복 시간이기에 2로 나누었고, 측정한 거리를 cm로 단위로 표현하기 위해 1000으로 또 나누었다. 다음은 함수에 대한 설명이다.

① pulseIn(핀번호, 신호): 설정 신호를 다시 입력 받을 때까지 걸리는 시간을 측정하기 위한 함수

-매개변수 -> 핀번호: 함수를 사용할 핀 설정

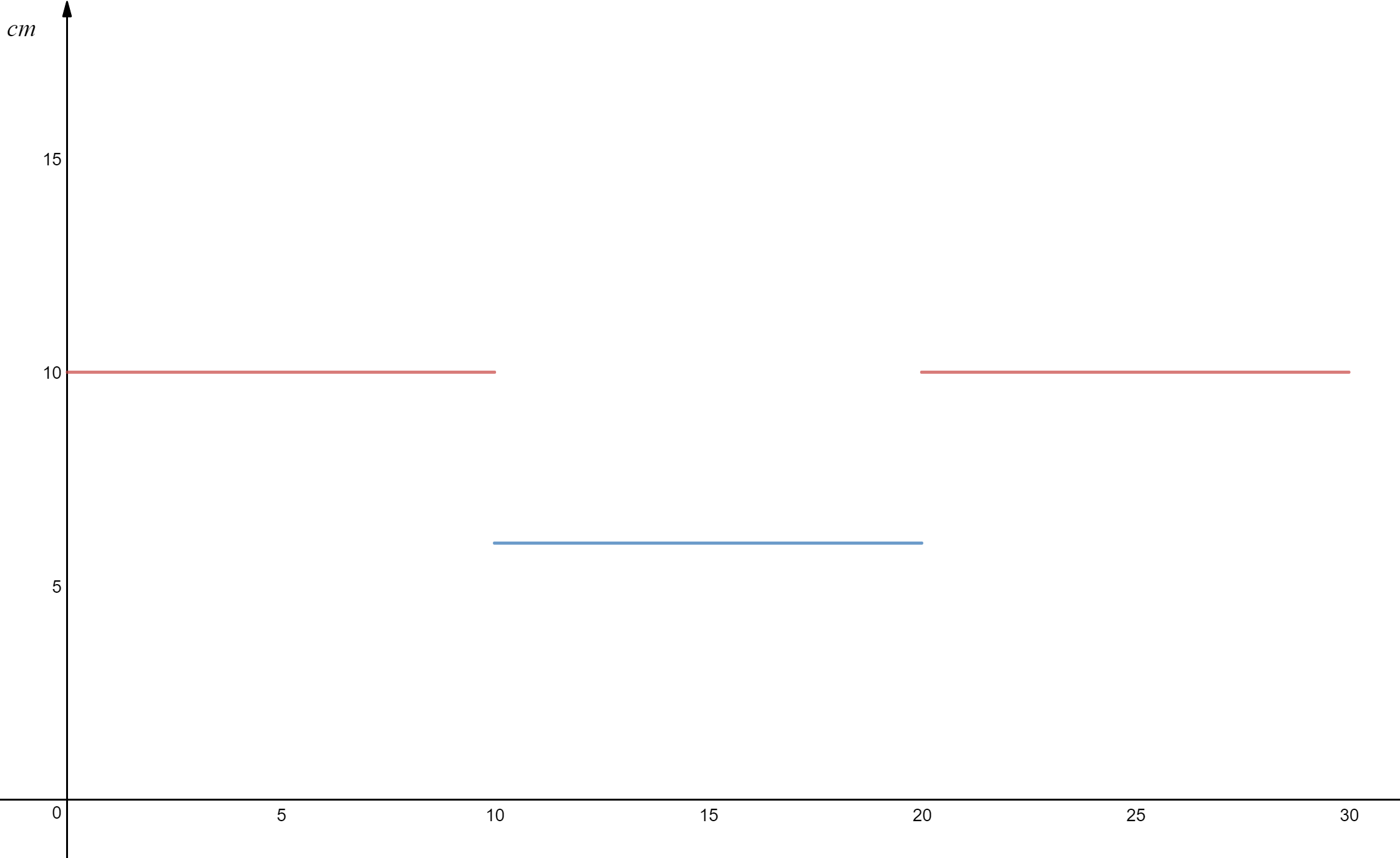
-신호: HIGH 또는 LOW

-반환값 -> 설정 신호가 입력될 때까지의 시간을 반환, 지정된 시간 동안 신호 변화가 없을 경우 0을 반환

② delayMicrosecondes(시간): 다음 명령어(코드)까지 입력 시간만큼 동작을 유보하는 함수

- 매개변수 -> 시간: 마이크로 초 단위로 숫자를 입력 (1,000,000 = 1초)

**●loop() 코드 해설**



① ② ③

distance() 함수를 이용해 거리 값을 temp에 저장 후 default 값과 비교하였다.

①: 우편물 도착했을 때 (default > temp)

default 🡨 temp

temp 값을 default 값으로 다시 설정

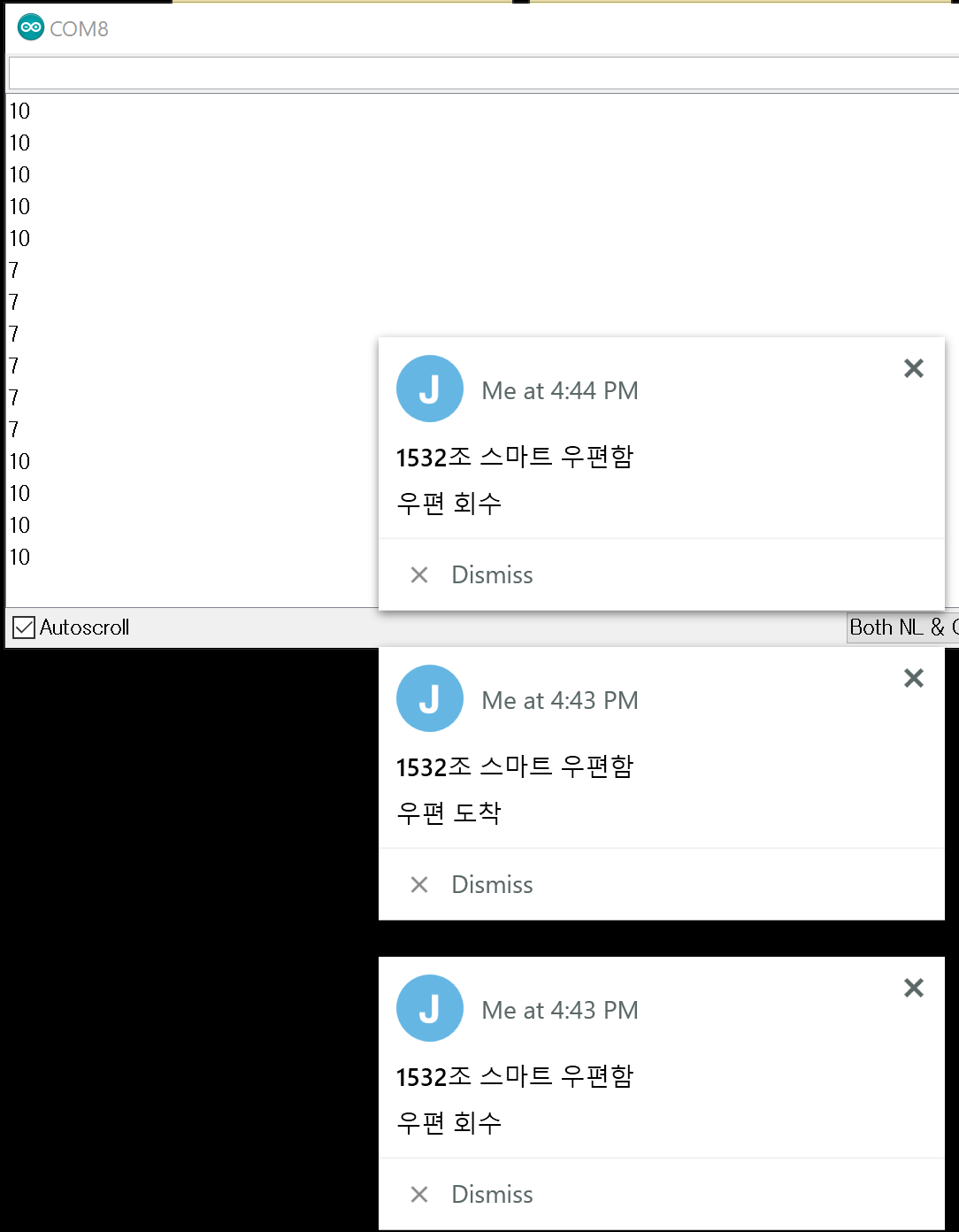
즉, 기준이 되는 값이 바뀜

②: 변화가 없을 때 (default == temp)

default 🡨 temp

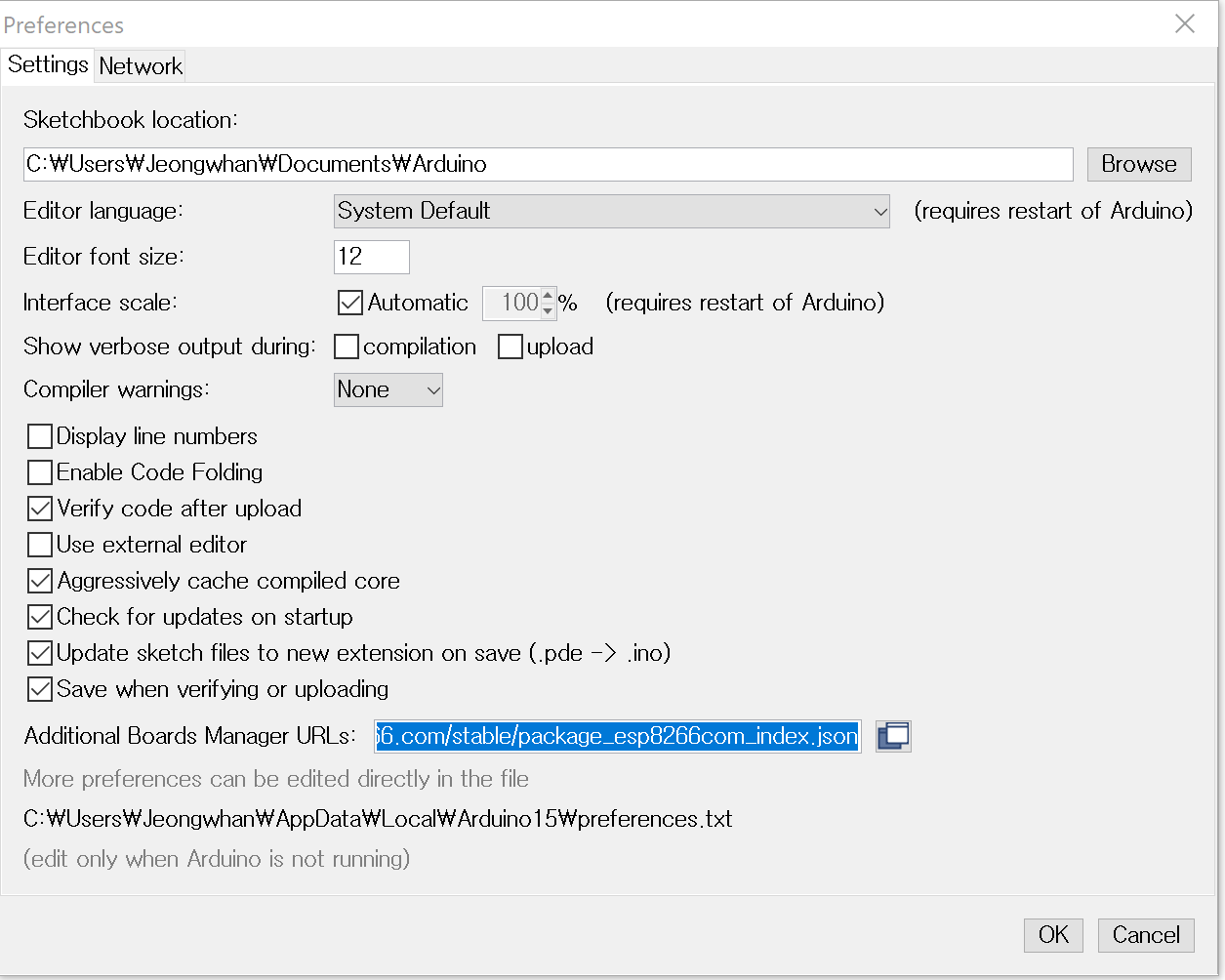
③: 우편물 회수했을 때 (default < temp)

default 🡨 temp



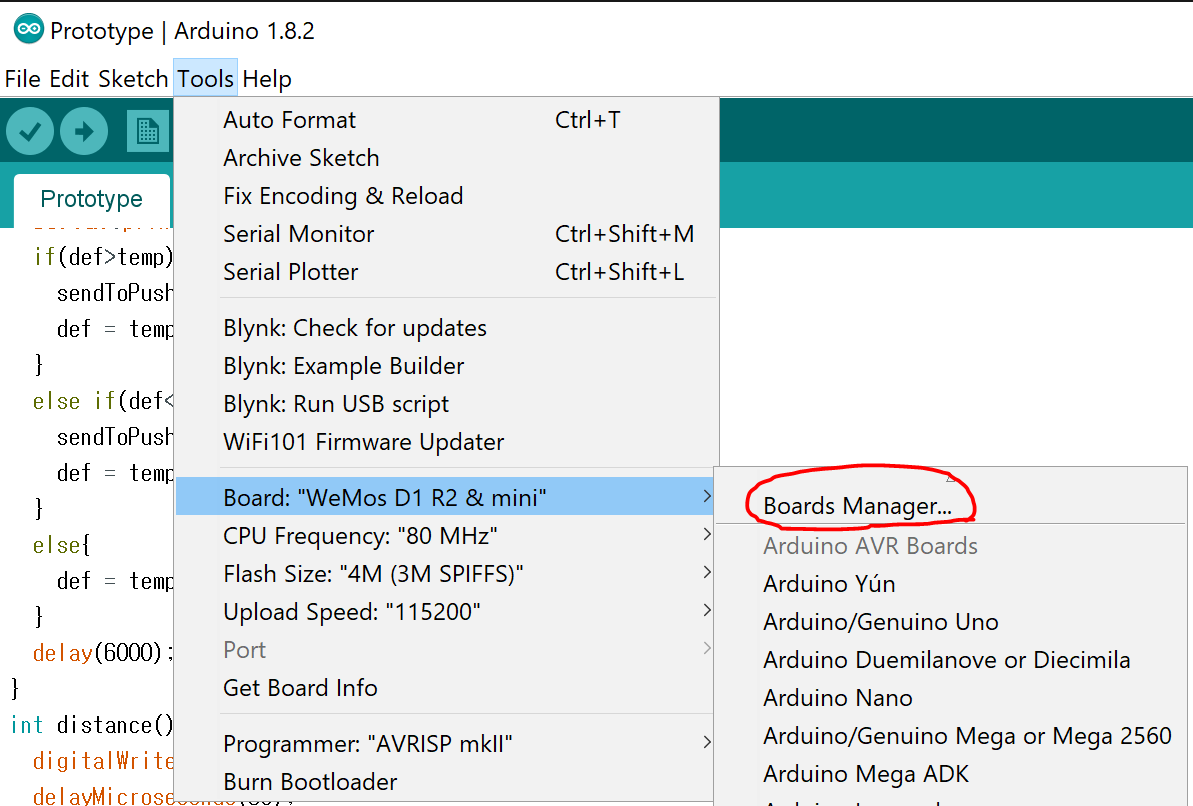
<실행화면>

# (3)아두이노 통신방법\_WeMosD1+PushBullet

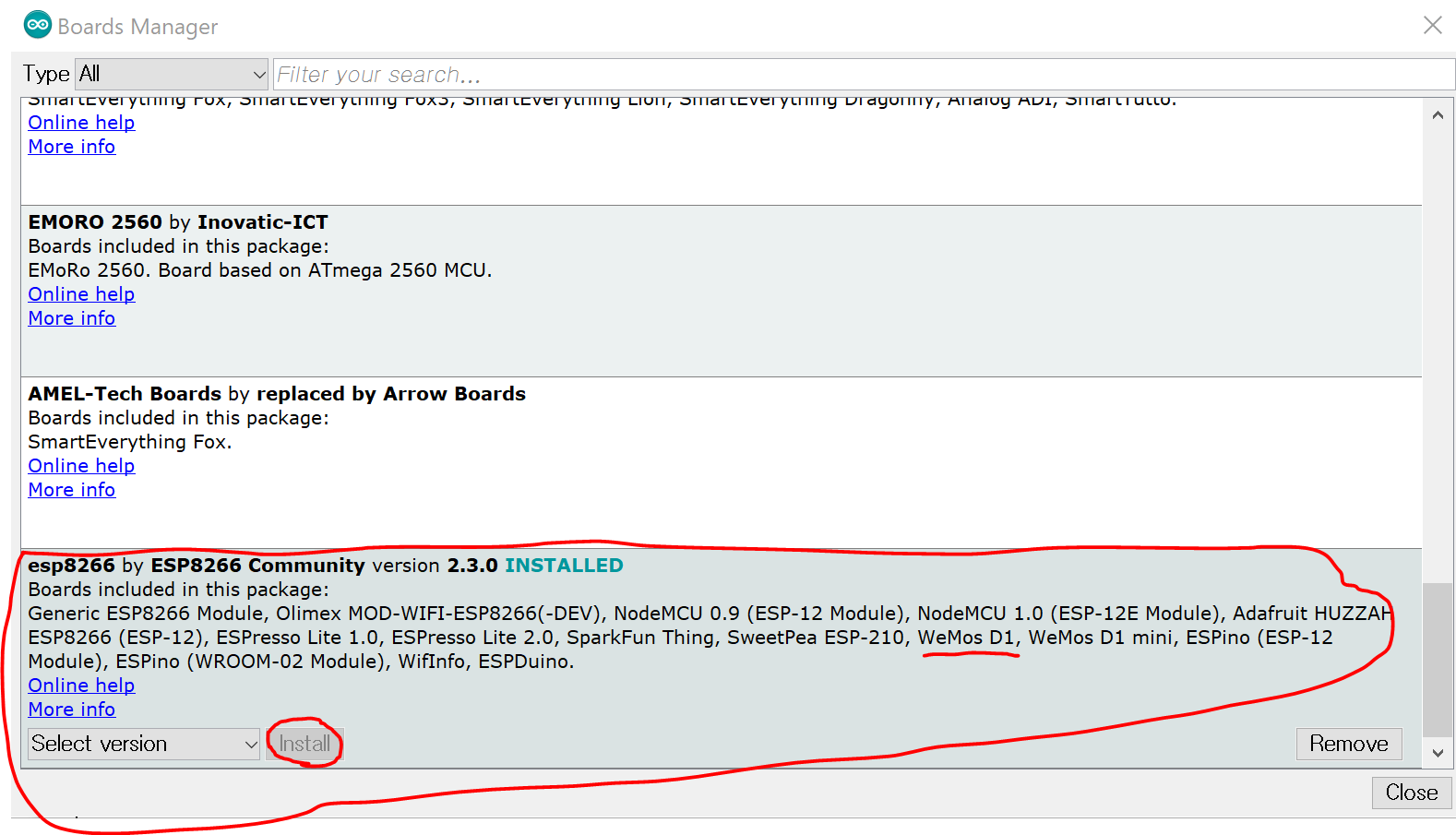


-Arduino IDE 실행 후 File – Preferences -Additional Board Manager URLs에

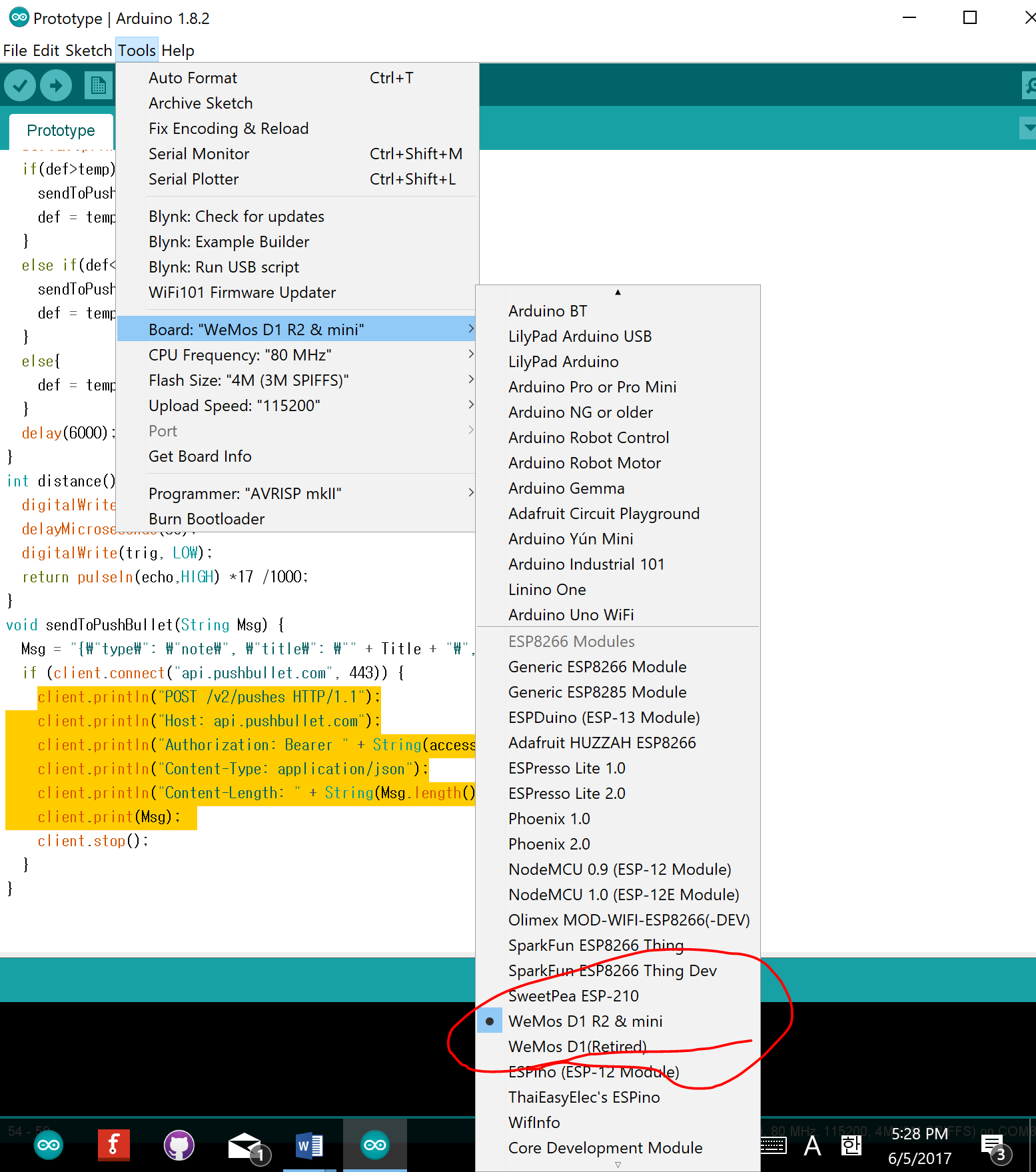
‘http://arduino.esp8266.com/stable/package\_esp8266com\_index.json’ 링크의 URL을 추가해준다.



-Tools 🡪 Board🡪 Boards Manager로 들어간다.



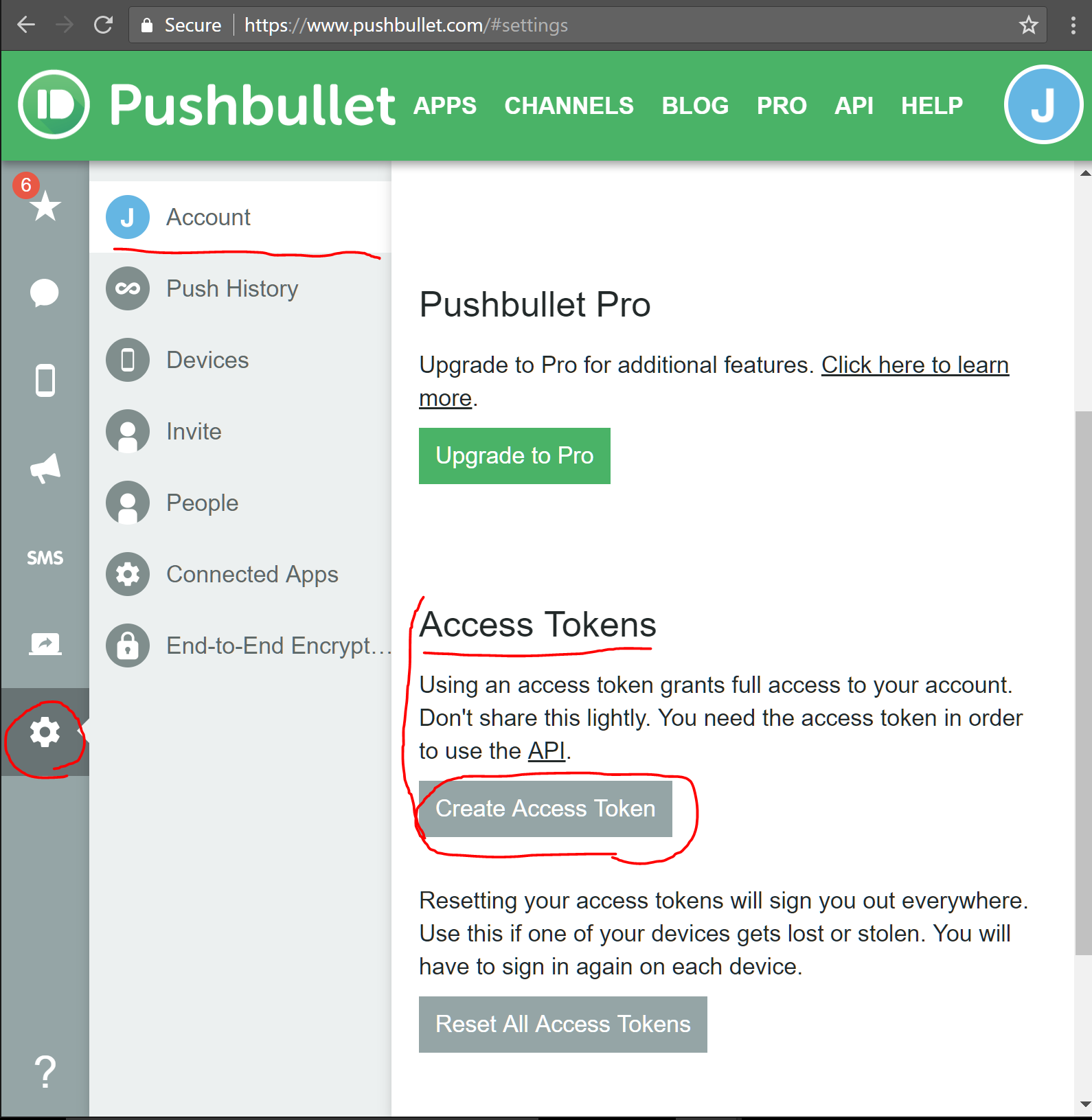
-esp8266 Community 설치(Install)를 눌러 최신버전 설치한다.



-Tools 🡪 Board 🡪 WeMos D1 R2 &mini 로 선택한다.



-esp8266-Blink 예제를 아두이노에 업로드 성공하고 이상 없음을 확인했다.



-로그인 후 환경설정에서 Access Token을 받는다. 그리고 프로그래밍 시 const char\* accessToken = “xxxxxxxxxx”; 를 사용한다. xxxxxxxxx에 PushBullet Access Token을 작성하면 된다.

#include *<ESP8266WiFi.h>*

#include *<WiFiClient.h>*

#include *<WiFiClientSecure.h>*

#include *<WiFiServer.h>*

#include *<WiFiUdp.h>*

**const** **char**\* ssid = "\*\*\*\*\*\*"; *// 와이파이 이름*

**const** **char**\* password = "\*\*\*\*\*\*\*\*"; *// 와이파이 비밀번호*

**const** **char**\* accessToken = "xxxxxxxxxxxxxxxxxxx"; *// PushBullet 토큰*

**String** Title = "PushBullet 테스트입니당";

WiFiClientSecure client;

**void** setup() {

WiFi.begin(ssid, password);

**while** (WiFi.status() != WL\_CONNECTED) {

delay(500);

}

}

**void** loop() {

sendToPushBullet("설명하기 위한 푸시 알림");

delay(5000);

}

**void** sendToPushBullet(**String** Msg) {

Msg = "{**\"**type**\"**: **\"**note**\"**, **\"**title**\"**: **\"**" + Title + "**\"**, **\"**body**\"**: **\"**" + Msg + "**\"**}**\r\n**";

**if** (client.connect("api.pushbullet.com", 443)) {

client.println("POST /v2/pushes HTTP/1.1");

client.println("Host: api.pushbullet.com");

client.println("Authorization: Bearer " + **String**(accessToken));

client.println("Content-Type: application/json");

client.println("Content-Length: " + **String**(Msg.length()) + "**\r\n**");

client.print(Msg);

client.stop();

}

}

< Wifi를 구현하는 아두이노 스케치>

2. 조 장점과 문제해결 경험

# (1)1532조의 장점

첫째로 1532조는 구글링을 통해 쉽게 얻을 수 있는 코드만을 사용하지 않았다. 최종 보고서의 1.’스마트 우편함’-(2)아두이노 스케치 단락에서 보듯이, 우리 조는 간단하지만 직접 작성한 로직과 코드를 사용하였다. 쉽고도 간단한 이 코드를 구현하기 위해, 팀원들이 대화를 나누며 문제점을 개선해 나가는 과정에서 깊은 성취감을 느낄 수 있었다.

둘째로 최종 보고서의3. TOON\_1532단락에서 볼 수 있듯이 보고서를 읽는 이로 하여금 흥미를 느끼게 하고, 프로젝트 진행 시에 팀원 간의 긴장완화를 위하여 웹툰 형식의 4컷만화를 주마다 그렸다. 이 만화는 주간 회의 및 실험 중에 겪었던 다양한 해프닝과 조원들의 심리 변화를 담고 있어, 뒤이어 프로젝트를 진행할 후배들에게도 좋은 자료가 될 것이라 기대한다.

셋째로 교수님께서 강의 시간마다 ‘아두이노’와 연계해 와이파이를 구현하는 것이 쉽지 않다고 자주 언급을 하셨다. 그러나 1532조는 와이파이 통신을 구현하는데 성공해 결과물을 제출할 수 있게 됐다. 1532조 말고 와이파이 구현에 성공한 조가 1조 더 있는 것으로 알고 있지만, 그러한 사실을 감안하더라도 예상치 못한 좋은 성과라고 자신한다.

넷째로 1532조는 ‘스마트 우편함’의 기능을 버전별로 기록하고 단계적으로 발전시켰다. 별첨한 실험노트를 확인하면 알 수 있듯이 최초 버전은 우편물을 인식하면 단순하게 LED를 반짝이는 수준에서 머물렀지만, 뒤이어 블루투스를 이용하여 스마트폰에 신호를 전송하도록 발전시켰다. 더불어 우리는 이에서 멈추지 않고, 와이파이를 이용한 통신에 도전하여 보기 좋게 성공시켰다. 이는 결과물을 내는 과정을 꼼꼼히 기록하고, 각 버전을 발전시키기 위해 성실하게 노력한 결과라고 사료된다.

# (2)문제 해결의 경험

**●**초음파 센서의 오작동 해결

초음파 센서는 초음파를 쏘고 부딪혀서 초음파가 돌아오는 시간을 재어 거리를 측정하는 방식이다. 이러한 초음파의 성질 중에는 산란(Scatter)이라는 부분이 있다. 이는 초음파 빔의 파장보다 불규칙하거나 작은 면을 가진 반사체에 부딪칠 때 초음파 빔이 여러 방향으로 흩어지는 것을 말한다.

이러한 성질을 가진 초음파 센서가 부착되어 있는 우편함 내에는 다양한 우편함 부속물로 인해 불규칙하고 작은 면을 가진 부분들이 있었다. 때문에 최초에 초음파 센서를 우편함 입구 정면에 설치했을 때는 측정되는 거리 값이 불안정했다. 이 문제를 해결하기 위해 우리는 우편물 삽입각도를 제한하거나 우편함의 가로방향에 초음파센서를 설치하는 등의 조치를 해보았으나 결과는 크게 달라지지 않았다.

고민 끝에 초음파센서를 우편함 내부의 상단에 설치하여 바닥에 초음파를 쏘는 방식을 취하였다. 우편함의 바닥은 다른 면에 비해 매끄러워서 측정되는 거리 값이 안정적이었다. 해결하기 어려울 것이라고 생각이 돼 센서를 교환할 생각까지 했는데, 생각보다 간단한 해결방법에 맥이 빠지기도 했다. 그러나 문제점을 해결하는 순간의 쾌감은 정말 상당했다.

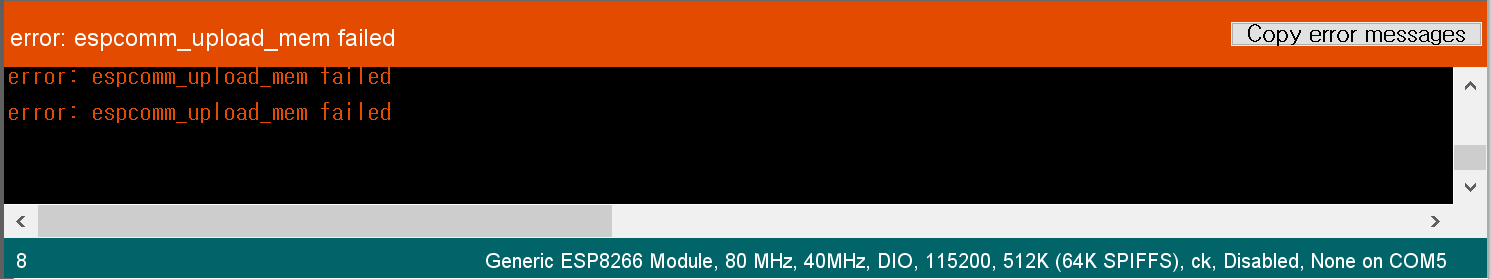
**●**아두이노 통신문제 해결

아두이노 UNO를 통해 esp8266에 펌웨어 업로드를 하고 와이파이 연결까지 성공했다. 그러나 코드를 업로드 할 때 “error: espcomm\_upload\_mem failed”라는 오류가 발생했다. 그 대안으로 USB TO TTL을 사용해 esp8266에 업로드를 해보았지만 해결하지 못했다. 이를 해결하기 위해, 구글링을 해 비교적 자주 사용되는 FTDI(USB TO UART)를 사용해 업로드를 시도해보았지만 이번에도 실패하였다.

계속된 실패로 2주동안 계획이 정체되자 우리 조는 esp8266이 내장되어 있는 보드인 Node MCU와 WeMos D1을 사용키로 결정하고 주문을 했다. 두 보드 중에 WeMos D1을 먼저 사용해 코드 업로드를 성공했다. 떄문에 Node MCU는 시도해보지 못했고 우리 조와 같이 esp8266 코드 업로드에 어려움을 겪고 있는 ‘호두마루’조에 Node MCU를 빌려주었다.

esp8266 코드 업로드를 우히ㅐ 2주 동안 어려움을 겪어 무척 힘들었다. 그러나 관련 정보를 수집하는 노력을 게을리 하지 않아, 와이파이를 통한 푸시알림 기능을 구현할 수 있게 됐다.

|  |  |  |
| --- | --- | --- |
| 업로드 방법 | 코드 업로드 성공 여부 | 사진 |
| Esp-01 + 아두이노 UNO | X | C:\Users\Jeongwhan\OneDrive\20170608_111915953_iOS.jpg |
| esp-01 + USB TO TTL | X | C:\Users\Jeongwhan\OneDrive\20170608_111949943_iOS.jpg |
| esp-01 + USB TO UART (FTDI) | X | C:\Users\Jeongwhan\OneDrive\20170608_112003625_iOS.jpg |
| WeMos D1 | O | C:\Users\Jeongwhan\OneDrive\20170608_112048062_iOS.jpg |
| Node MCU | 시도 해보지 못함 | (다른 조 빌려줌) |



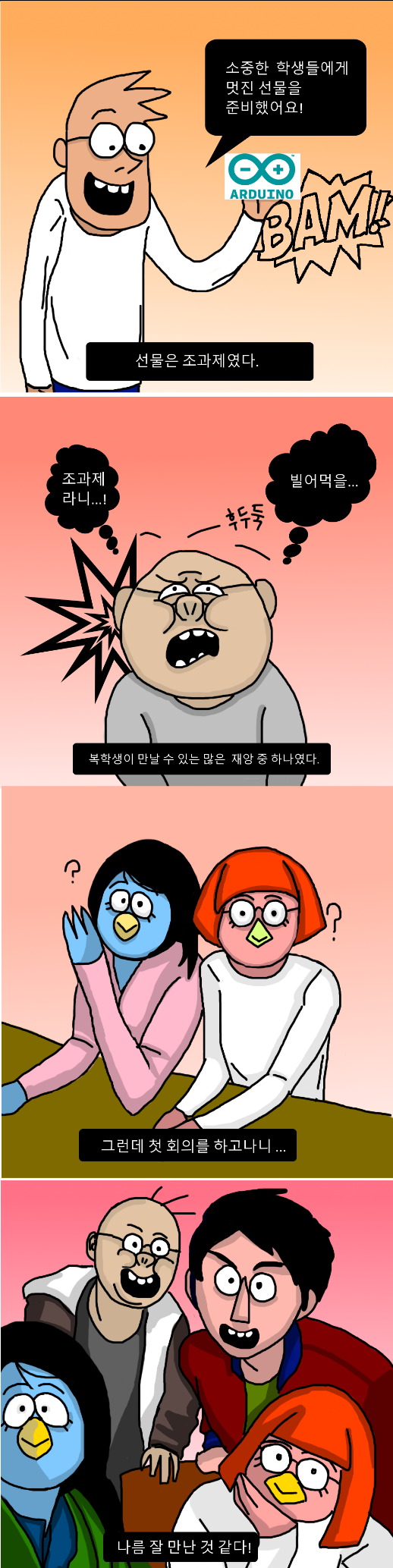
<코드 업로드에 실패했을 때의 결과화면>



<WeMos D1보드를 이용하여 코드 업로드에 성공>

3. Toon\_1532

# 1주차



# 2주차



# 3주차



# 4주차

# C:\Users\user\AppData\Local\Microsoft\Windows\INetCache\Content.Word\합.png

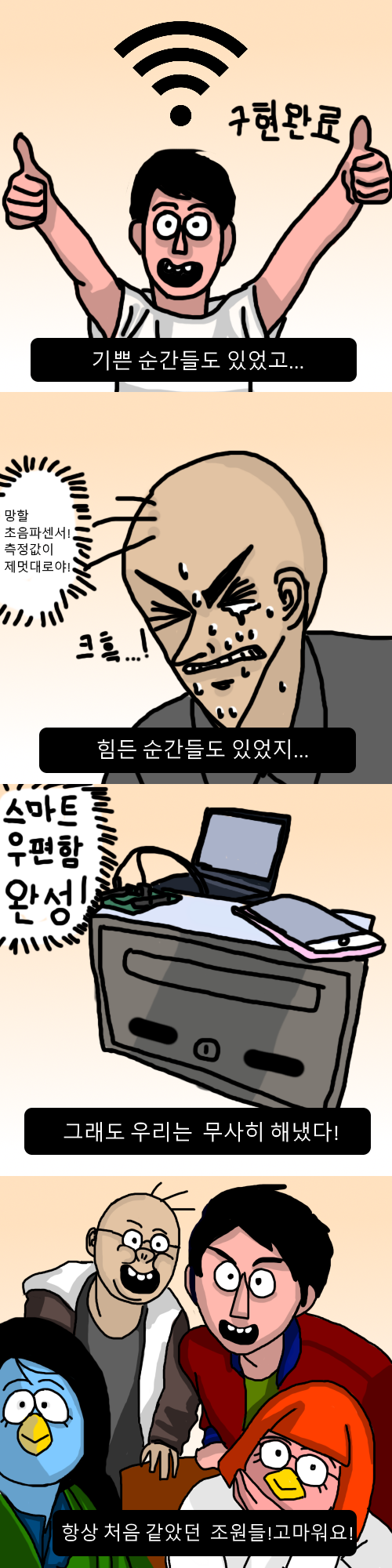
# 5주차



# 6주차



# 7주차



4. 소감

|  |  |
| --- | --- |
| 박종호 | 무언가를 개발하는 것의 기쁨을 처음으로 깨닫게 된 소중한 시간이었다. 4명의 조원들이 같은 문제로 다같이 고민하고, 그것이 해결될 때의 기쁨을 공유하는 경험을 통해, 개발이라는 행위의 순수한 행복감을 느낄 수 있었다.  더불어 조원과의 의사소통과 업무 분배를 직접 진행해가면서, 실질적인 그룹활동의 운영 감각과 경험을 쌓는 좋은 기회였다. ‘아두이노’라는 제품의 사용법을 배워보는 것도 실속 있는 학습이었다고 생각한다.  부족한 조원을 데리고, 한결같이 프로젝트를 진행해준 조원들과 이렇게 멋진 경험을 선사해주신 교수님께 감사의 말씀을 드리고 싶다. |

|  |  |
| --- | --- |
| 오나연 | 조원 모두가 ‘아두이노’라는 것을 처음 접해봐서 프로젝트를 진행하는 데에 많은 어려움들이 있었다. 처음 프로젝트를 시작하고 LED를 켜는 것을 할 때까지는 해볼 만 하다는 생각을 가지고 있었지만 프로젝트를 진행하면 할수록 문제점들이 생겼다.  작고 간단한 문제도 있었지만 프로젝트의 진행을 방해하는 큰 문제점이 하나씩 생기면서 고민도 많아지고 머리도 아팠다. 하지만 든든한 조원들과 함께 그 문제들을 해결해 나가는 과정이 재미있었고 ‘아두이노’에 흥미를 느끼게 되었다.  흔치 않은 좋은 기회로 ‘아두이노’를 직접 다룰 수 있어서 정말 좋은 경험이 되었고 어려웠지만 조원들과 잘 맞아서 순조롭고 재미있는 조별 과제였다. 다음에 기회가 된다면 또 다시 ‘아두이노’를 이용한 프로젝트를 진행해보고 싶다. |

|  |  |
| --- | --- |
| 이현정 | 처음 ‘아두이노’ 프로젝트에 대해 설명을 들었을 때, 한번도 해보지 못한 것에 대한 두려움이 컸다. 그리고 한 학기 동안 팀별로 활동해야 하는 것이 였기 때문에 혹시나 팀원들과 불화가 있을까 걱정이 되었다.  하지만 수많은 걱정과 달리 좋은 팀원들을 만나 순조롭게 프로젝트를 진행할 수 있었으며, 다른 팀원들의 도움으로 ‘아두이노’에 대해 이해하고 만들어보는 것에 대해 흥미를 느낄 수 있었다.  프로젝트 후반으로 갈수록 원하는 대로 되지않아 답답하기도 했지만 그것들을 극복했을 때 기분은 정말 최고였다. 열심히 노력한 팀원들에게 고맙고, 나에게 좋은 경험이 되는 프로젝트였다. |

|  |  |
| --- | --- |
| 최정환 | ‘아두이노’를 팀원들과 함께 두 달 가까이 다루어 보았다. 실험을 할 때 단추를 잘못 끼우면 제대로 된 위치에 다시 끼우기가 쉽지 않았다. 다음 번엔 그럴 리 없겠다 생각했지만 매 단계마다 잘못 끼우고 제자리에 다시 끼우기를 반복하였다.  따라서 모든 ‘아두이노’ 실험이 성공적이지 않았지만 문제 해결을 할 때에는 4명의 팀원이 하나가 되어 즐거워했다.  이번 프로젝트를 통해 ‘아두이노’에 대한 지식을 많이 쌓았다. 하지만 팀원들에게서 무언가를 배울 수 있는 사람이 세상에서 제일 현명하다는 점을 느꼈다는 것이 다른 무엇보다 더 중요하다. |