

Arduino



코딩교육 전문쇼핑몰 [에듀이노]





시작하기 전에

- 아두이노 프로그램 설치(Arduino-1.8.13 windows)
- 아두이노 호환 드라이버 설치(CH341SER.zip)
 - 압축을 푼 후 CH341SER폴더안에 setup실행
- PC와 아두이노 연결하기
- 보드와 포트 설정하기

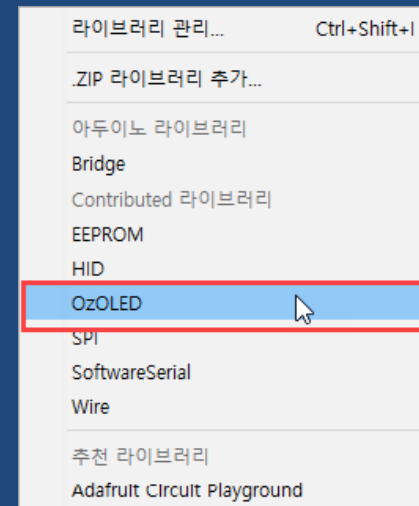
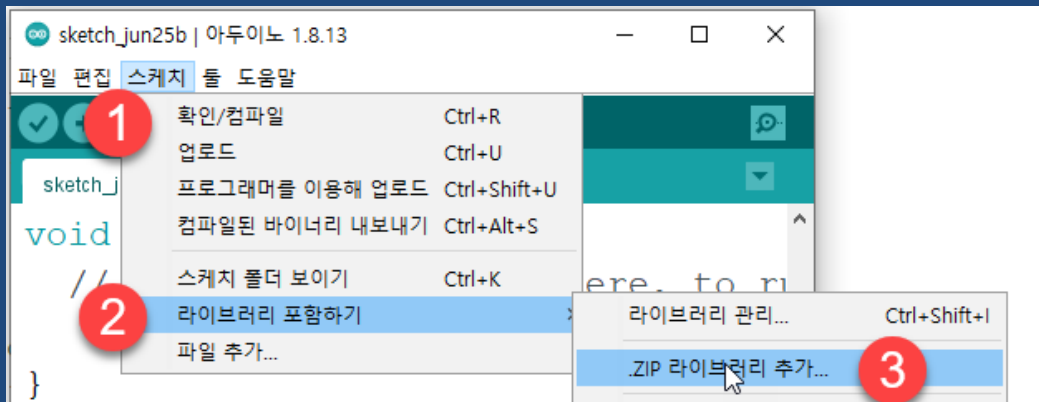




OzOLED 라이브러리 추가하기

- 아두이노 스케치 프로그램 실행 후

스케치 – 라이브러리 포함하기 – ZIP 라이브러리 추가
에서 다운받은 파일 선택



- 스케치 – 라이브러리 포함하기 에서 목록에 OzOLED라이브러리



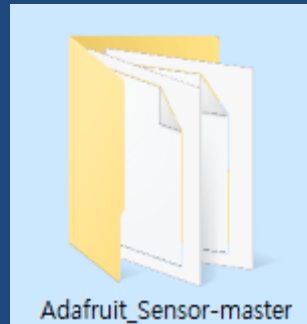
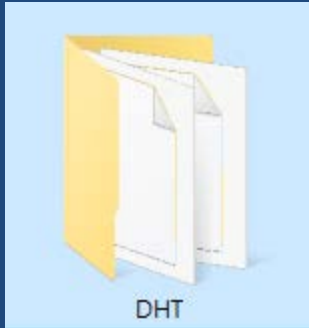
있는지 확인



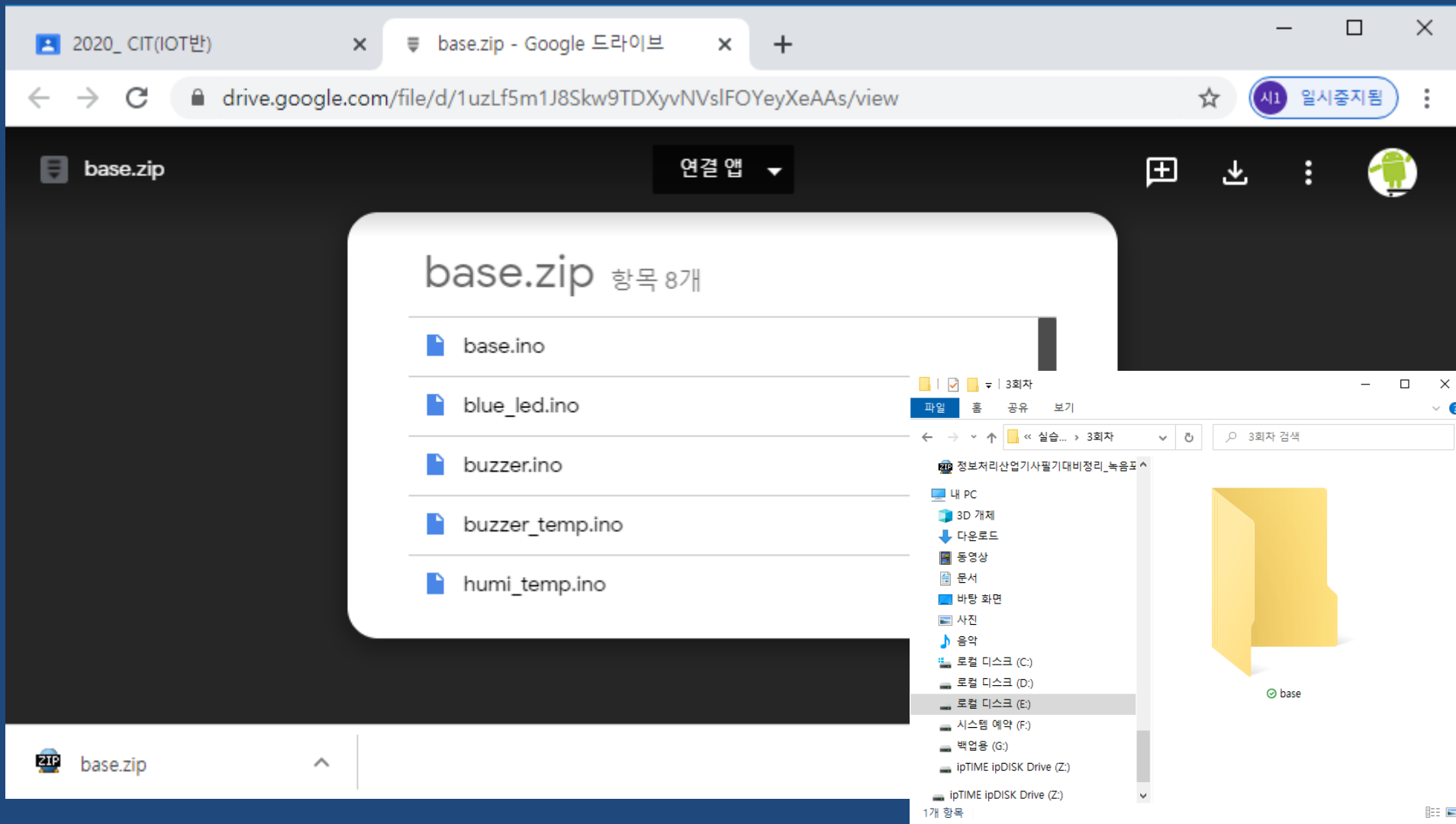


온습도센서 라이브러리 복사하기

- DHT 폴더와 Adafruit_Sensor-master폴더를
Arduino\libraries 하위폴더에 복사



이전 학습 자료 내려 받아 압축 풀기



The image shows a web browser window with two tabs. The first tab is labeled '2020_CIT(IOT반)' and the second tab is 'base.zip - Google 드라이브'. The address bar shows the URL 'drive.google.com/file/d/1uzLf5m1J8Skw9TDXyvNVsIFOYeyXeAAs/view'. The page content displays 'base.zip' with a download button and a list of 8 items. A Windows File Explorer window is open over the bottom right of the browser, showing the contents of the 'base' folder. The File Explorer window has a search bar with '3회차' and a list of files including 'base.ino', 'blue_led.ino', 'buzzer.ino', 'buzzer_temp.ino', and 'humi_temp.ino'.

base.zip 항목 8개

- base.ino
- blue_led.ino
- buzzer.ino
- buzzer_temp.ino
- humi_temp.ino

File Explorer: 3회차 검색

- 내 PC
- 3D 개체
- 다운로드
- 동영상
- 문서
- 바탕 화면
- 사진
- 음악
- 로컬 디스크 (C:)
- 로컬 디스크 (D:)
- 로컬 디스크 (E:)
- 시스템 예약 (F:)
- 백업용 (G:)
- ipTIME ipDISK Drive (Z:)
- ipTIME ipDISK Drive (Z:)

1개 항목

이전 학습 자료 내려 받아 압축 풀기

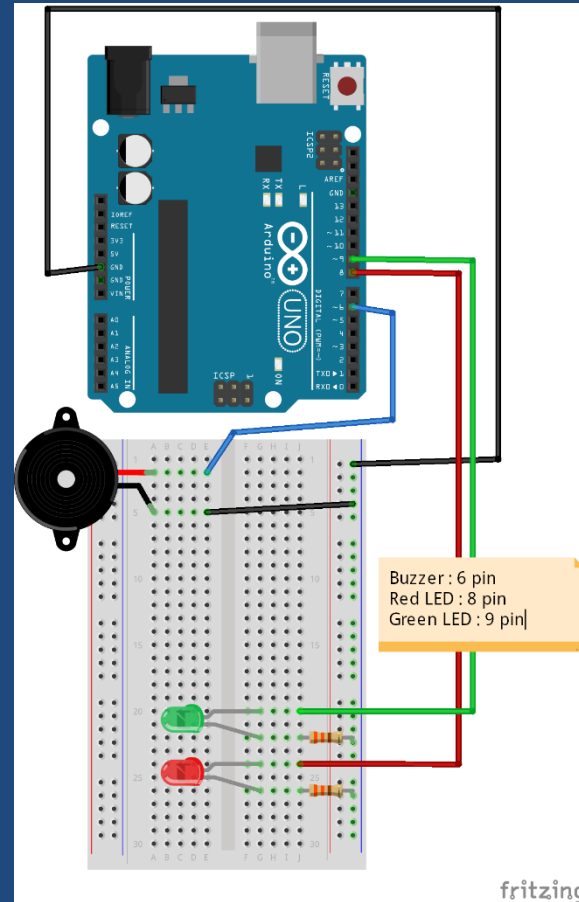
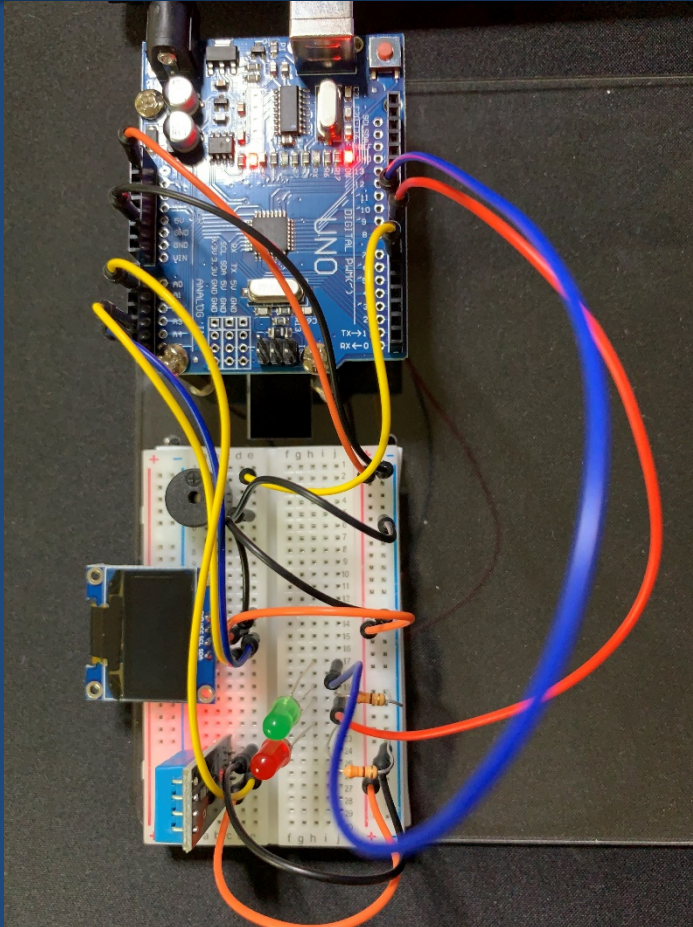


base	OzoLed	blue_led	buzzer	buzzer_temp	humi_temp	pitches.h	serial_menu
------	--------	----------	--------	-------------	-----------	-----------	-------------

- 왼쪽부터 오른쪽으로 순서대로 실행
- base : setup(), loop() 함수 존재
- OzoLed : Led 화면에 습도/온도 표시
- buzzer : 멜로디 재생
- buzzer_temp : 25도를 초과하면, 멜로디를 5번 재생
- humi_temp : 습도/온도를 측정
- pitches.h : 멜로디 재생용 헤더파일
- serial_menu : 시리얼 모니터로 메뉴 표시하고, 입력에 따른 명령 실행



LED를 연결해 볼까요?



부저는 6번 pin, 빨강 LED는 8번 pin, 녹색 LED는 9번 pin 입니다.

저항은 330옴(주주갈)



파랑 LED 동작(탭명 : blue_led) 은 다음과 같습니다.

```
#define BLUE_PIN 9
```

```
boolean flag = true;  
int fadeValue = 0;
```

```
void Blue_setup(){  
  pinMode(BLUE_PIN, OUTPUT);  
}
```

- 아날로그형태로 출력합니다.
- flag 가 true 이면, 서서히 (단계 20) 밝아지고, 밝기의 세기가 255가 넘으면, 다시 서서히 (단계 20) 어두워짐.



```
void Blue_loop(){  
  if(flag == true){  
    if(fadeValue <= 255) {  
      fadeValue += 20;  
      if(fadeValue > 255){  
        fadeValue = 255;  
        flag = false;  
      }  
    }  
  }  
  else {  
    if(fadeValue >= 0) {  
      fadeValue -= 20;  
      if(fadeValue < 0){  
        fadeValue = 0;  
        flag = true;  
      }  
    }  
  }  
  analogWrite(BLUE_PIN, fadeValue);  
}
```





빨강 LED 동작(탭명 : red_led) 은 다음과 같습니다.

```
#define RED_PIN 8
```

```
void Red_setup(){  
  pinMode(RED_PIN, OUTPUT);  
}
```

```
void Red_loop(){  
  digitalWrite(RED_PIN, digitalRead(RED_PIN)^1);  
  delay(300);  
}
```



- 디지털 형태로 출력합니다.
- ^ (배타적 연산자)로 인해 빨강 LED가 켜지고 꺼집니다.





이제 base.ino 에 동작을 위해 함수를 추가해 볼까요?

```
void setup() {  
  Serial.begin(9600);  
  Ozo_setup();  
  Serial_setup();  
  Red_setup();  
  Blue_setup();  
}
```

```
void loop() {  
  Humi_temp_loop();  
  Serial_loop();  
  Red_loop();  
  Blue_loop();  
  (하략)  
}
```





LED 동작도 메뉴화해서 동작시킬 수 없을까요?





이제 base.ino 에 시리얼 모니터에 메뉴 생성을 위한
boolean 변수 추가 설정

```
boolean led_humi_temp = false;  
boolean buzz = false;  
boolean temp_chk = false;  
boolean red_led_chk = false;  
boolean blue_led_chk = false;
```





이제 base.ino 에 boolean 변수의 값이 따른 실행을
위한 loop() 함수 수정

```
void loop() {
```

```
  Humi_temp_loop();
```

```
  Serial_loop();
```

```
  Red_loop();
```

```
  Blue_loop();
```

```
    if(led_humi_temp == true)
```

```
      Ozo_loop();
```

```
    if(buzz == true)
```

```
      Buzzer_loop();
```

```
    if(temp_chk == true){
```

```
      Ozo_loop();
```

```
      Buzzer_temp_loop();
```

```
    }
```

```
    if(red_led_chk == true){
```

```
      Ozo_loop();
```

```
      Red_loop();
```

```
    }
```

```
    if(blue_led_chk == true){
```

```
      Ozo_loop();
```

```
      Blue_loop();
```

```
    }
```





이제 serial_menu.ino 를 활용해 메뉴를 추가해 봅니다.

```
void Showmenu(){  
    Serial.println("1. 온습도 표시/끄기");  
    Serial.println("2. 멜로디 재생/끄기");  
    Serial.println("3. 온도 체크/취소");  
    Serial.println("4. 빨강 LED 켜기/끄기");  
    Serial.println("5. 초록 LED 켜기/끄기");  
}
```





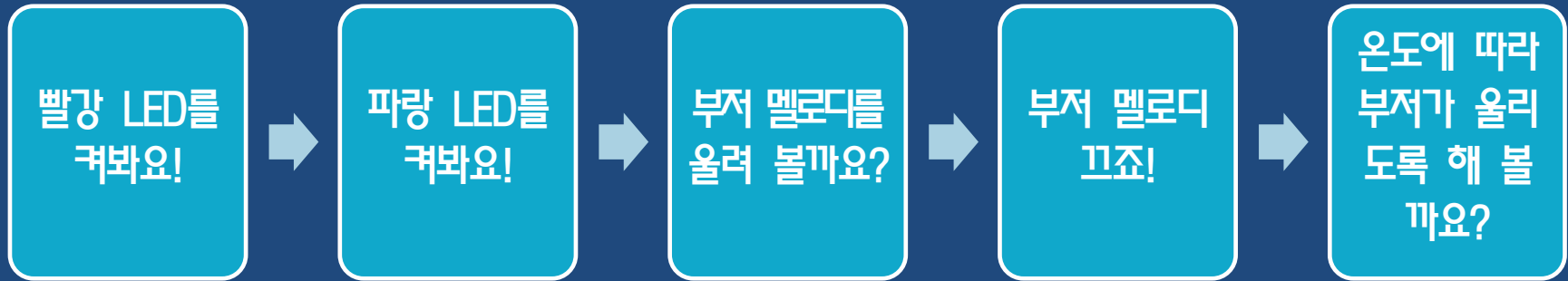
이제 serial_menu.ino 를 활용해 메뉴를 추가해 봅니다.

```
switch(input_menu){  
  case '1' : led_humi_temp = !led_humi_temp;  
    OzOled.init();  
    break;  
  case '2' : buzz = !buzz;  
    OzOled.init();  
    break;  
  case '3' : temp_chk = !temp_chk;  
    OzOled.init();  
    break;  
  case '4' : red_led_chk = !red_led_chk;  
    OzOled.init();  
    break;  
  case '5' : blue_led_chk = !blue_led_chk;  
    OzOled.init();  
    break;  
  case char(10):  
    break;  
  default:  
    Serial.println("1~5까지 입력하세요");  
    break;  
}
```





동작이 원활하게 되나요?



여러분의 관찰 결과를 말해 볼까요?





millis() 함수를 활용한 주기적인 동작제어



- delay() 함수를 활용한 동작 제어는 위의 문제 상황을 해결하기엔 역부족
- time 함수인 millis() 함수를 활용
 - 현재 모듈(프로그램)을 실행하기 시작한 이후 경과된 밀리초의 수를 반환하며, 약 50일 이후에 오버플로우로 인해 0으로 리셋
 - 데이터 유형 : unsigned long 형



millis() 함수를 활용한 알고리즘

이전 시간 저장용 변수 선언 = 0;
(type unsigned long)

반복 시간 설정 = 2000;
(type : long)

현재 경과시간 변수 = millis()
(type unsigned long)

현재경과시간-이전시간>반복시간

이전 시간 = 현재 경과 시간

반복할 동작



- 이전 시간 저장용 변수나, 현재 경과 시간 변수는 각 모듈 또는 제어할 대상별로 각각 선언해 주어야 합니다.

loop() 함수





빨강 LED를 2초 간격으로 깜빡이도록 해 볼까요?



- 이전 시간 저장용 변수 : `red_led_previousMillis = 0;`
- 현재 경과 시간 변수 : `currentMillis = millis();`
- 빨강 LED용 반복 시간 : `red_led_interval = 2000;`
- 위 변수명은 의미를 위한 표현이지, 반드시 위와 같이 표시해야 하는 것은 아닙니다.



여러분이 먼저 수정해 볼까요?





빨강 LED를 2초 간격으로 깜빡이도록 해 볼까요?



- 이전 시간 저장용 변수 : `red_led_previousMillis = 0;`
- 현재 경과 시간 변수 : `currentMillis = millis();`
- 빨강 LED용 반복 시간 : `red_led_interval = 2000;`
- 위 변수명은 의미를 위한 표현이지, 반드시 위와 같이 표시해야 하는 것은 아닙니다.

```
#define RED_PIN 8
```

```
unsigned long red_led_previousMillis = 0;  
long red_led_interval = 2000;
```

```
void Red_setup(){  
  pinMode(RED_PIN, OUTPUT);  
}
```

```
void Red_loop(){  
  
  unsigned long currentMillis = millis();  
  
  if(currentMillis - red_led_previousMillis >= red_led_interval)  
  {  
    red_led_previousMillis = currentMillis;  
    digitalWrite(RED_PIN, digitalRead(RED_PIN)^1);  
  }  
}
```





파랑 LED를 2초 간격으로 깜빡이도록 해 볼까요?



- 이전 시간 저장용 변수 : `blue_led_previousMillis = 0;`
- 현재 경과 시간 변수 : `currentMillis = millis();`
- 빨강 LED용 반복 시간 : `blue_led_interval = 50;`
- 위 변수명은 의미를 위한 표현이지, 반드시 위와 같이 표시해야 하는 것은 아닙니다.



여러분이 먼저 수정해 볼까요?





LED 동작이 서로의 반복 주기에 따라 동작하나요?
그렇다면, 멜로디를 재생시키면 어떻게 되나요?





buzzer 의 반복주기를 millis()로 제어한다면?

```
unsigned long buz_previousMillis = 0;  
long buz_interval = 5000;
```

```
int melody[] = {  
  NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4  
};
```

```
// note durations: 4 = quarter note, 8 = eighth note, etc.:  
int noteDurations[] = {  
  4, 8, 8, 4, 4, 4, 4, 4  
};
```

```
void Buzzer_setup() {  
  
}
```

```
void Buzzer_loop() {
```

```
  unsigned long currentMillis = millis();  
  if ( currentMillis - buz_previousMillis >= buz_interval ) {  
    buz_previousMillis = currentMillis;  
    melody_call();
```

```
  }  
}
```

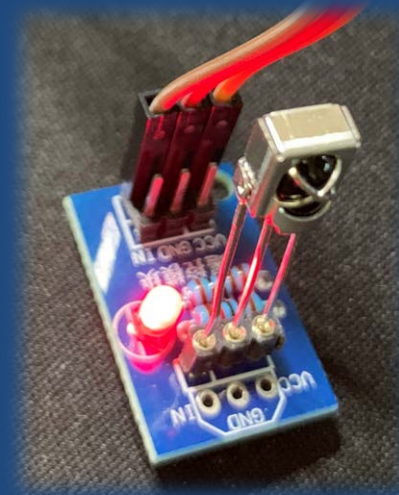




적외선 리모콘/수신기를 연결해 볼까요?



- 적외선 신호를 만들어 방출하는 포토트랜지스터와 적외선 신호를 수신하는 수광다이오드가 한쌍으로 구성됨

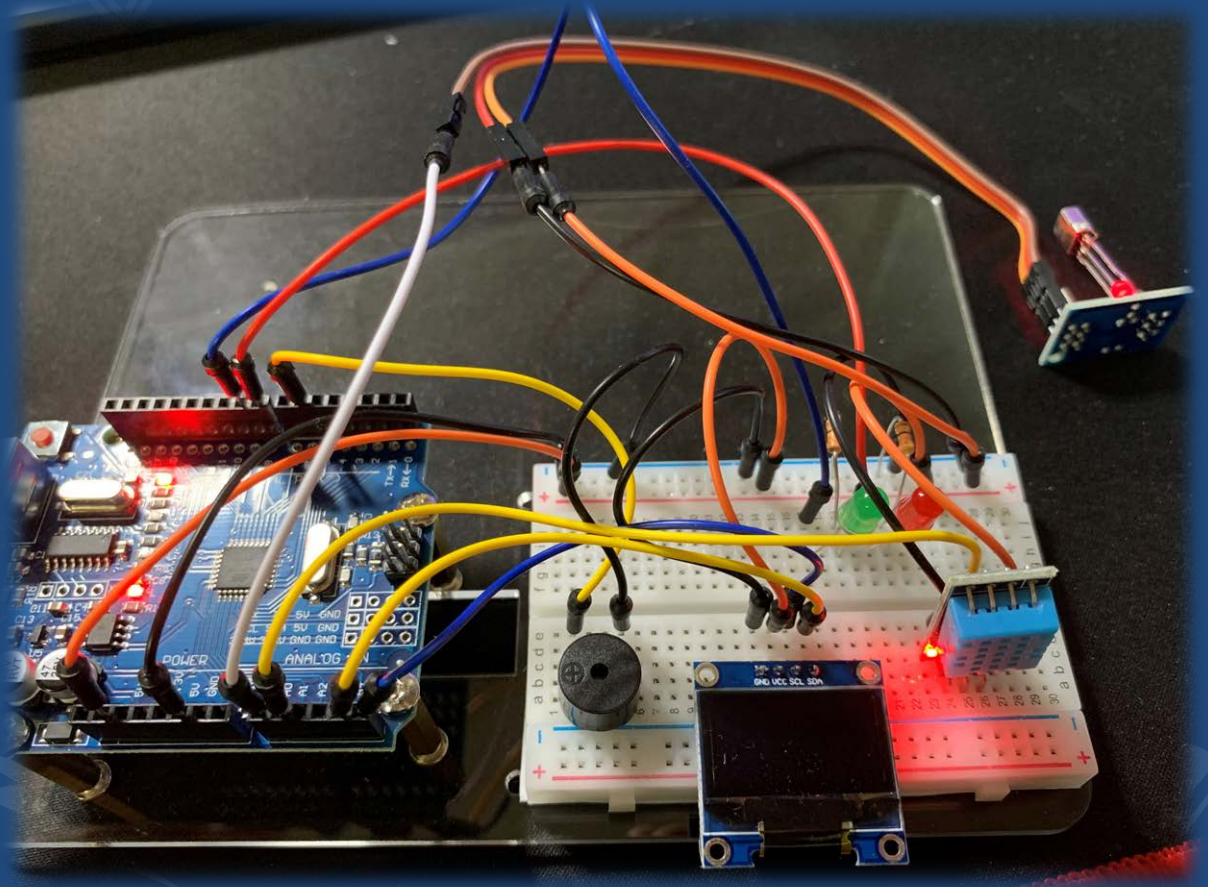




적외선 리모콘/수신기를 연결해 볼까요?



- VCC : 5v
- GND : GND
- SIG : A0





리모콘 키에 따른 16진수값



- A0 에서 읽어 들인 값이, 0xFF6897 이라면, 이 버튼을 눌렀음을 의미합니다.
- **주의 : 테스트 해 보니, buzzer 탭의 tone () 함수와 notone () 함수가 충돌이 되어, 예러가 납니다. 이 2개의 함수를 주석 처리하겠습니다.**



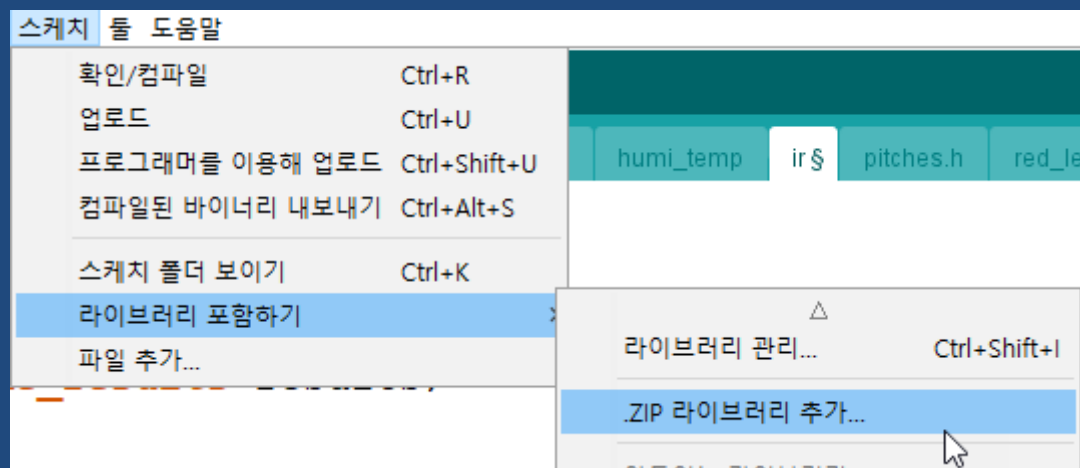
```
IRvalueData irData[21] =
```

```
{  
  { "0", 0xFF6897 },  
  { "1", 0xFF30CF },  
  { "2", 0xFF18E7 },  
  { "3", 0xFF7A85 },  
  { "4", 0xFF10EF },  
  { "5", 0xFF38C7 },  
  { "6", 0xFF5AA5 },  
  { "7", 0xFF42BD },  
  { "8", 0xFF4AB5 },  
  { "9", 0xFF52AD },  
  { "10+", 0xFF9867 },  
  { "200+", 0xFFB04F },  
  { "-", 0xFFE01F },  
  { "+", 0xFFA857 },  
  { "EQ", 0xFF906F },  
  { "<<", 0xFF22DD },  
  { ">>", 0xFF02FD },  
  { ">|", 0xFFC23D },  
  { "CH-", 0xFFA25D },  
  { "CH", 0xFF629D },  
  { "CH+", 0xFFE21D }  
};
```



리모콘 키 인식하기

1. 먼저, zip 라이브러리 추가 - IR_library.zip



2. 헤더 파일 추가 - 라이브러리 포함하기 - '<Irremote.h>'





리모콘 키 인식하기

3. 새 탭 : ir

```
int RECV_PIN = A0;
```

```
IRrecv irrecv(RECV_PIN);  
decode_results results;
```

```
void ir_setup(){  
  irrecv.enableIRIn();  
}
```

```
void ir_loop(){  
  if(irrecv.decode(&results)){  
    input_check();  
    irrecv.resume(); //다음 값을 입력 받는다.  
  }  
}
```



- base.ino 파일의 setup() 과 loop() 함수에 ir_setup() 함수와 ir_loop() 함수를 각각 추가합니다.



```

void input_check(){
    switch (results.value) { // 리모콘 버튼의 고유값에 따라
    case 0xFF6897 : // 리모콘의 0버튼이 눌리면, 키 값에 대한 정보만 표시하고, LED 모두 끈다.
        // LED 가 동작하고 있으면 의미 없음.
        OzOled.init();
        analogWrite(9, 0);
        digitalWrite(8, LOW);
        LED_OLED(0, 0xFF6897);
        break;
    case 0xFF30CF : // 리모콘의 1버튼이 눌리면,
        LED_OLED(1, 0xFF30CF);
        led_humi_temp = !led_humi_temp;
        break;
    case 0xFF10EF : // 리모콘의 4버튼이 눌리면,
        LED_OLED(4, 0xFF10EF);
        break;
    case 0xFF42BD : // 리모콘의 7버튼이 눌리면,
        LED_OLED(7, 0xFF42BD);
        break;
    }
}

```



- 번호 키에 대한 동작
- 0번은 LED 모두 끄고, 0zoled 온습도 표시 지움
- 1번은 온습도 표시
- 2번은 빨강 LED 동작
- 3번은 파랑 LED 동작
- 각 버튼의 동작은 누를 때마다 동작이 토글됨.




```
case 0xFF18E7: // 리모콘의 2버튼이 눌리면,  
    LED_OLED(2, 0xFF18E7);  
    red_led_chk = !red_led_chk;  
    break;  
case 0xFF38C7: // 리모콘의 5버튼이 눌리면,  
    LED_OLED(5, 0xFF38C7);  
    break;  
case 0xFF4AB5: // 리모콘의 8버튼이 눌리면,  
    LED_OLED(8, 0xFF4AB5);  
    break;  
case 0xFF7A85: // 리모콘의 3버튼이 눌리면,  
    LED_OLED(3, 0xFF7A85);  
    blue_led_chk = !blue_led_chk;  
    break;  
case 0xFF5AA5: // 리모콘의 6버튼이 눌리면,  
    LED_OLED(6, 0xFF5AA5);  
    break;  
case 0xFF52AD: // 리모콘의 9버튼이 눌리면,  
    LED_OLED(9, 0xFF52AD);  
    break;
```



```
void LED_OLED(long num, long button){  
    OzOled.printNumber(num, 0, 5);  
    OzOled.printString(" bun : ", 1, 5);  
    OzOled.printNumber(button, 7, 5);  
}
```



- 몇 번 버튼을 눌렀는지, Ozoled 에 표시하기 위한 함수
- 습도/온도 표시와 표시 위치가 중복될까봐 6번째 행에 표시





여러분들의 상상을 바탕으로 한 결과물 만들기



- 지금까지 배우신 것을 토대로 무언가로 탈바꿈시켜볼까요?
- 하드웨어는 그대로 두고, 그 하드웨어를 어떻게 어떤 모습으로 사용하지 (소프트웨어)를 고민해서 그 결과물을 만들어 보아요!

