

교과목 포트폴리오 발표

01

소개

- 자기소개
- 프로젝트 소개

02

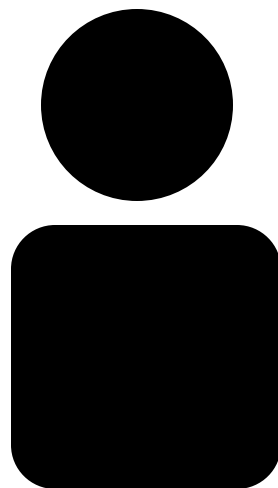
학습내용

- 안드로이드
- 블루투스
- 깃과 깃허브

03

후기

- 후기



이름

정진교

학과

컴퓨터 정보 공학과

수행내용

팀장 / 아두이노 / 기획 및 프로젝트 제작 관리

학습내용(팀프로젝트)



화염감지 센서



1. 불꽃 감지 센서(Flame Sensor)란 적외선 LED를 통해 화재 시 연소반응에 의해 불꽃에서 파생되는 열 복사인 적외선 파장을 감지하여 아날로그 또는 디지털 신호를 수신하는 센서
2. 스파크성 불꽃이나 근접 거리에서의 발화체 감시 및 점화 확인 가능
3. 최대 감지 거리는 약 17~18cm

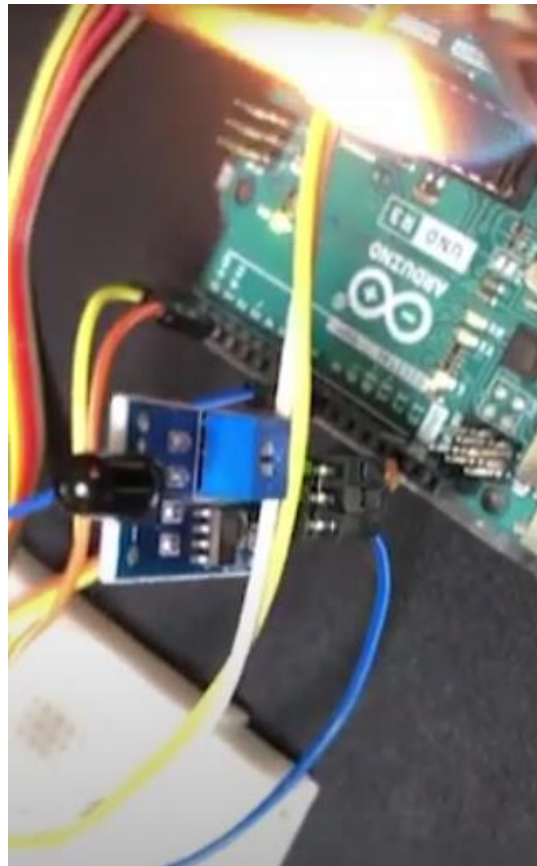
화염감지 센서

```
int led = 13;
int flame = 7;
int state = 0;

void setup() {
  pinMode(led, OUTPUT);
  pinMode(flame, INPUT);
  Serial.begin(9600);
}

void loop() {
  state = digitalRead(flame);
  digitalWrite(led, LOW);

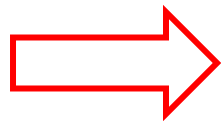
  if(state == 0){
    Serial.println("ON");
    delay(100);
  }
  else{
    Serial.println("OFF");
    digitalWrite(led, LOW);
    delay(100);
  }
  delay(100);
}
```



LED ON!!

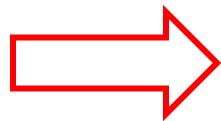
화염감지 센서

```
int led = 13;  
int flame = 7;  
int state = 0;
```



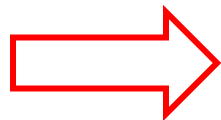
아두이노 핀 번호 설정

```
void setup() {  
  pinMode(led, OUTPUT);  
  pinMode(flame, INPUT);  
  Serial.begin(9600);  
}
```



LED는 출력으로 flame은 입력으로 설정
및 시리얼 통신 실행

```
void loop() {  
  state = digitalRead(flame);  
  digitalWrite(led, LOW);  
  
  if(state == 0){  
    Serial.println("ON");  
    delay(100);  
  }  
  else{  
    Serial.println("OFF");  
    digitalWrite(led, LOW);  
    delay(100);  
  }  
  delay(100);  
}
```



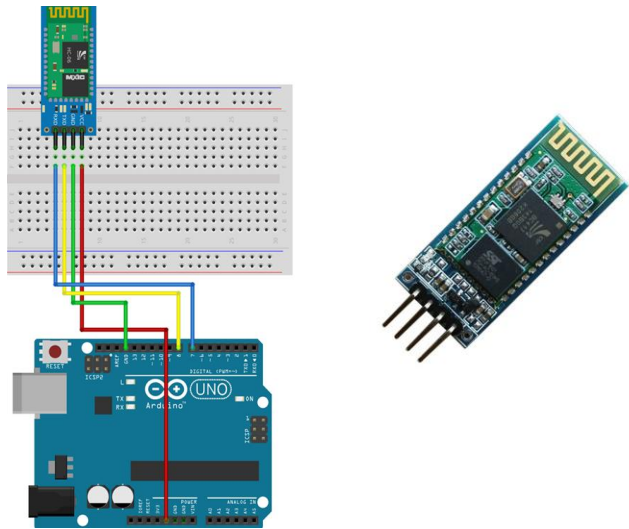
State를 통해 flame 센서 값 확인

If문을 이용해 화염이 감지되면 시리얼 모니터에 ON 출력

아니라면 (else) OFF 출력

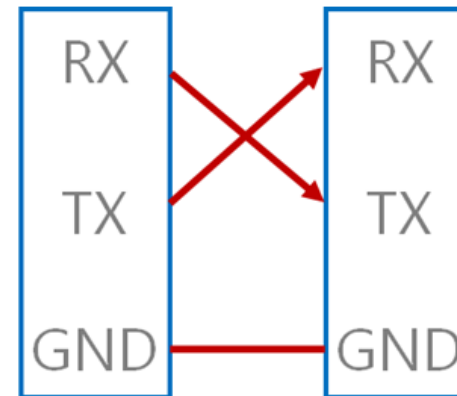
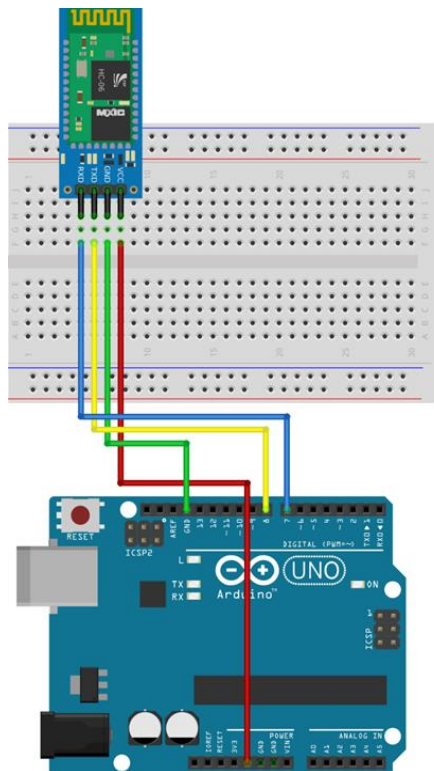
```
}|
```

블루투스 센서



1. 블루투스는 1994년에 최초로 개발 된 근거리 무선 통신을 위한 산업 표준
2. 프로젝트에 사용한 HC-06 모델은 아두이노에서 시리얼 통신을 이용하여 데이터 값을 주고 받을 수 있는 모듈이다.
3. 10M거리에서 무선 통신 이용 가능(아이폰에서는 사용 불가)
4. 실생활에서는 블루투스 이어폰과 같이 무선 전자기기 통신을 위해 주로 사용한다.

블루투스 센서



```
1 #include <SoftwareSerial.h>
2
3 #define BT_RXD 8
4 #define BT_TXD 7
5 SoftwareSerial bluetooth(BT_RXD, BT_TXD);
```

- ▶ TXD, RXD 연결시 아두이노와 블루투스 모듈을 반대로 연결합니다.
즉, 코드에서 입력한 RX, TX 핀번호는 아두이노의 핀번호이며,
아두이노의 RX핀은 블루투스 모듈의 TX에 연결하고,
아두이노의 TX핀은 블루투스 모듈의 RX에 연결합니다..

RX TX는 송신신호선과 수신신호선이 분리되어있고
그런신호를 입출력하는 포트에는 수신신호가 들어오는곳이 Rx
가 되고 송신신호선이 나가는곳이 Tx

Rx는 Receive 의 R
Tx는 Transmitt의 T
으로 송수신을 담당한다

블루투스 센서

블루투스 연결 확인

```
#include <SoftwareSerial.h>

SoftwareSerial bt(3,2);

void setup() {
  Serial.begin(9600);
  bluetooth.begin(9600);
}

void loop() {
  if(bt.available()) {
    Serial.write(bt.read());
  }
  if(Serial.available()) {
    bt.write(Serial.read());
  }
}
```

시리얼 통신



블루투스 연결 확인

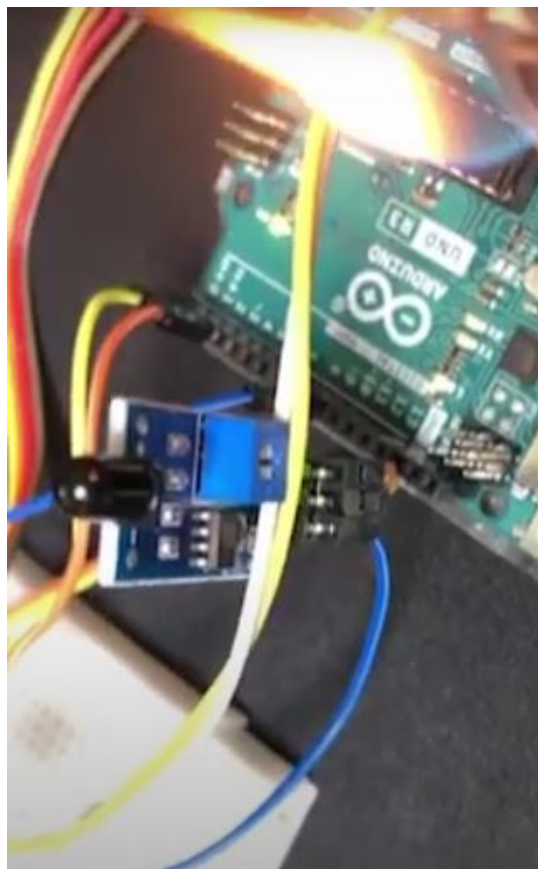
아두이노 전체 코드

```
#include <SoftwareSerial.h>

SoftwareSerial bt(3, 2); //Tx,Rx 핀번호 설정
int flame = 10;          //화염감지센서 핀번호
int co = 1;
void setup(){
  pinMode(flame,INPUT);
  Serial.begin(9600);     // 아두이노 <> PC 통신
  bt.begin(9600);         // 블루투스 <> 아두이노 통신
}
void loop(){
  if(bt.available()){     // 블루투스에서 데이터가 들어오면
    Serial.write(bt.read()); // 읽은데이터를 PC로 전송
  }
  if(Serial.available()){ // PC에서 데이터가 들어오면
    bt.write(Serial.read()); // 읽은데이터를 블루투스로 전송
  }
}
```

```
if(digitalRead(flame) == 0){ // 불이 감지되면
  bt.print(" fire!!!"); // 안드로이드로 센서값 출력
  delay(10000);
  co=0;
}
if(co==0){
  bt.print(" no fire");
  co=1;
}
}
```

아두이노 전체 코드



목차



깃

- 분산 버전 관리 시스템
- 리누스 토발스가 2005년 개발

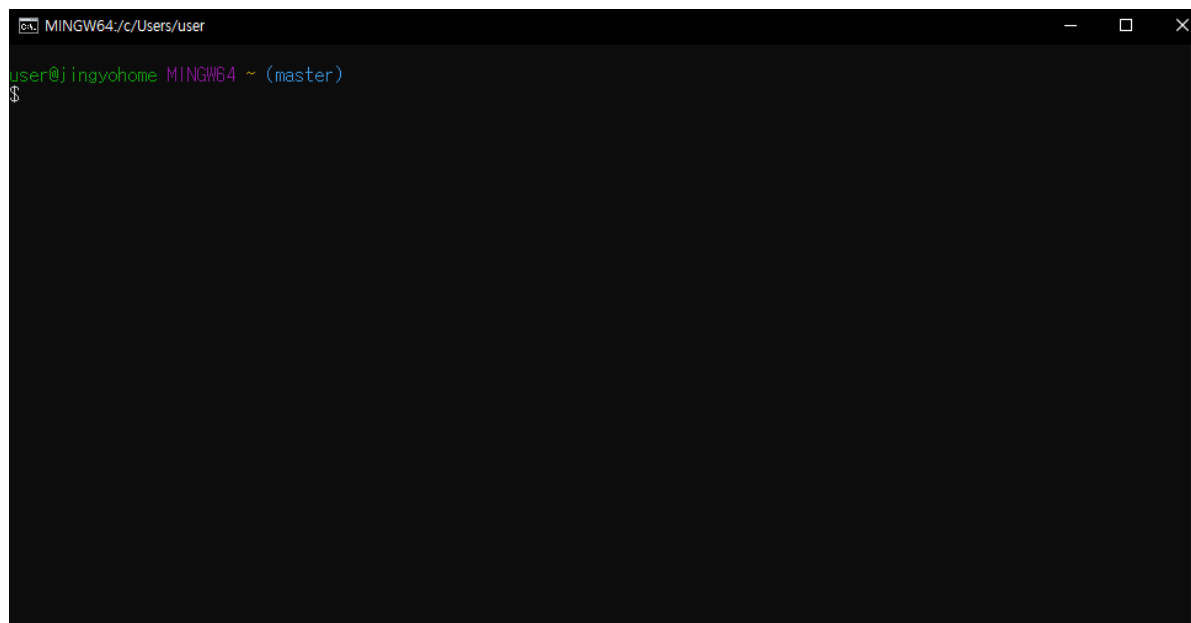
깃허브

- 깃의 저장소 및 관리 서비스

깃 git = 분산 버전 제어 SW

깃 github = git을 위한 웹 저장소(Repository)
+ 커뮤니티 협의 공간

깃과 깃허브는 연결되어 있지만 다름



```
MINGW64: c:/Users/user
user@jinyohome MINGW64 ~ (master)
$
```

Git bash 실행

git add commit log

-> Git bash 초기 설정

도움말 보기

\$ git help

버전 보기

\$ git --version

사용자 설정

\$ git config --global user.name "jjk"

\$ git config--globaluser.email 1213@gmail.com

설정 확인

\$ git config --list

로컬 저장소

저장소 Git repository

파일이나 폴더를 저장해 두는 곳

파일이 변경 이력 별로 구분되어 저장

비슷한 파일이라도 실제 내용 일부 문구가 서로 다르
면 다른 파일로 인식

파일을 변경사항 별로 구분해 저장

깃

\$ git init 프로젝트 name

- 레포지토리를 만들기 위해 사용

디렉토리를 git repository로 만들어야 git으로 버전 관리가 가능

\$ git clone [url]

마스터 저장소

탐색기 확인

.git

\$ git status - 저장소 만든 후 초기 , 현 상황 확인

파일 2개 git-cli.txt, git-study.txt 만든 후

commit -> 실질적인 저장소에 파일 저장

Clear , Git bash 초기화

\$git add 폴더 생성

\$git commit 새 버전 업로드

\$git list : commit 목록 확인

명령어 \$ git config --list

```
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
core.editor="C:\\Program Files (x86)\\Notepad++\\notepad++.exe" -multiInst -notabbar -nosession -noPlugin
pull.rebase=false
credential.helper=manager-core
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
gui.recentrepo=C:/Users/user/Git practice
gui.recentrepo=C:/Users
user.name=tkdgksrk
user.email=wjdwlsry1213@gmail.com
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
core.repositoryformatversion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
core.symlinks=false
core.ignorecase=true
gui.wmstate=normal
gui.geometry=824x435+650+182 179 196
(END)
```

간혹 \$ git config --list를 했을때 end가 뜨면서
입력이 되지 않을 때가 있는데 이때는 quit을 뜻
하는 알파벳 q를 입력하면 다시 깃 명령어를 입
력할 수 있다.

후기

프로젝트 수행에 있어 계획만큼 진행 한다는 사실이 꽤 힘들다는 사실을 알았습니다.

기존에 기능을 개발하던 도중 개발 기술의 부족(와이파이 쉘드)으로 인해 프로젝트 기획서를 한번 전면 수정하게 되면서 시간이 부족하게 되었지만 최초 기획에서 넣고자 하려했던 기능들을 구현하고자 했고 또 완성도가 마음에 들지 않더라도 기능을 공부함에 있어선 부족하지 않았다고 생각합니다.

이번 프로젝트에 참여하며 개발 진행에서의 순서나 여러 방법들을 공부하며 추후 직접 공부하게 될 것들 혹은 취업이후 개발을 진행할때에 도움이 될 것 이라고 생각합니다.

학우 여러분들 올 한해 고생하셨습니다

기말고사 잘 마무리 하시길 바랍니다.