

Chap01. 사물인터넷과 NodeMCU

Contents

1.1 사물인터넷

1.2 NodeMCU

1.3 NodeMCU 아두이노 IDE 환경 설정

1.4 Blynk 환경 설정

1.5 시리얼 통신

1.1 사물인터넷

◆ 사물인터넷(IoT : Internet of Things)

- 다양한 사물들(things)이 연결된 네트워크
- 현실세계(Physical World)에 존재하는 사물들과 가상세계(Cyber World)에 존재하는 객체가 상호 연결되어 동작하는 것

1.1 사물인터넷

◆ 사물

- 컴퓨팅 기기(컴퓨터, 스마트폰 등)
- 가전제품(TV, 냉장고, 에어컨 등)
- 일반 사물(안경, 핸드백, 현관문 등)

1.1 사물인터넷

◆ 사물들의 인터넷에 연결

- 사물인터넷에서 가장 기본
- 자신이 생성한 데이터 공유
- 다른 사물이 생성한 데이터 이용
- 어떤 사물의 상태에 따라 다른 사물의 상태 변화시키는 것 가능

1.1 사물인터넷

◆ 사물들의 인터넷에 연결

- 이더넷이나 wifi 같은 유무선 통신 기술 이용
- 3G나 LTE 같은 셀룰러 통신 기술 이용
- 블루투스 비콘(BLE Beacon) 또는 RFID 기술 이용

1.1 사물인터넷

◆ IoT 사용 분야

- Retail, Manufacturing
- Health Care
- Transportation and Logistics
- Government
- Energy

1.1 사물인터넷

◆ IoT 주요 기술과 연동 방식

- Data management and streaming analytics
 - 필터링, 정규화, 표준화, 변환, 집계, 상관 관계 및 시간 분석 등
- Big data analytics
 - 예측 분석, 텍스트 마이닝 클라우드 컴퓨팅 데이터 마이닝
 - 데이터 레이크 및 Hadoop

1.1 사물인터넷

◆ IoT 주요 기술과 연동 방식

- Artificial intelligence

- 머신 러닝, 딥 러닝, 자연어 처리, 컴퓨터 비전 등

1.2 NodeMCU

◆ NodeMCU

- NodeMCU는 보드에 와이파이 모듈 탑재
- 사물인터넷 노드용(IoT Node) MCU 장치 개발에 적합
- ESP8266(ESP-12E)



1.2 NodeMCU

◆ NodeMCU 장점

- NodeMCU 하나를 사용하는 것이 가격도 저렴하고 부피도 적음
 - 아두이노 + WIFI통신장치
- 속도가 빠름
 - NodeMCU 내장 프로세서(80MHz대)
 - 아두이노 내장 프로세서(16MHz대)

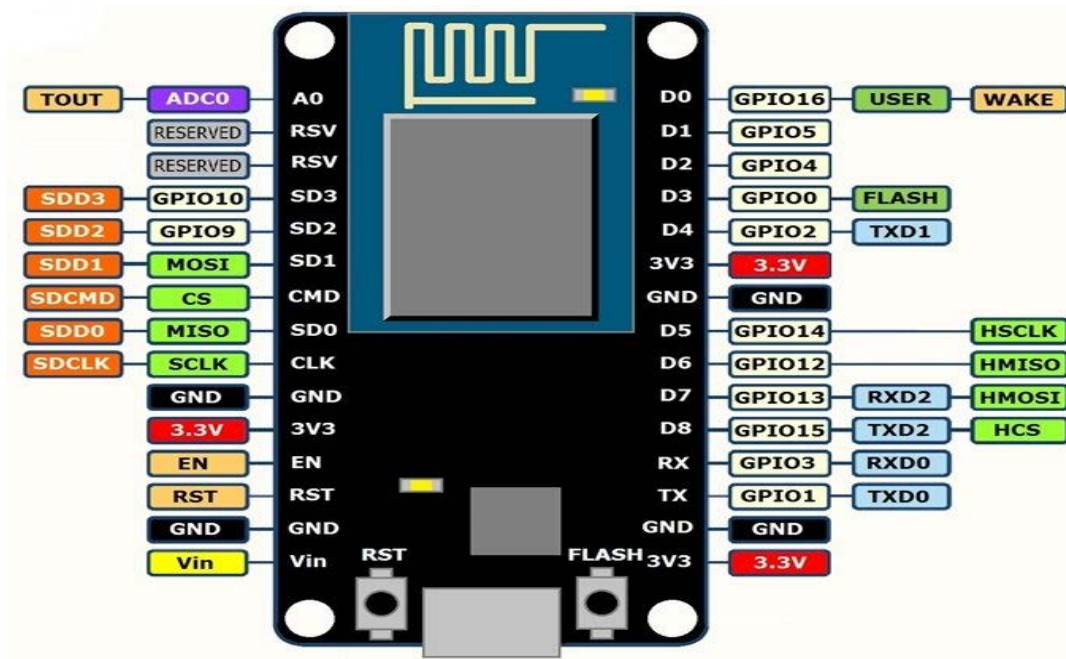
1.2 NodeMCU

◆ NodeMCU 단점

- 입출력 핀 수가 적어 여러 장치 연결 불가능
 - 아날로그 입력핀(1개)
 - 디지털 입출력핀(5~8개)
 - 아두이노 나노(아날로그 6개, 디지털 14개)
- WIFI가 가동될 때 전력소비가 큼(배터리환경 지속 곤란)
 - 절전Sleeping 모드 사용할 수 가능

1.2 NodeMCU

◆ NodeMCU 핀 구성



1.2 NodeMCU

◆ NodeMCU 핀 설명

- GPIO0,2,15(부팅모드 설정하기 위해 내정) 사용시 주의
- GPIO1,3(시리얼 통신용)
- GPIO6~11(플래시메모리용) 사용 불가능
- GPIO16(Wake용 – Sleep mode escape)

1.2 NodeMCU

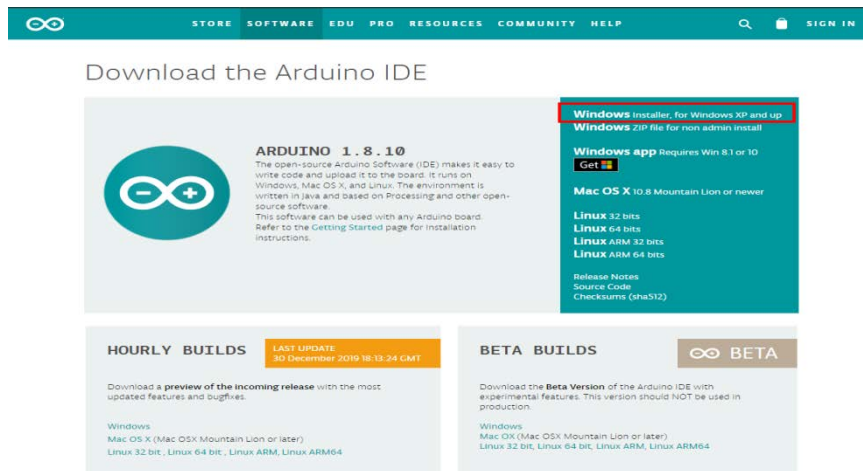
◆ NodeMCU 핀 설명

- 사용자가 조건 없이 범용으로 사용 가능한 핀
 - GPIO4(D2), GPIO5(D1), GPIO12~14(D6,D7,D5)
- GPIO2(내장LED 연결)
 - 아두이노 나노(디지털 13번)

1.3 NodeMCU 아두이노 개발 환경

◆ IDE 다운로드

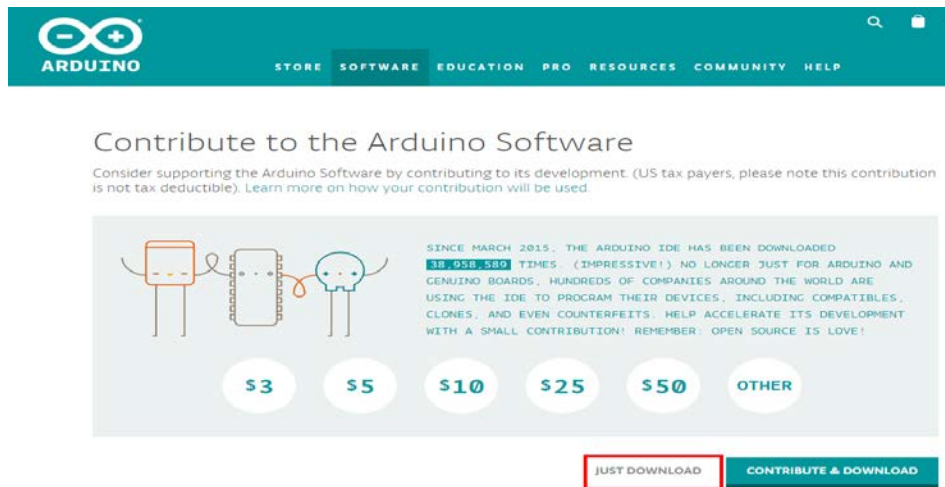
- <https://www.arduino.cc>
 - SOFTWARE → DOWNLOADS → Windows installer



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 다운로드

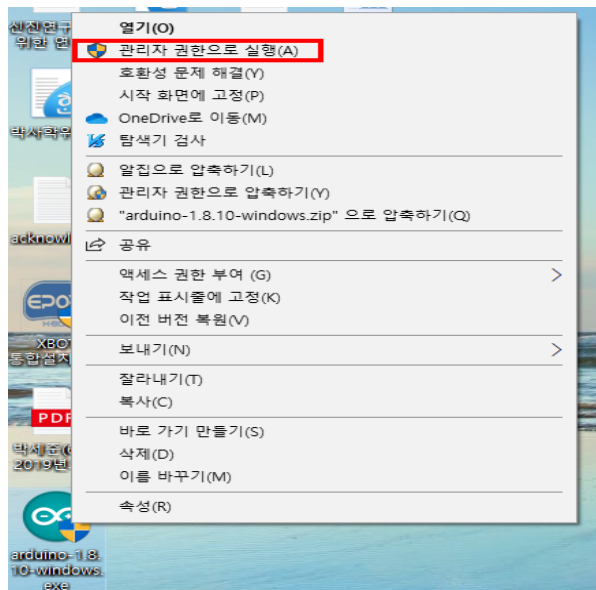
- <https://www.arduino.cc>
 - JUST DOWNLOAD



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 설치

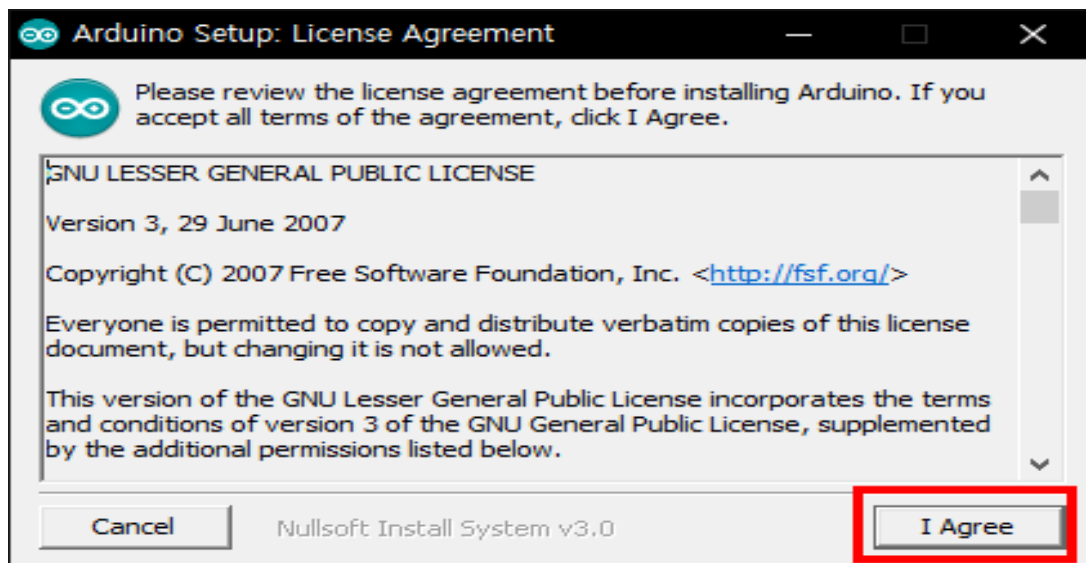
- arduino-1.8.10-windows 관리자 권한으로 실행



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 설치

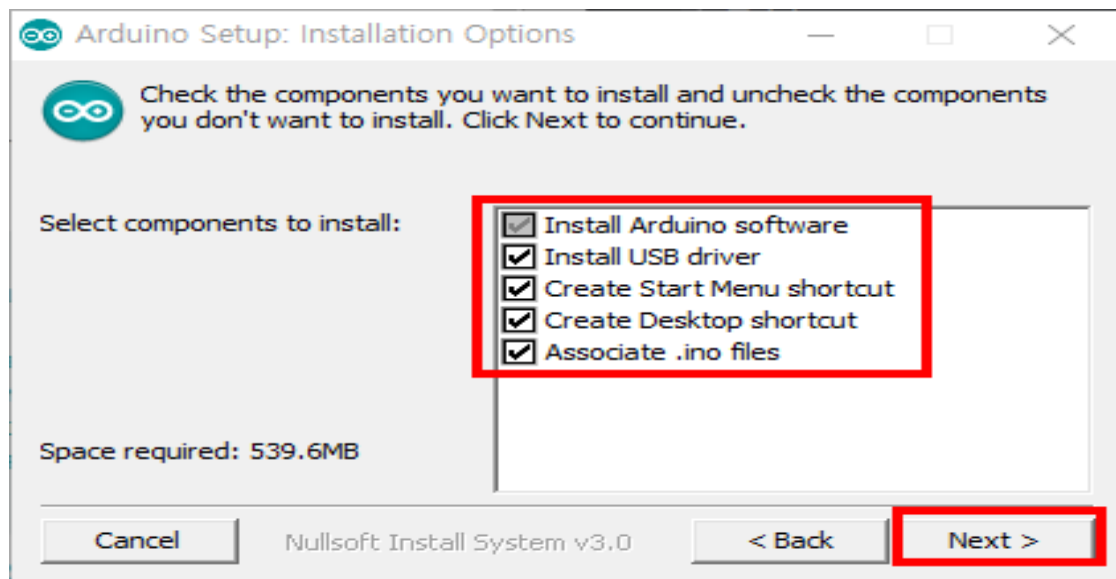
- I Agree(라이선스 동의 화면)



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 설치

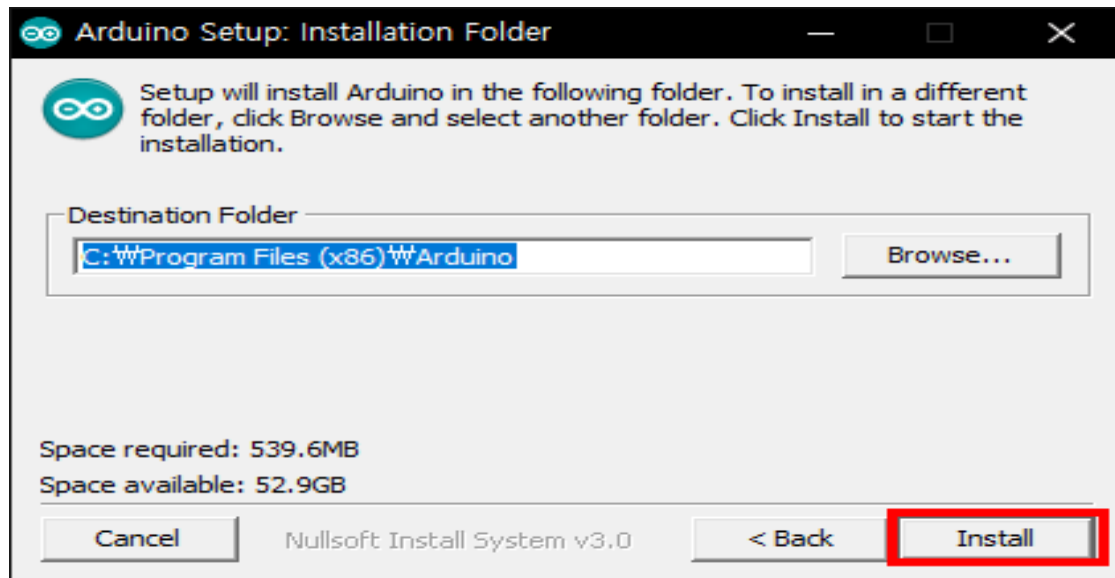
- All check(설치 옵션 선택) → Next



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 설치

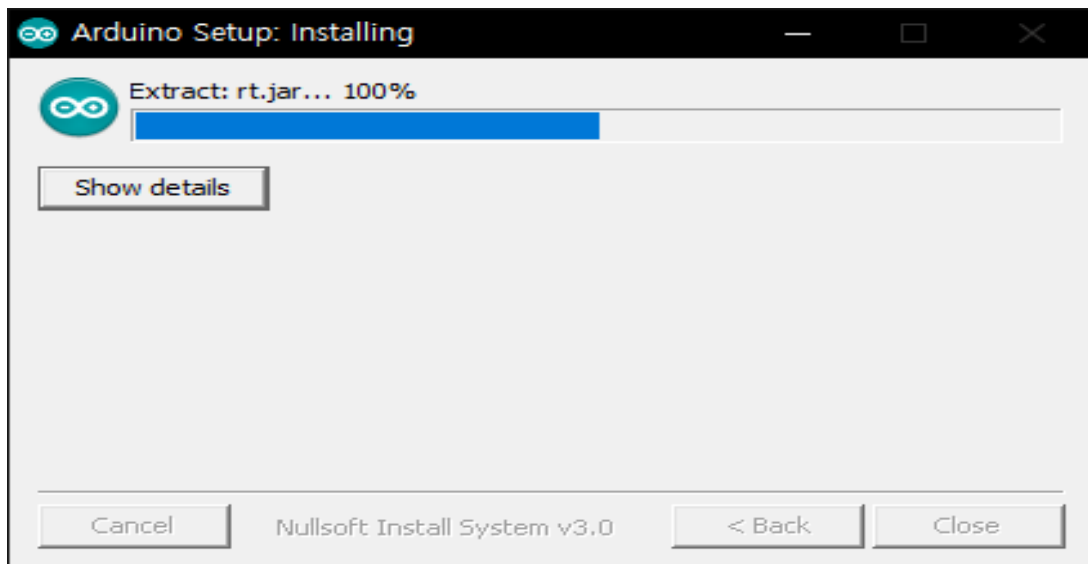
- Install(설치 장소 선택)



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 설치

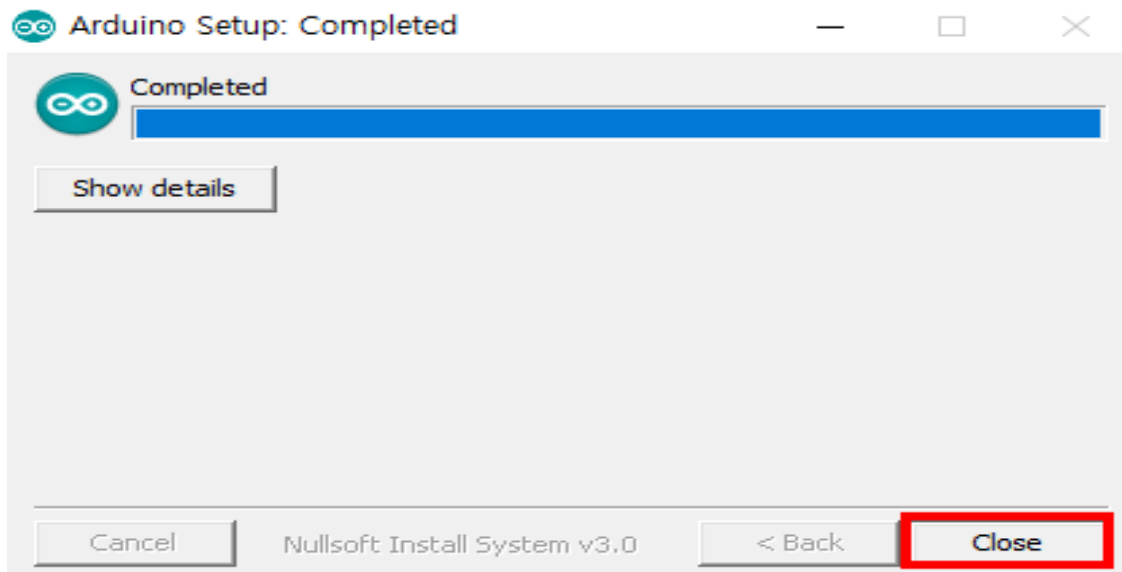
- Install(설치 중)



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 설치

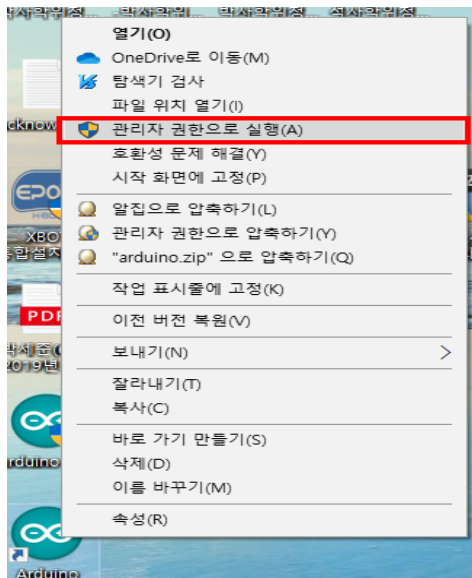
- Close(설치 완료)



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

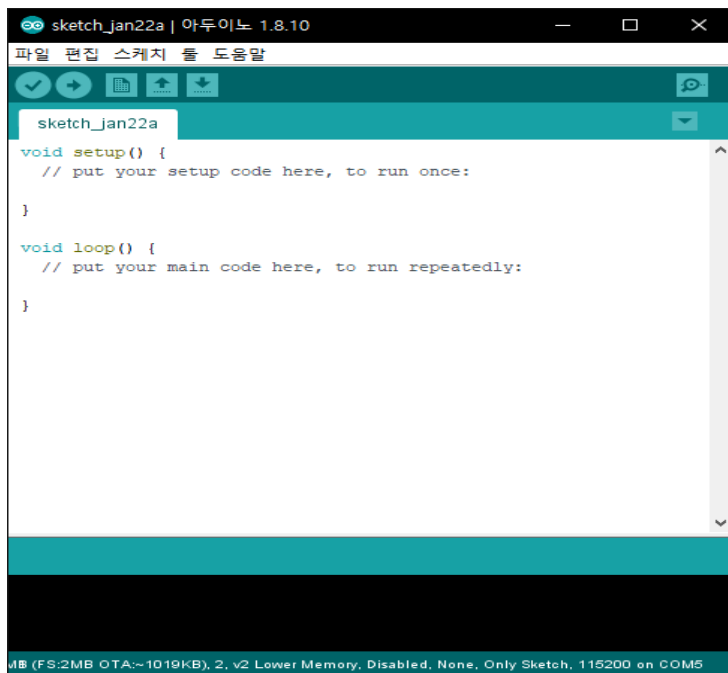
- arduino 관리자 권한으로 실행



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

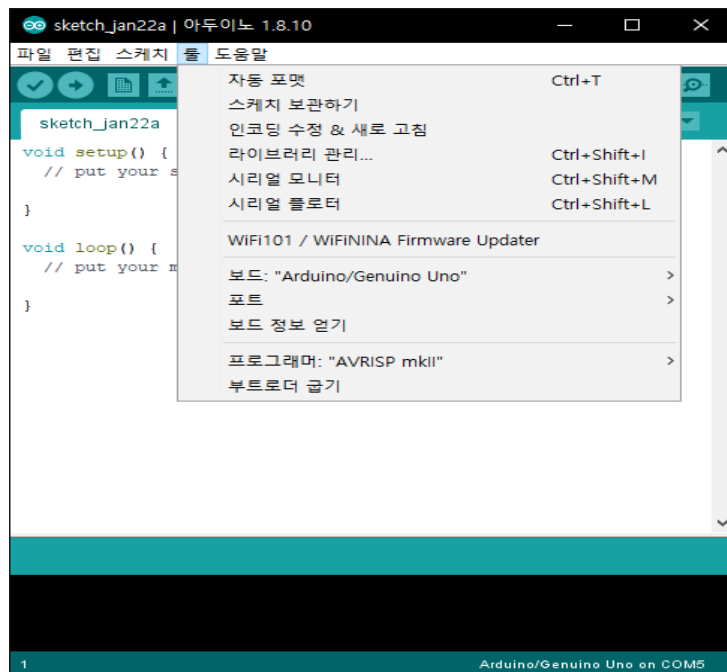
- 아두이노 IDE 에디터 화면



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

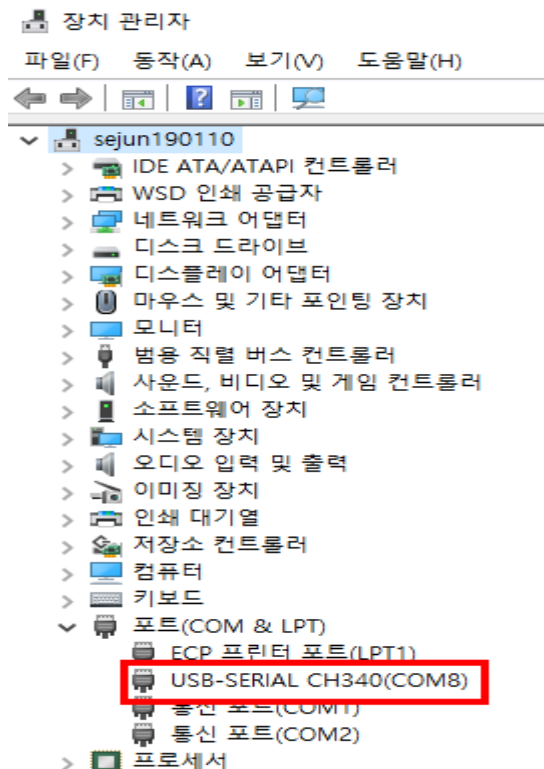
● 보드와 포트 설정



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

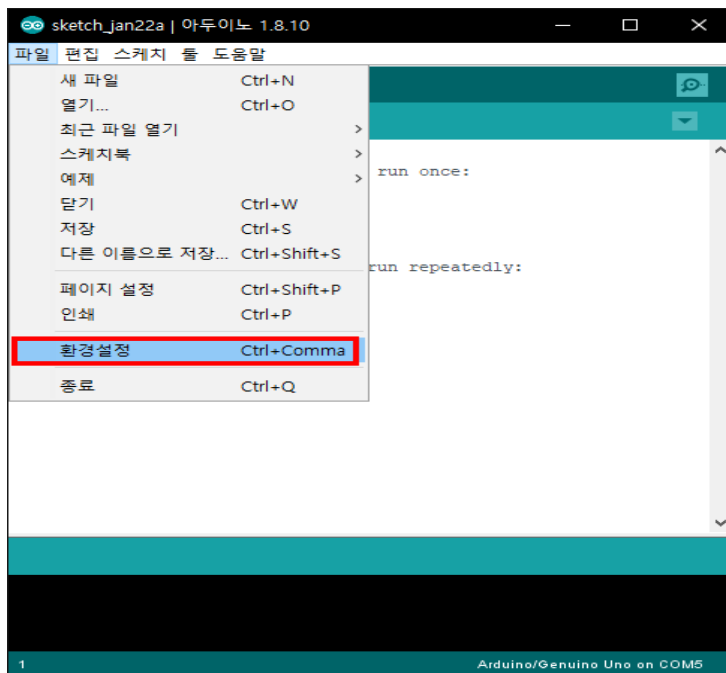
- 장치관리자에서 포트 확인



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

● 환경 설정

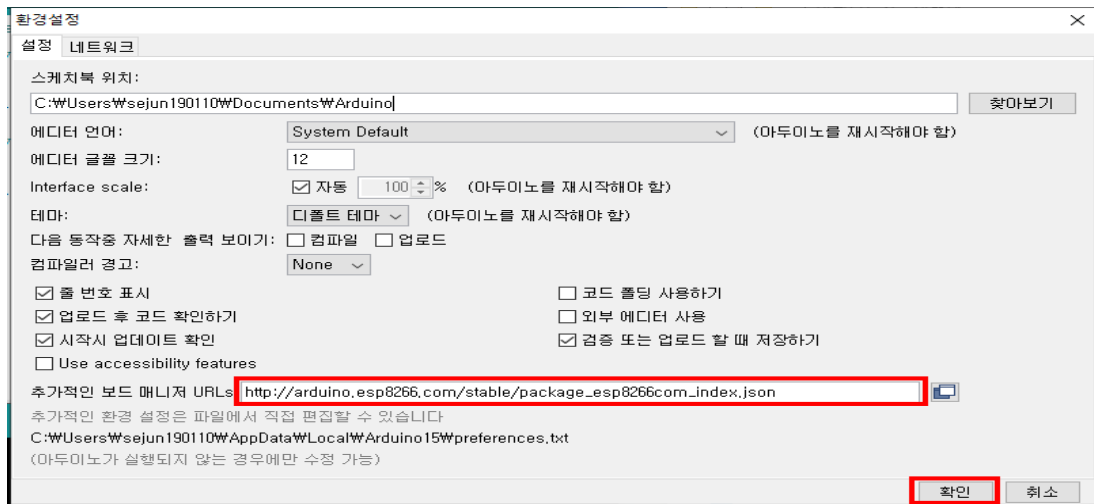


1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

- 추가적인 보드 매니저 URLs 입력

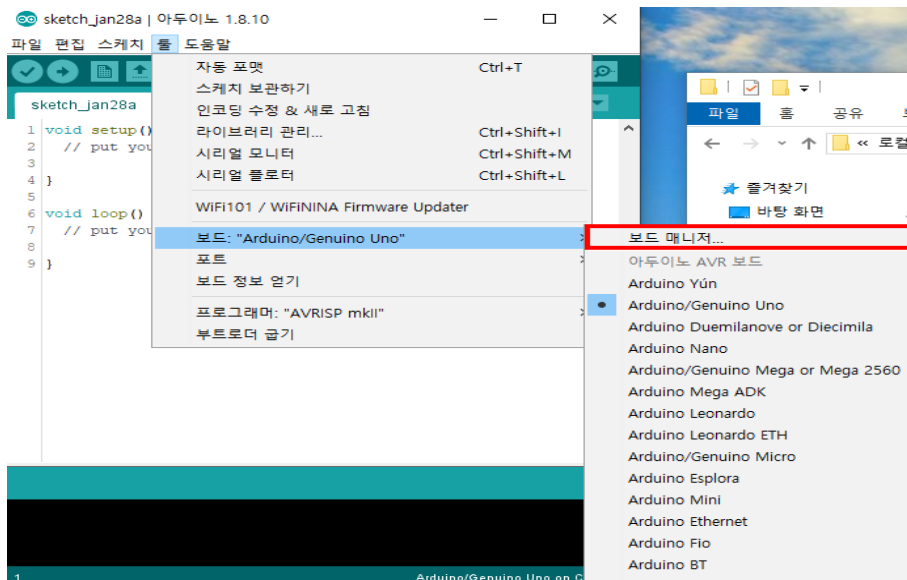
- http://arduino.esp8266.com/stable/package_esp8266com_index.json



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

- 아두이노 IDE에서 이용할 개발 툴킷 선택(보드 매니저)



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

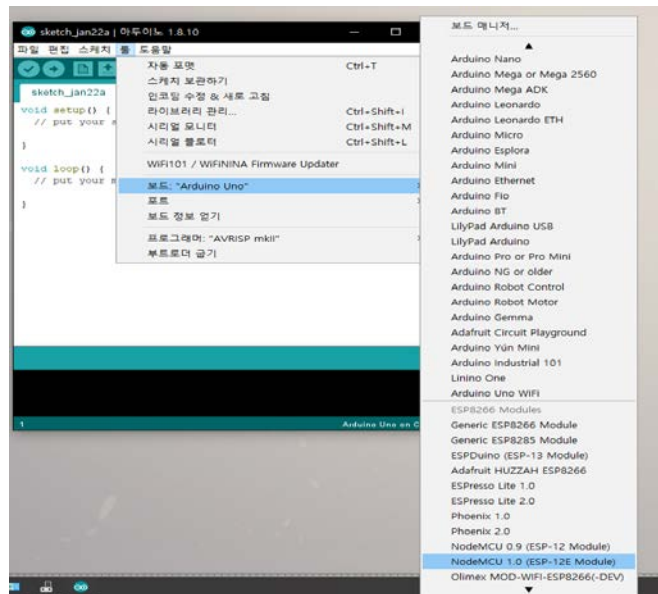
- ESP 관련 모듈 검색 후 설치



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

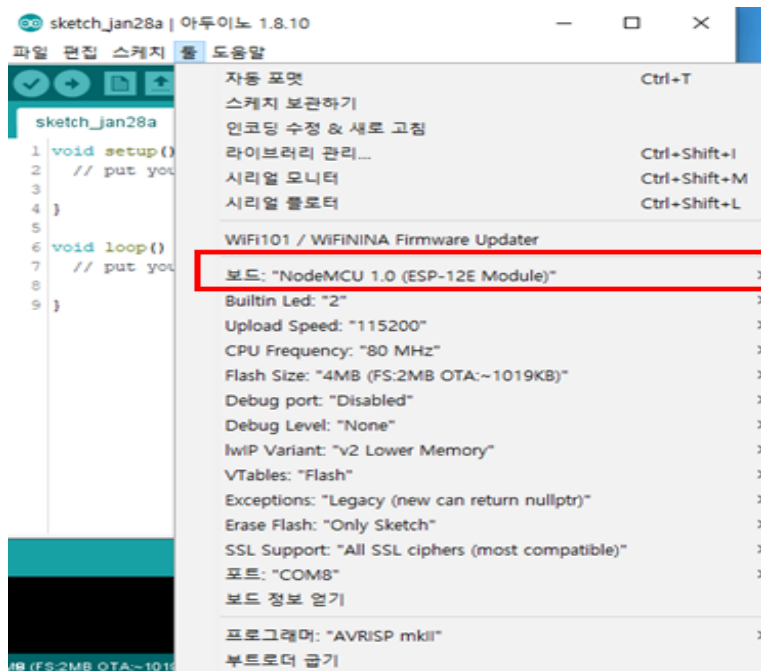
- NodeMCU 1.0(ESP-12E Module) 선택



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

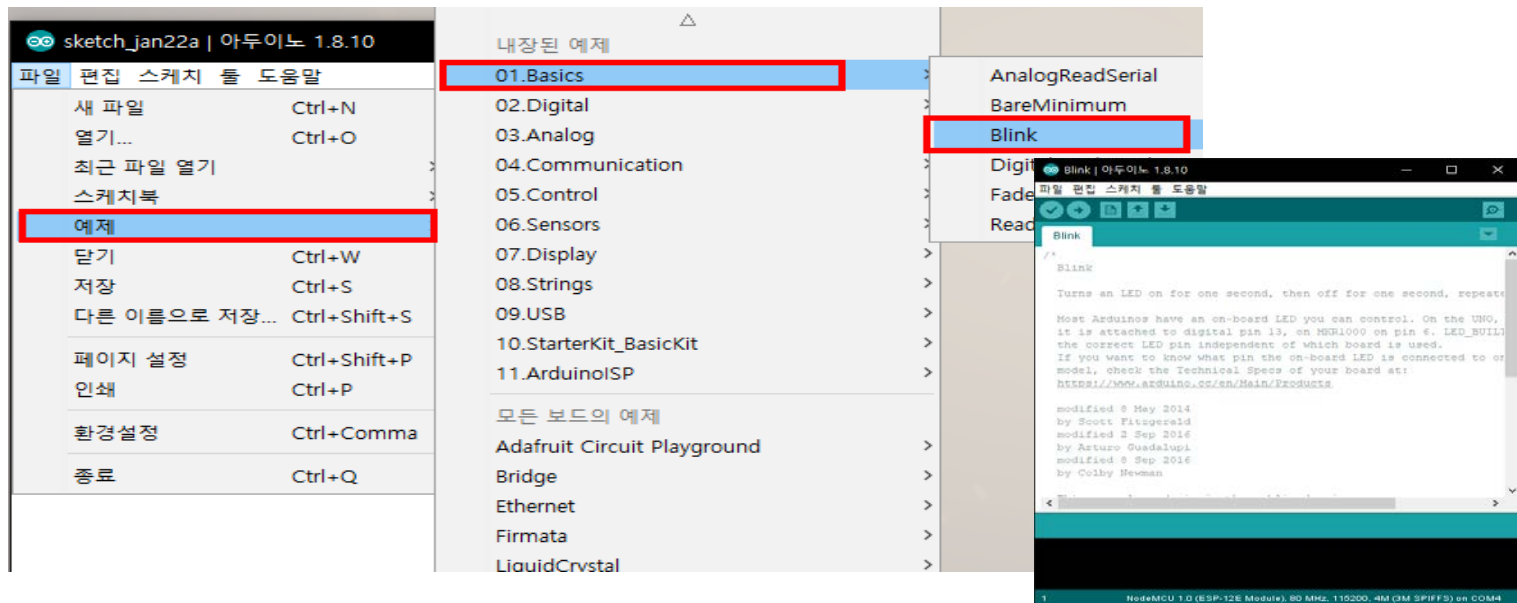
- 기본 설정이 끝난 아두이노



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

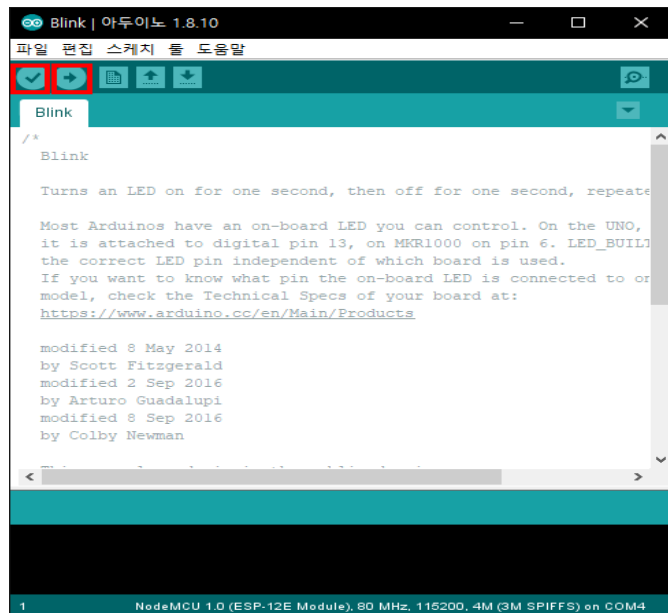
● Blink 예제 파일 불러오기



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

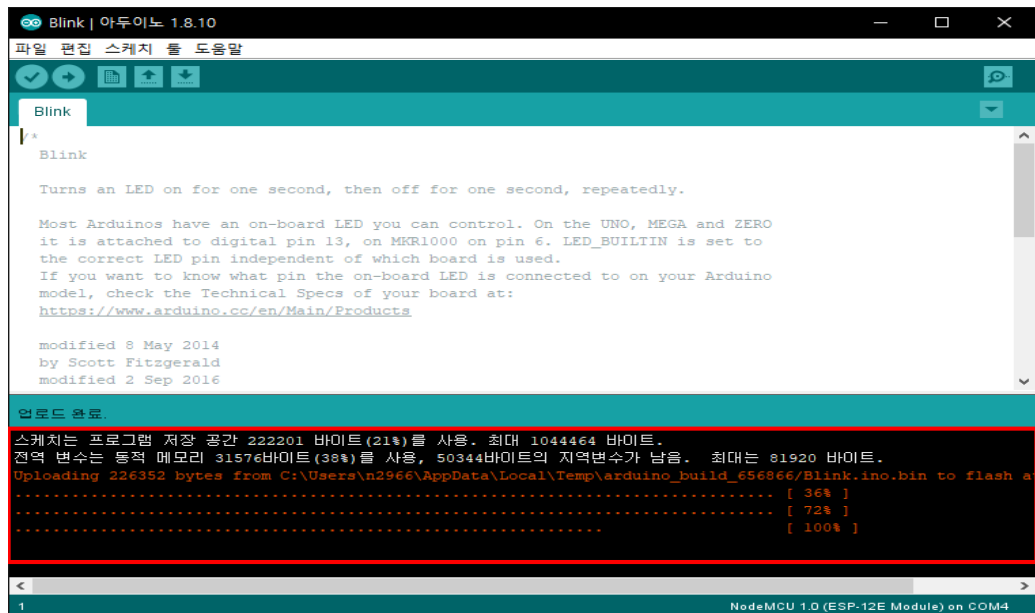
- 아두이노 IDE 컴파일(✓)과 업로드(→)



1.3 NodeMCU 아두이노 개발 환경

◆ IDE 실행

- 아두이노 IDE 상태 표시창(파란색 LED 깜박)



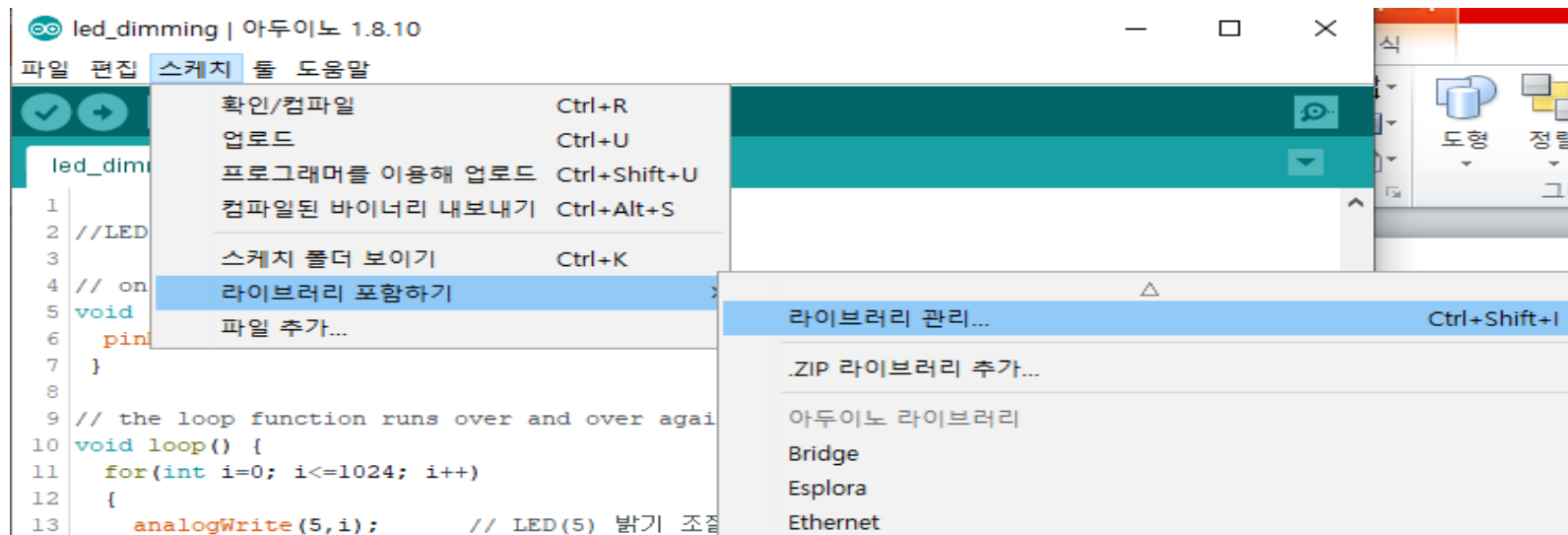
The screenshot shows the Arduino IDE interface. The main window displays the 'Blink' sketch, which includes a comment explaining the function: 'Turns an LED on for one second, then off for one second, repeatedly.' Below the comment, it states: 'Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to the correct LED pin independent of which board is used. If you want to know what pin the on-board LED is connected to on your Arduino model, check the Technical Specs of your board at: <https://www.arduino.cc/en/Main/Products>'.

At the bottom of the IDE, the status bar shows 'NodeMCU 1.0 (ESP-12E Module) on COM4'. Above the status bar, the '업로드 완료' (Upload Complete) message is displayed, indicating that the sketch was successfully uploaded to the NodeMCU board.

```
업로드 완료
스케치는 프로그램 저장 공간 222201 바이트(21%)를 사용. 최대 1044464 바이트.
전역 변수는 동적 메모리 31576바이트(38%)를 사용, 50344바이트의 지역변수가 남음. 최대는 81920 바이트.
Uploading 226352 bytes from C:\Users\n2966\AppData\Local\Temp\arduino_build_656866/Blink.ino.bin to flash at
..... [ 36% ]
..... [ 72% ]
..... [ 100% ]
```

1.4 Blynk 환경 설정

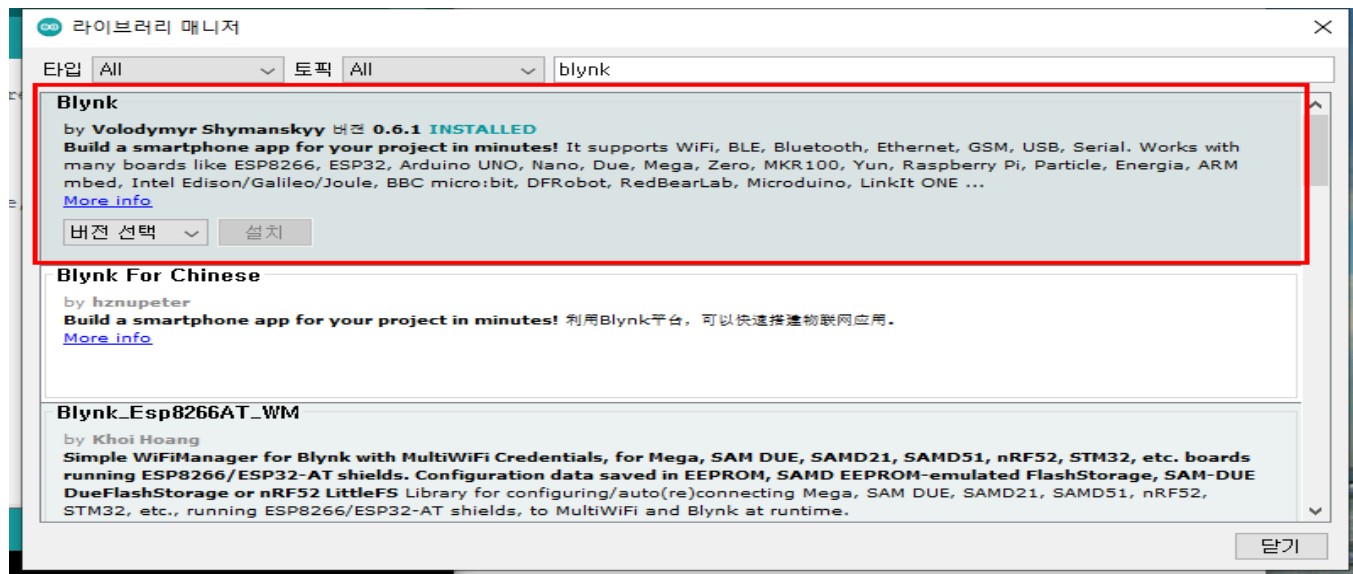
◆ Blynk 라이브러리 설치



1.4 Blynk 환경 설정

◆ Blynk 라이브러리 설치

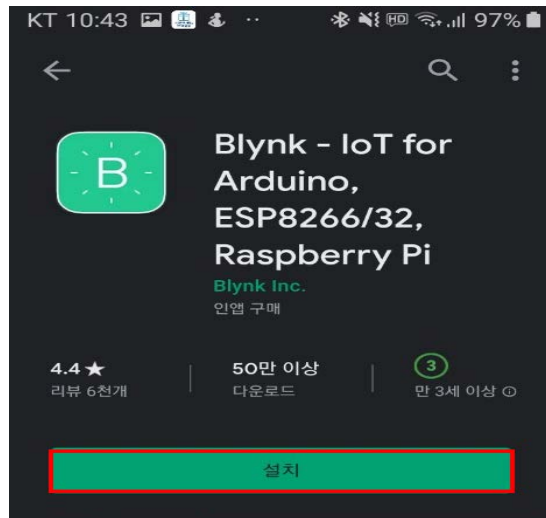
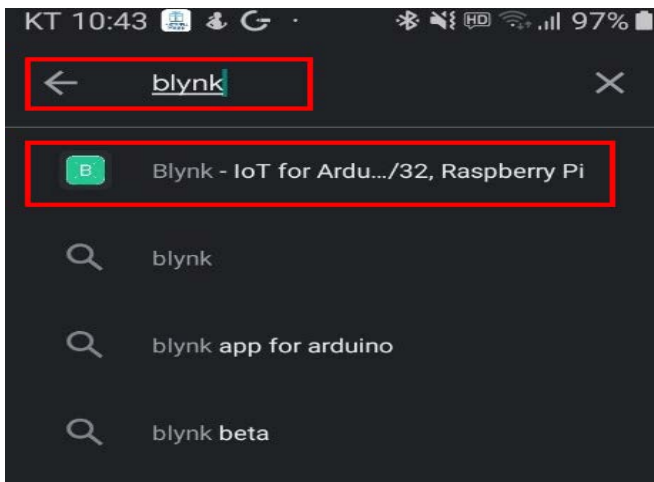
● Blynk 검색 후 설치



1.4 Blynk 환경 설정

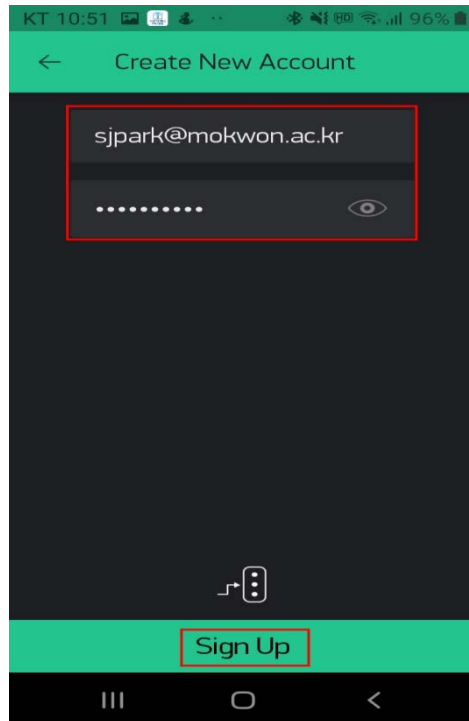
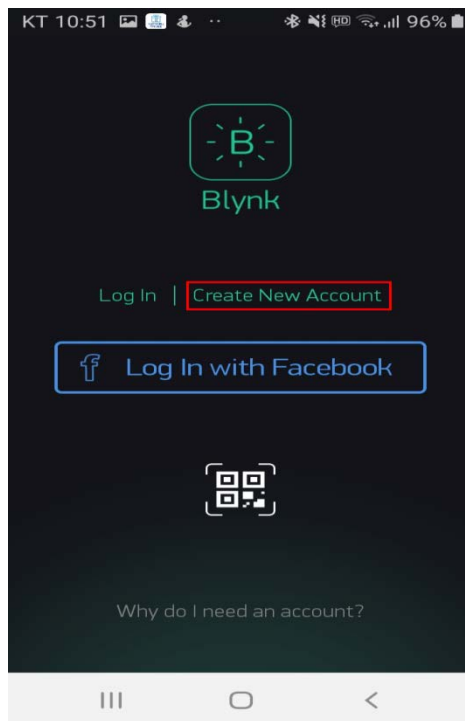
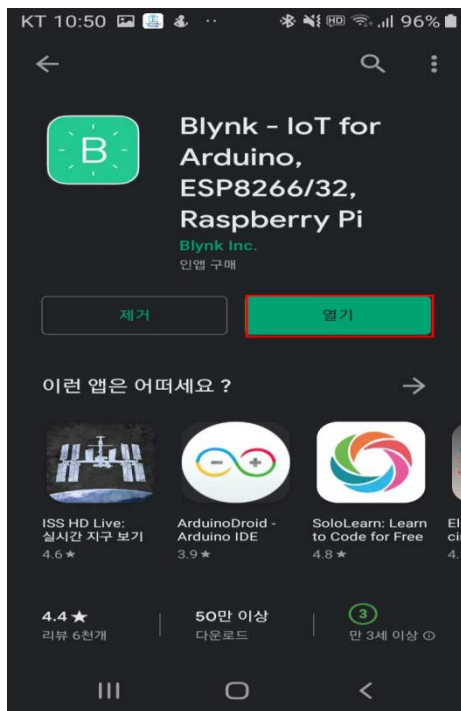
◆ 스마트폰 Blynk 앱 설치

- 안드로이드(Play Store)
- IOS(App Store)



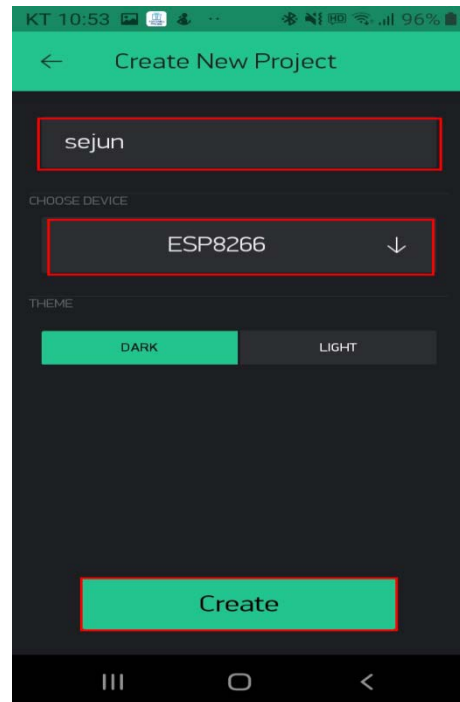
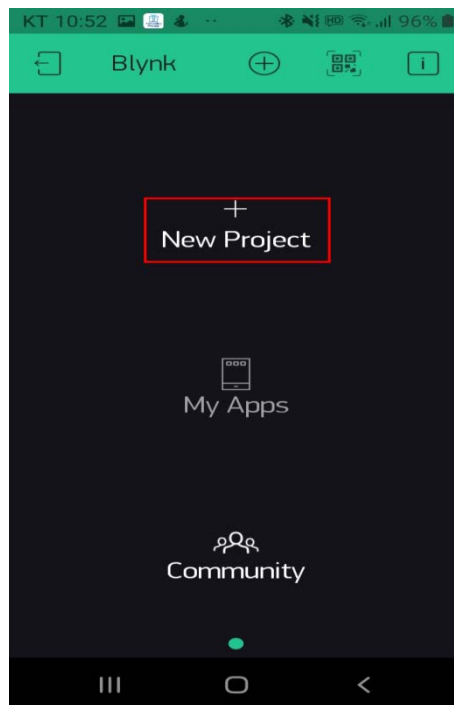
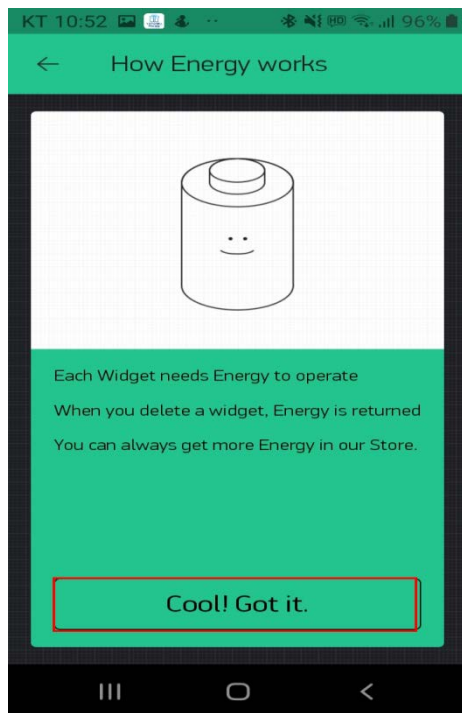
1.4 Blynk 환경 설정

◆ 스마트폰 Blynk 앱 설정



1.4 Blynk 환경 설정

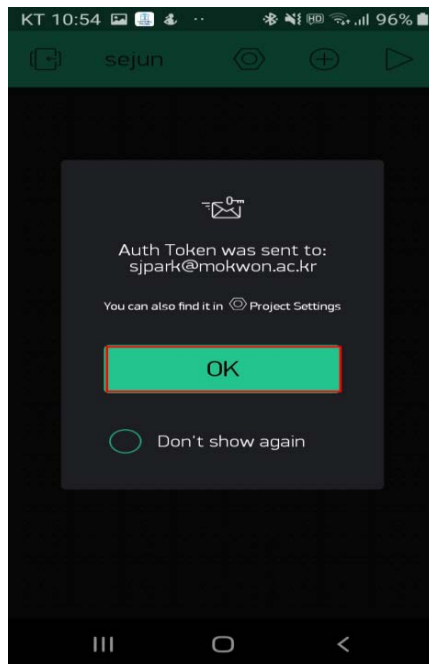
◆ 스마트폰 Blynk 앱 설정



1.4 Blynk 환경 설정

◆ 스마트폰 Blynk 앱 설정

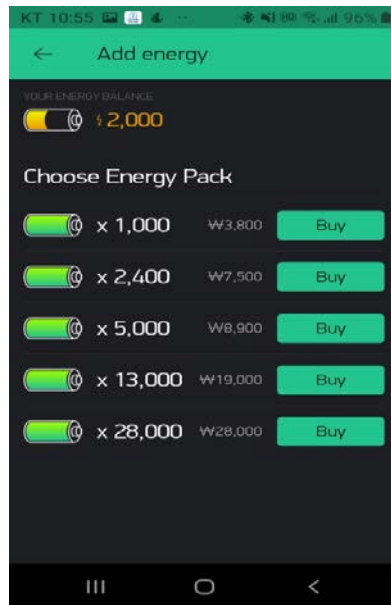
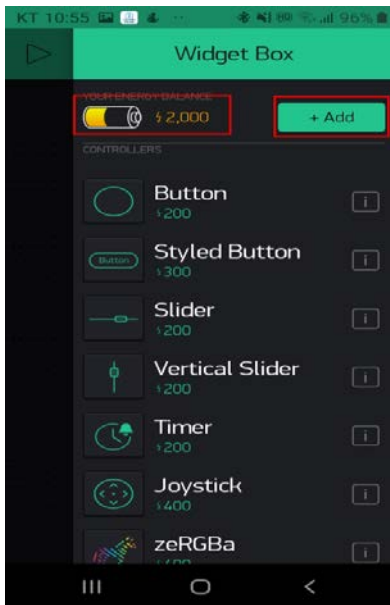
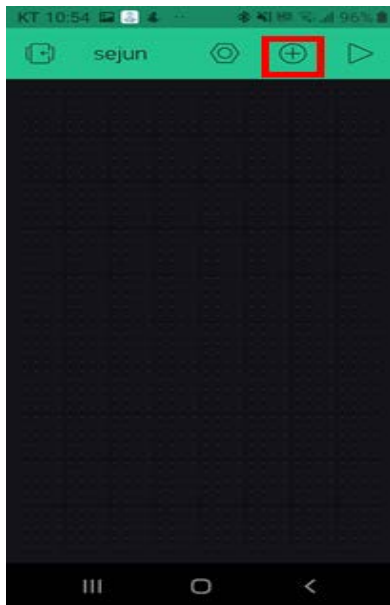
- 이메일에서 토큰 확인



1.4 Blynk 환경 설정

◆ 스마트폰 Blynk 앱 설정

- Widget 추가(사용 제한)



1.5 시리얼 통신

◆ 시리얼 통신

UART (Universal Asynchronous Receiver/Transmitter)

RS-232

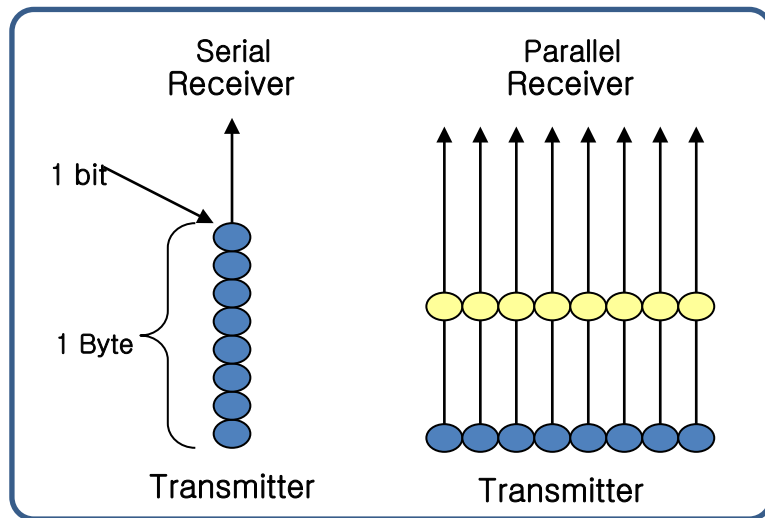
RS-422

RS-485

Arduino에서는 다음과 같은 목적으로 사용

Debugging : 프로그램의 오류를 수정하는 작업

데이터 통신 : Arduino와 컴퓨터(다른 장치)와 통신



1.5 시리얼 통신

◆ 시리얼 모니터(Serial Monitor)

- NodeMCU와 데이터를 주고 받기 위해 사용하는 툴
- 주로 NodeMCU 디버깅 위해 사용



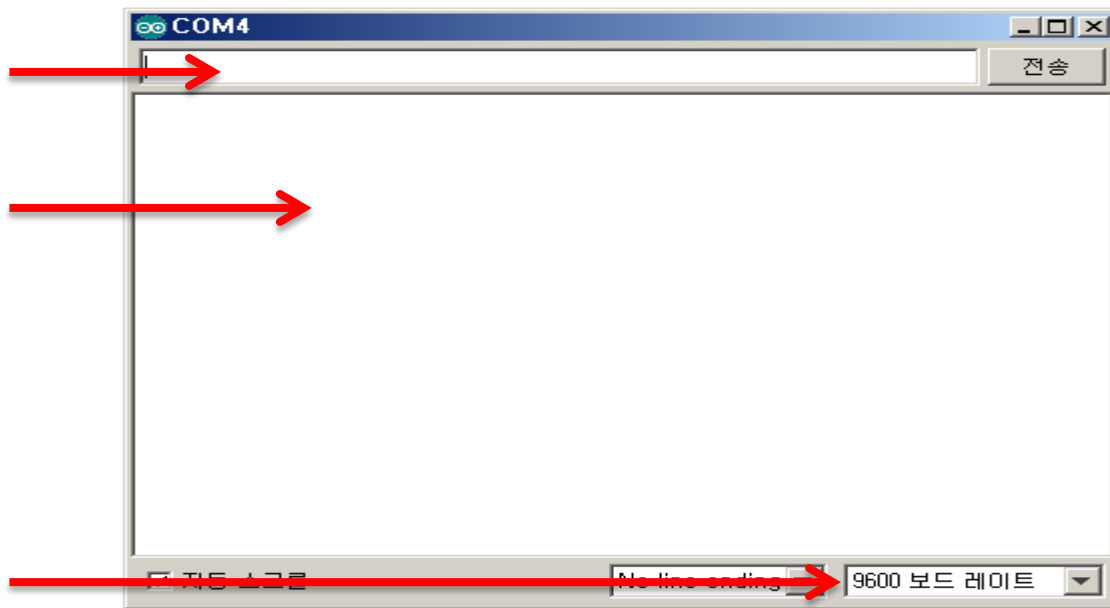
1.5 시리얼 통신

◆ 시리얼 모니터 사용 방법

메시지 입력 창

수신 메시지 표시 창

통신 속도 설정



1.5 시리얼 통신

◆ 시리얼 통신 Commands

- Serial.begin(전송속도)
 - 시리얼 통신 포트 컴퓨터 연결
 - 전송속도(bps :bits per sec)
 - 9600, 19200, 57600, 115200 등의 값 설정 가능
 - 통신을 할 때 속도가 빠를수록 좋지만 , 속도에 비례하여 관련 장비 가격이 비쌘

1.5 시리얼 통신

◆ 시리얼 통신 Commands

- Serial.print(전송내용)
 - 괄호 안의 내용 시리얼 통신으로 전송
 - 따옴표 있음 - 텍스트 직접 전송
 - 따옴표 없음 - 변수 값 전송
- Serial.println(전송내용)
 - 'Serial.print()'와 같으나 전송 후 한 줄 내리기

1.5 시리얼 통신

◆ 시리얼 통신 Commands

- delay(지연시간)
 - 잠시 동작 지연
 - 1/1000초 단위(1초=1000)
- Serial.available()
 - 시리얼 통신에 수신 데이터가 있는지 확인(있을 경우 - true)
- Serial.read()
 - 시리얼 통신을 통하여 수신된 값 read