

제3장. 아두이노 클래스

- 함수와 클래스
- 클래스 & 시리얼 통신
 - 시리얼통신
- 기본 클래스(Class)
 - Serial 클래스
 - String 클래스
- 별첨. 라이브러리 폴더

1. 함수와 클래스(Class)

- 함수(Function) 정의
 - 특정 기능을 수행하는 프로그램의 묶음
 - 아두이노 IDE의 경우
setup(), loop() 기본 함수 사용

- 함수 구조

```
전달받을 데이터 형   함수명(인수) {  
    .....  
    return 전달할 데이터에 대한 값 또는 변수  
}
```

- 사례

```
void 함수명(인수) {  
    .....  
}
```

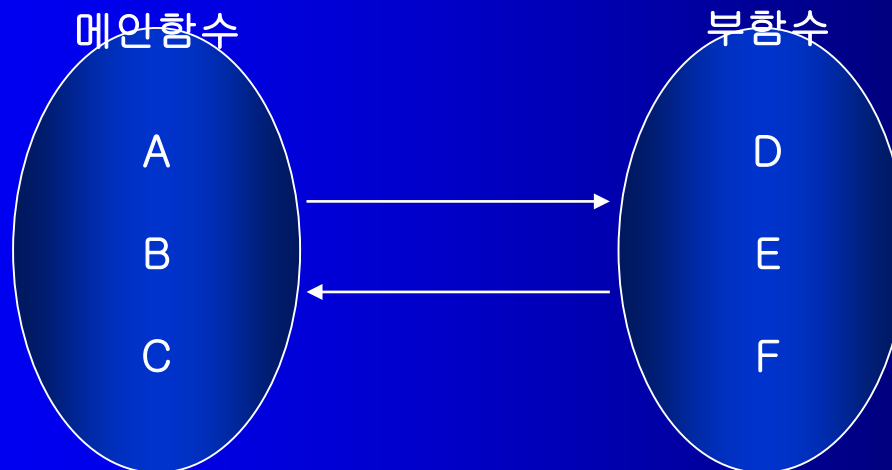
■ 함수의 호출 및 인수

- 호출(Call)

- 함수(메인함수, 서브루틴 또는 부함수)간의 함수 활용
- 함수호출 방법, 함수명

- 인수

- 함수간 데이터의 전달 방법
- 전달 방법
 - 값을 사용하는 전달(Call by Value)
 - 변수(&)를 사용하는 전달(Call by Reference)



- 사례-값을 사용하는 전달(Call by Value)

```
int a1=2;
```

```
int a2=3;
```

```
int a3;
```

```
void setup(){
```

```
  a3=sum(a1,a2);
```

```
}
```

```
void loop(){
```

```
}
```

```
int sum(int a, int b){
```

```
  c=a+b;
```

```
  return c;
```

```
}
```

실 인수(인수)

가 인수(매개변수)

- 사례-변수를 사용하는 전달(Call by Reference)

```
int a1=2;
```

```
int a2=3;
```

```
int a3;
```

```
void setup(){
```

```
  sum(a1,a2,a3);
```

```
}
```

```
void loop(){
```

```
}
```

```
void sum(int a, int b, int&c){
```

```
  c=a+b;
```

```
}
```

실 인수(인수)

가 인수(매개변수)

■ 함수의 프로토타입(Prototype)

- 코딩과정에서 사용하게 될 **프로그램의 요약정보**를 선언한 것
- 프로그램의 요약정보 ?
 - 돌려받는 데이터의 형, 함수 명, 인수

- 사례

void sum(int a, int b, int& c);

void setup(){

.....

}

void loop(){

}

void sum(int a, int b, int& c){

.....

}

- 1. 아두이노 IDE 경우
컴파일과정에서 자동으로 생성
- 1. C++ 경우
명시적으로 선언해야 함

□ 클래스(Class)

라이브러리

폴더기반의 프로그램 묶음

▪ 클래스

- 재사용이 가능한 프로그램의 묶음
- 아두이노는 많은 함수와 클래스로 구성
- IoT기반의 대용량의 프로그램 작성을 위해서는
함수단위 또는 클래스 단위로 프로그램 작성 권장
- 클래스 기본 구조

```
class class_name{  
    public:  
        class_name();  
        class_name(argument.....);  
        ....  
        methods(functions)  
    private:  
        variables  
};
```

Class는
명령어 이므로 반드시 끝에 “;”

■ 클래스 기본 구조(1)– public과 private

- 클래스는 public과 private 부문으로 구성
- public
 - 클래스의 외부에 알려져야 할 메소드(함수)의 프로토타입 선언
- private
 - 메소드 안에서 사용하게 될 변수선언

```
class class_name{  
    public:  
        class_name();  
        class_name(argument.....);  
        ....  
        methods(functions)  
    private:  
        variables  
};
```


■ 클래스 기본 구조(2)-클래스와 메소드

Class -> Led
객체 -> led

- 클래스명-class_name
 - 일반적으로 **대문자**로 시작함
 - 클래스를 사용하기 위해 객체(Object)를 생성 할때, **소문자**를 사용하여 구분
 - 사례 : Led led(argument)
- 메소드 또는 함수(Method or Function)
 - 클래스에서 사용하는 함수의 **프로토타입** 설정
(뒤에서 코딩하게 될 함수에 대한 요약)
 - 생성자(Constructor)
 - 클래스를 사용하기위해 객체(Object)를 생성할때 사용하는 함수
 - 클래스명과 동일명을 사용하는 함수
 - 사례 : Led::Led(....)

```
class class_name{  
    public:  
        class_name();  
        class_name(argument.....);  
        ....  
        methods(functions)  
    private:  
        variables  
};
```

■ 클래스 기본 구조(3)-메소드 만들기

- 클래스문 밖에서 메소드(함수)작성
- 임의의 클래스에 속하는 메소드 작성시,
“**class_name::**” 사용
- 사례 : **class_name::메소드**
Led::led(••••)
- 범위지정연산자(scope resolution operator), “**::**”
 - 메소드가 속하는 클래스를 지정

Class -> Led
메소드 -> led

□ 스케치 개발방법

- 아두이노 기반의 스케치 개발방법
 - 아두이노 기본구조(setup 및 loop함수)기반 방법
 - 함수기반 개발방법
 - 클래스기반 개발방법(1)-기본
클래스기반 개발방법(2)-Advanced

▪ 아두이노 IDE기반 스케치 개발-함수단위

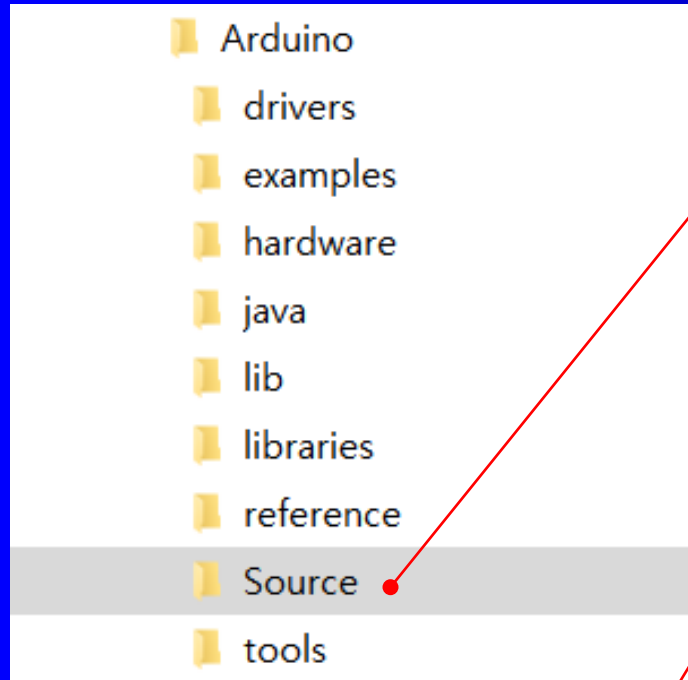
- 임의의 스케치를 함수단위로 구성
- 스케치를 함수단위로 구분
- 샘플 스케치
 - serial-led.ino
 - 내장 LED를 1초 간격으로 On/Off시키는 스케치

- 샘플 스케치-serial-led.ino

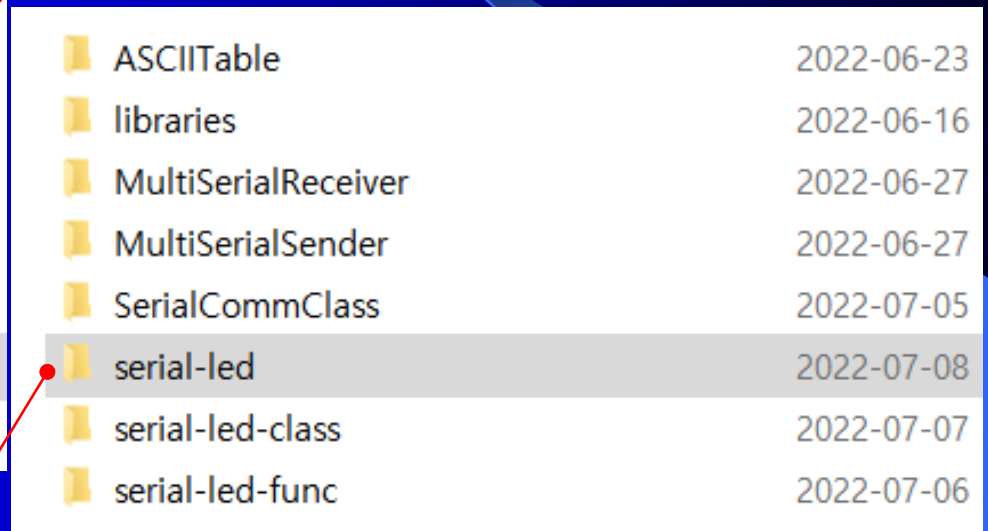
```
serial-led
1 #define LED_PIN LED_BUILTIN
2
3 #define LED_OFF LOW
4 #define LED_ON HIGH
5
6 void setup() {
7     Serial.begin(9600);
8     pinMode(LED_PIN, OUTPUT);
9
10 }
11
12 void loop() {
13
14     digitalWrite(LED_PIN, LED_ON);
15     delay(1000);
16     digitalWrite(LED_PIN, LED_OFF);
17     delay(1000);
18
19 }
```

LED에 대해서
On/Off부분을 함수화

• 샘플 스케치-폴더구조



Arduino 메인
Source폴더 생성



IDE환경에서

-> Serial-led폴더 생성

-> Serial-led.ino파일생성
(메인파일)



- 함수 단위 구분
 - 샘플스케치에서 LED의 On/Off부분
 - void onLed(), void offLed()

```
serial-led
20                                     )_BUILTIN
21 void onLed() {                               /
22     digitalWrite(LED_PIN, LED_ON);           ;H
23 }
24
25 void offLed() {                               );
26     digitalWrite(LED_PIN, LED_OFF);          OUTPUT);
27 }
```

```
12 void loop() {
13
14     digitalWrite(LED_PIN, LED_ON);
15     delay(1000);
16     digitalWrite(LED_PIN, LED_OFF);
17     delay(1000);
18
19 }
```

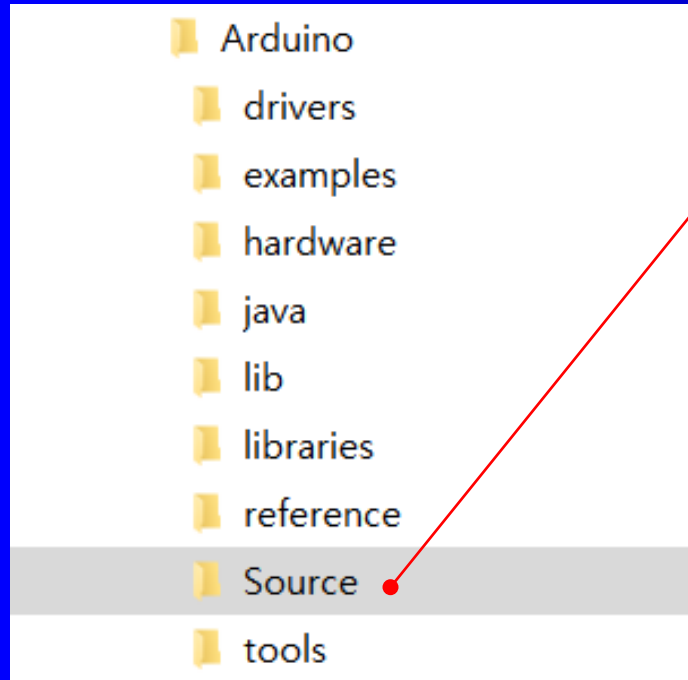
- 메인함수에서 호출

serial-led-func

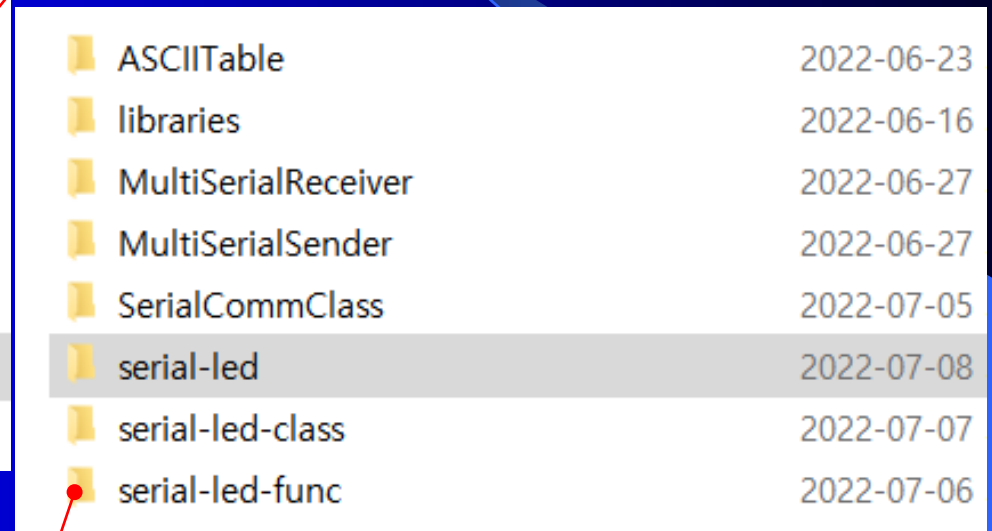
```
1 #define LED_PIN LED_BUILTIN
2
3 #define LED_OFF LOW
4 #define LED_ON HIGH
5
6 void setup() {
7     Serial.begin(9600);
8     pinMode(LED_BUILTIN, OUTPUT);
9
10 }
11
12 void loop() {
13
14     onLed();
15     delay(1000);
16     offLed();
17     delay(1000);
18
19 }
```

```
20
21 void onLed() {
22     digitalWrite(LED_PIN, LED_ON);
23 }
24
25 void offLed() {
26     digitalWrite(LED_PIN, LED_OFF);
27 }
```


• serial-led-func 스케치-폴더구조



Arduino 메인
Source폴더 생성



IDE환경에서

-> Serial-led-func폴더 생성

-> Serial-led-func.ino파일생성
(메인파일)



- 아두이노 IDE기반 스케치개발(1)-기본-클래스 단위

- ① 아두이노 IDE 탭(Tab)활용

- 아두이노 폴더기반의 다중 프로그램 관리 기능

- ② class파일 생성->class file.h

- ③ 아두이노 파일 생성-> Arduino file.ino

- 샘플 스케치

- serial-led.ino

- 내장 LED를 1초 간격으로 On/Off시키는 스케치

- 샘플 스케치-serial-led.ino

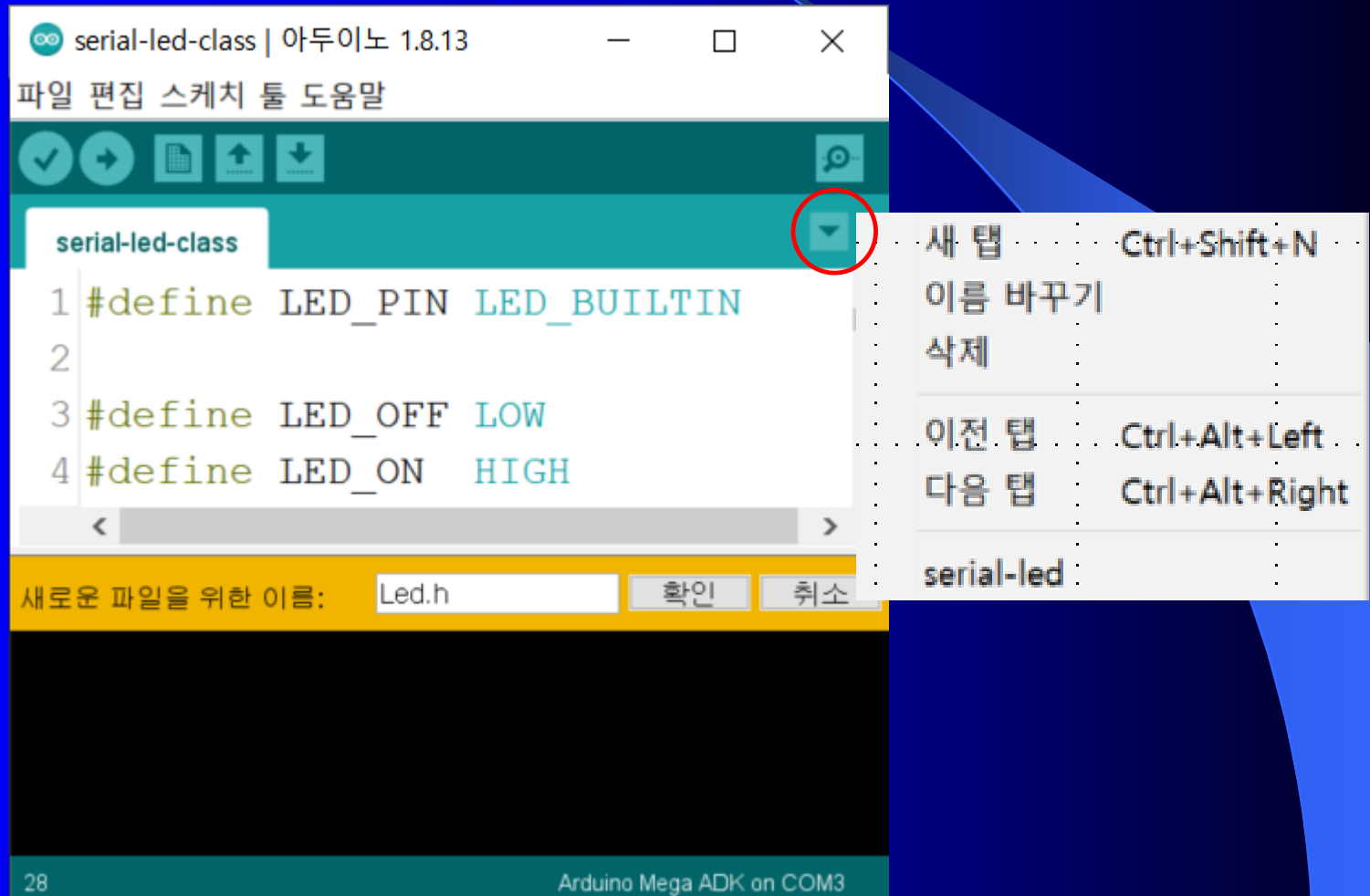
```
serial-led
1 #define LED_PIN LED_BUILTIN
2
3 #define LED_OFF LOW
4 #define LED_ON HIGH
5
6 void setup() {
7     Serial.begin(9600);
8     pinMode(LED_PIN, OUTPUT);
9
10 }
11
12 void loop() {
13
14     digitalWrite(LED_PIN, LED_ON);
15     delay(1000);
16     digitalWrite(LED_PIN, LED_OFF);
17     delay(1000);
18
19 }
```

LED에 대해서
On/Off부분을 예소드화

① 아두이노 IDE 탭(Tab)활용

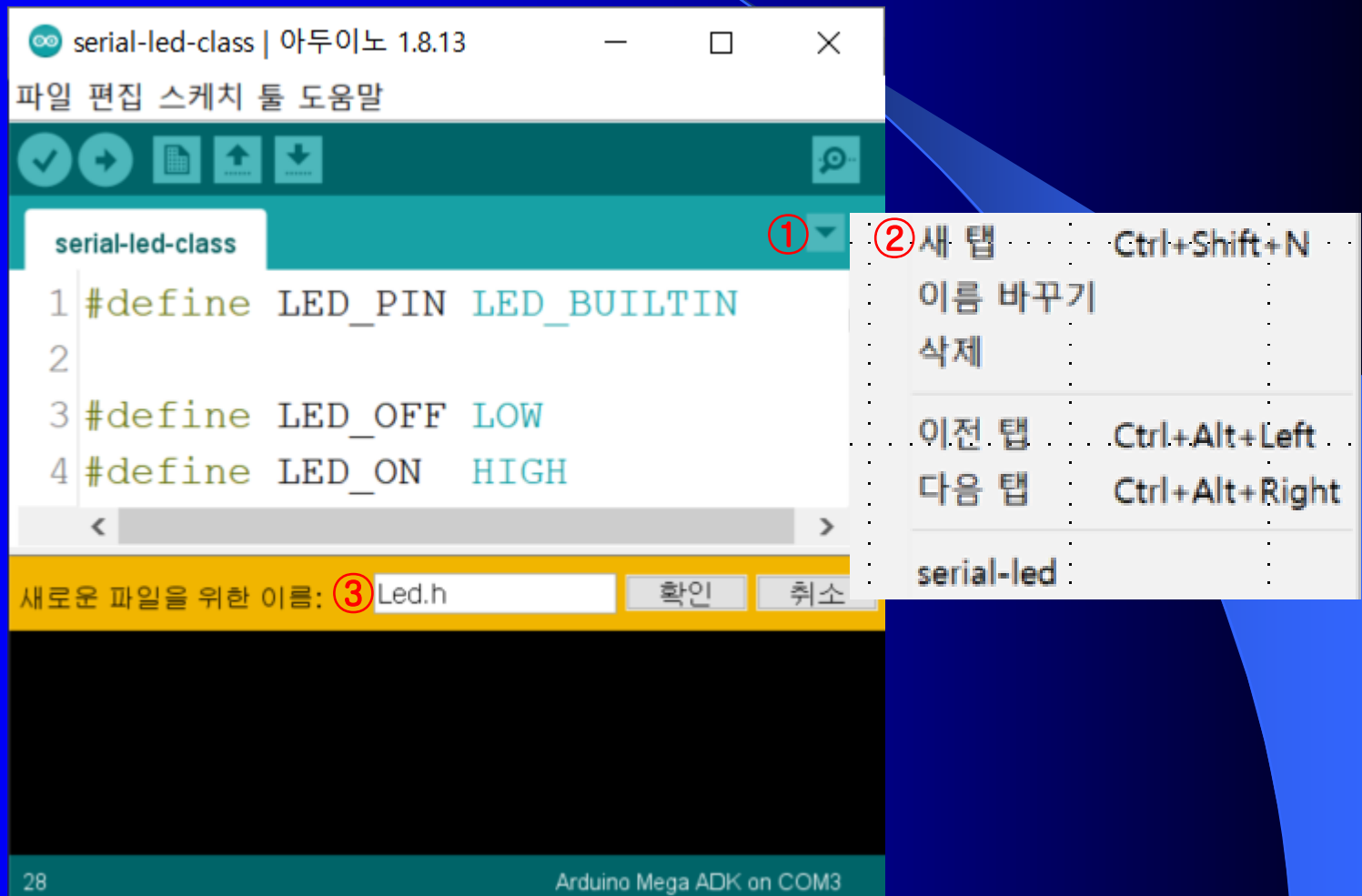
▪ IDE 탭

- 통합개발환경에서 다중 스케치 편집 기능

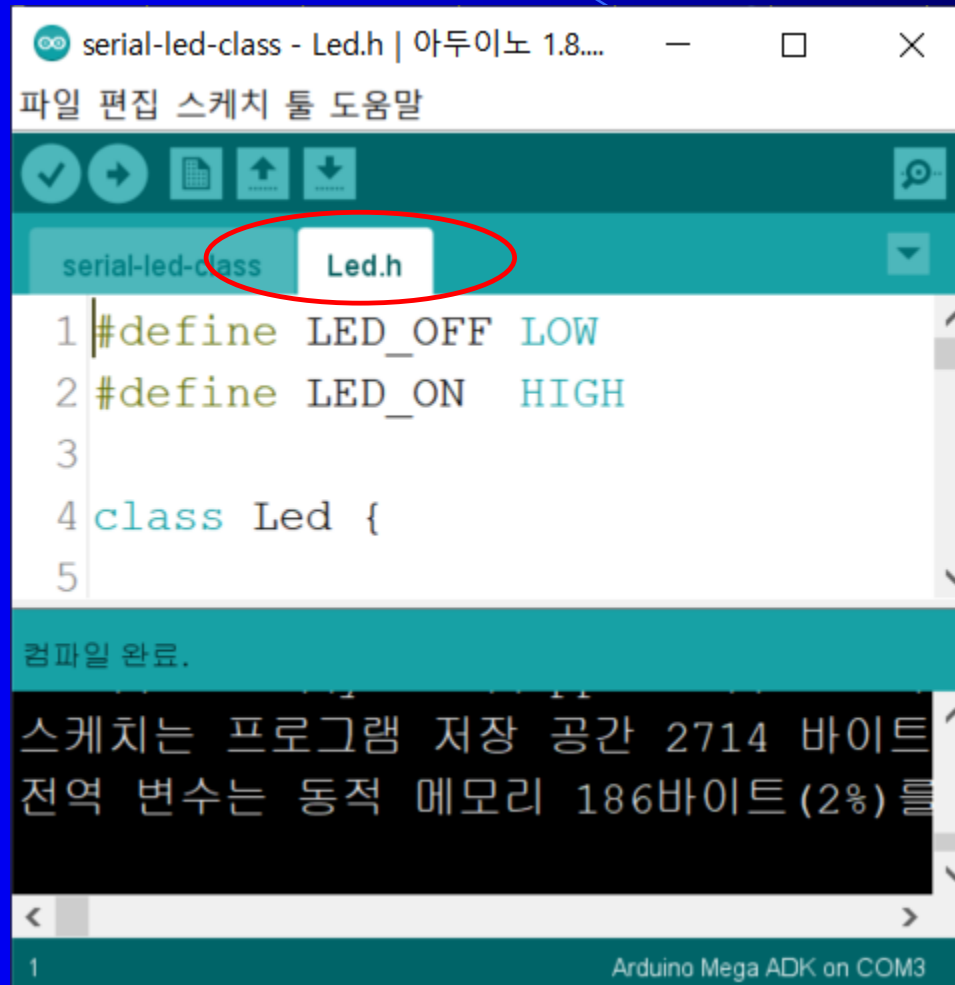


② 새로운 스케치 만들기-Led.h

- 탭 -> 새 탭 -> 새로운 파일을 위한 이름
- 새로운 파일 : Led.h



- 새 파일 작성-Led.h



serial-led-class - Led.h | 아두이노 1.8....

파일 편집 스케치 툴 도움말

serial-led-class Led.h

```
1 #define LED_OFF LOW
2 #define LED_ON HIGH
3
4 class Led {
5
```

컴파일 완료.

스케치는 프로그램 저장 공간 2714 바이트
전역 변수는 동적 메모리 186바이트 (2%) 를

1 Arduino Mega ADK on COM3

- class정의 부
 - 클래스 명 : Led
 - public부
 - 클래스 프로토타입 : Led(int pin)
 - 메소드 : void onLed()
void offLed()
 - private부
 - 변수 : pin
(Led I/O포트 번호)

```
serial-led-class  Led.h
1 #define LED_OFF LOW
2 #define LED_ON  HIGH
3
4 class Led {
5
6     public:
7
8         Led(int pin);
9
10        void  onLed();
11        void  offLed();
12
13    private:
14
15        int pin;
16
17};
```

- 메소드 부

- 메소드 :

- Led::Led()

- //생성자(클래스 이름과 동일한 메소드 이름사용)

- //클래스 형지정자는 없음

- void Led::onLed()

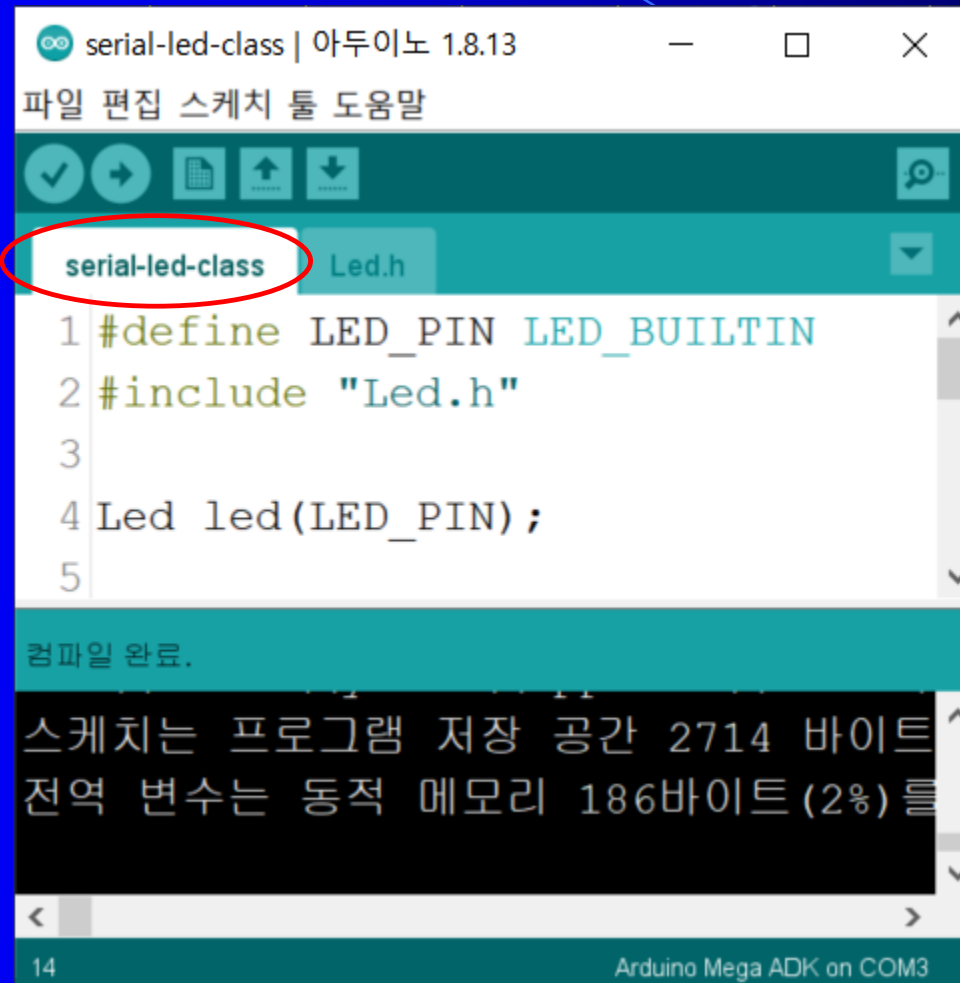
- //Led On메소드

- void Led::onLed()

- //Led On메소드

```
19 Led::Led(int INpin) {
20
21     pin = INpin;
22
23     pinMode(pin, OUTPUT);
24 }
25
26 void Led::onLed() {
27     digitalWrite(pin, LED_ON);
28 }
29
30 void Led::offLed() {
31     digitalWrite(pin, LED_OFF);
32 }
33
```


③ 새 파일 작성-serial-led-class.ino



serial-led-class | 아두이노 1.8.13

파일 편집 스케치 툴 도움말

serial-led-class Led.h

```
1 #define LED_PIN LED_BUILTIN
2 #include "Led.h"
3
4 Led led(LED_PIN);
5
```

컴파일 완료.

스케치는 프로그램 저장 공간 2714 바이트
전역 변수는 동적 메모리 186바이트 (2%) 를

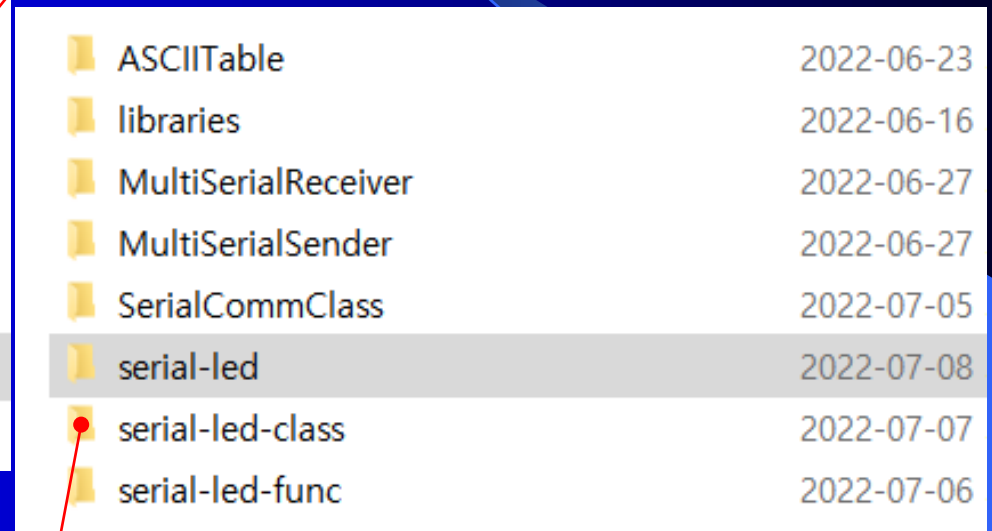
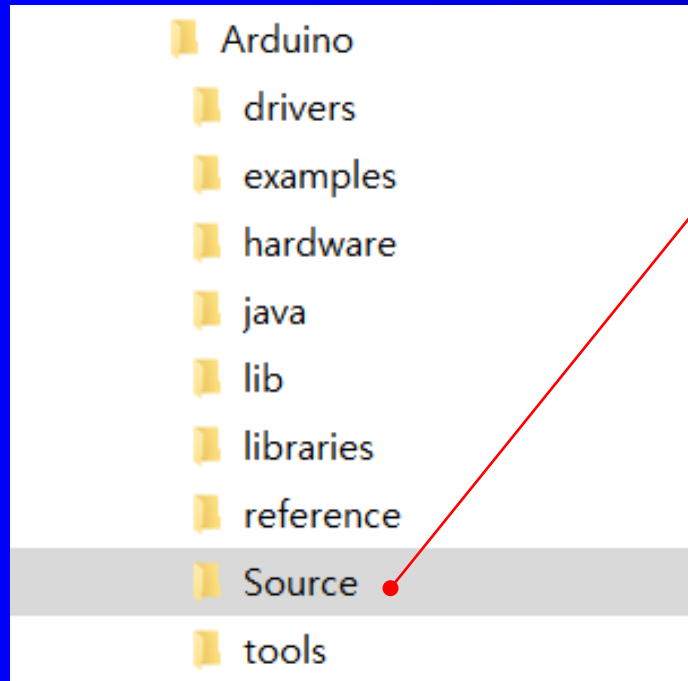
14 Arduino Mega ADK on COM3

- define LED_PIN LED_BUILTIN
 - 사용 LED I/O포트 설정
- include "Led.h"
 - 메인 프로그램에서 사용하는 Class파일 지정
- Led led(LED_PIN)
 - 클래스기반 객체 생성
- led.onLed() 및 led.offLed()
 - 메소드로 만들어진 프로그램

```
1 #define LED_PIN LED_BUILTIN
2 #include "Led.h"
3
4 Led led(LED_PIN);
5
6 void setup() {
7     Serial.begin(9600);
8 }
9
10 void loop() {
11
12     led.onLed();
13     delay(1000);
14     led.offLed();
15     delay(1000);
16
17 }
```

• serial-led-class 스케치-폴더구조

Arduino 메인
Source폴더 생성



IDE환경에서

- > Serial-led-class폴더 생성
- > Serial-led-class.ino파일 생성 (메인파일)
- > Led.h파일 생성



■ 아두이노 IDE기반 스케치개발(2)-Advanced-클래스 단위

① 아두이노 IDE 탭(Tab)활용

아두이노 폴더기반의 다중 프로그램 관리 기능

② class파일 생성(클래스 정의부)->class file.h

③ 라이브러리 파일 생성(메소드 정의부)->메소드 file.cpp

④ 아두이노 파일 생성-> Arduino file.ino

• 샘플 스케치

· serial-led.ino

· 내장 LED를 1초 간격으로 On/Off시키는 스케치

- 샘플 스케치-serial-led.ino

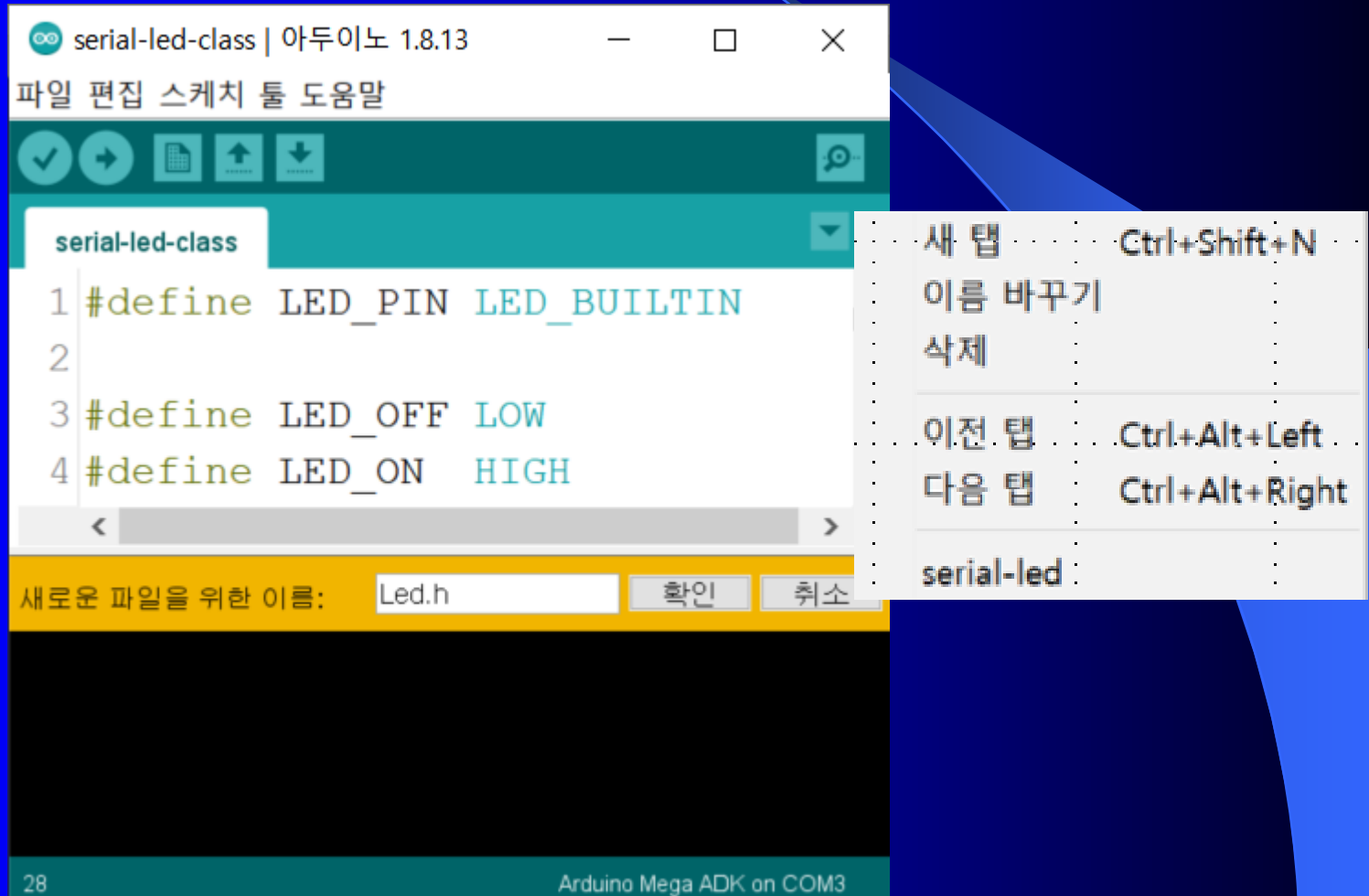
```
serial-led
1 #define LED_PIN LED_BUILTIN
2
3 #define LED_OFF LOW
4 #define LED_ON HIGH
5
6 void setup() {
7     Serial.begin(9600);
8     pinMode(LED_PIN, OUTPUT);
9
10 }
11
12 void loop() {
13
14     digitalWrite(LED_PIN, LED_ON);
15     delay(1000);
16     digitalWrite(LED_PIN, LED_OFF);
17     delay(1000);
18
19 }
```

LED에 대해서
On/Off부분을
라이브러리화

① 아두이노 IDE 탭(Tab)활용

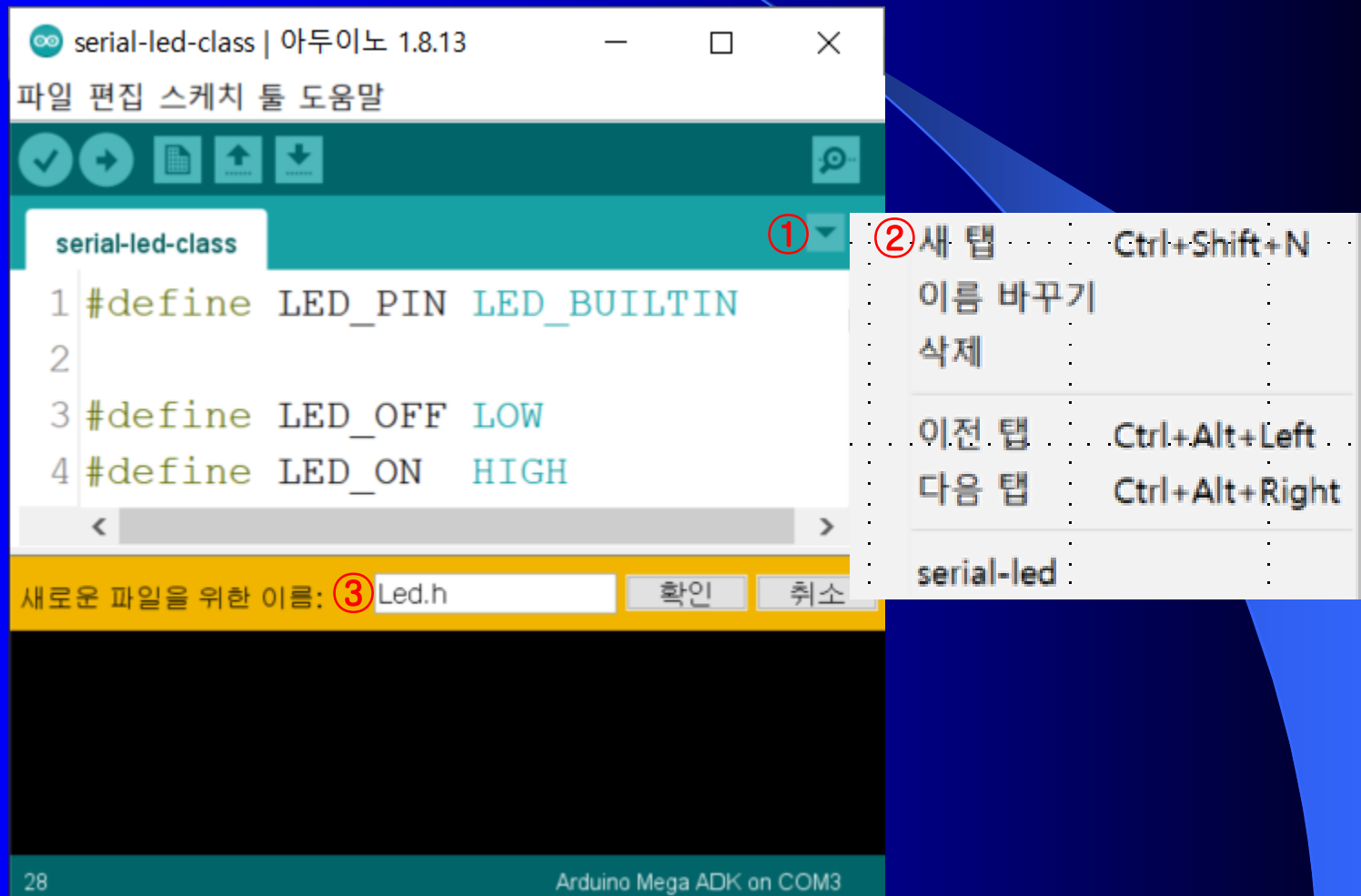
▪ IDE 탭

- 통합개발환경에서 다중 스케치 편집 기능

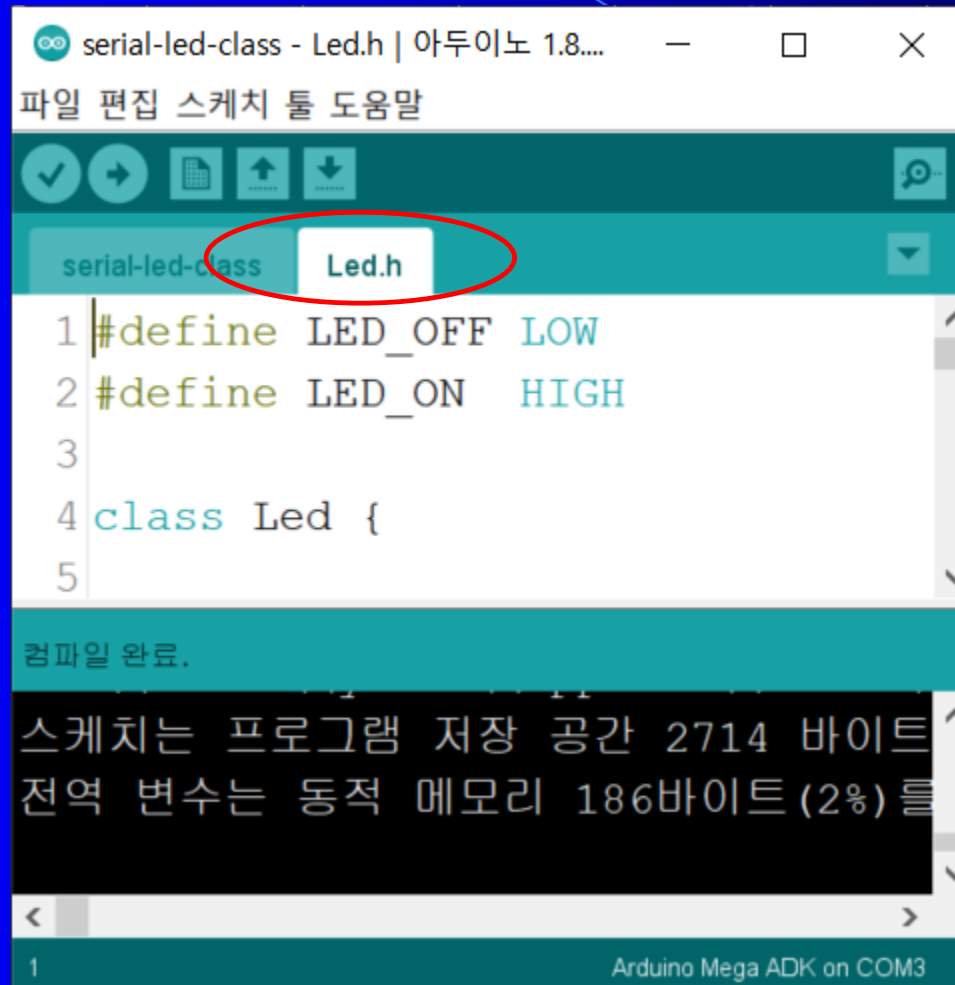


② 새로운 스케치 만들기-Led.h

- 탭 -> 새 탭 -> 새로운 파일을 위한 이름
- 새로운 파일 : Led.h



- 새 파일 작성-Led.h



serial-led-class - Led.h | 아두이노 1.8....

파일 편집 스케치 툴 도움말

serial-led-class Led.h

```
1 #define LED_OFF LOW
2 #define LED_ON HIGH
3
4 class Led {
5
```

컴파일 완료.

스케치는 프로그램 저장 공간 2714 바이트
전역 변수는 동적 메모리 186바이트 (2%) 를

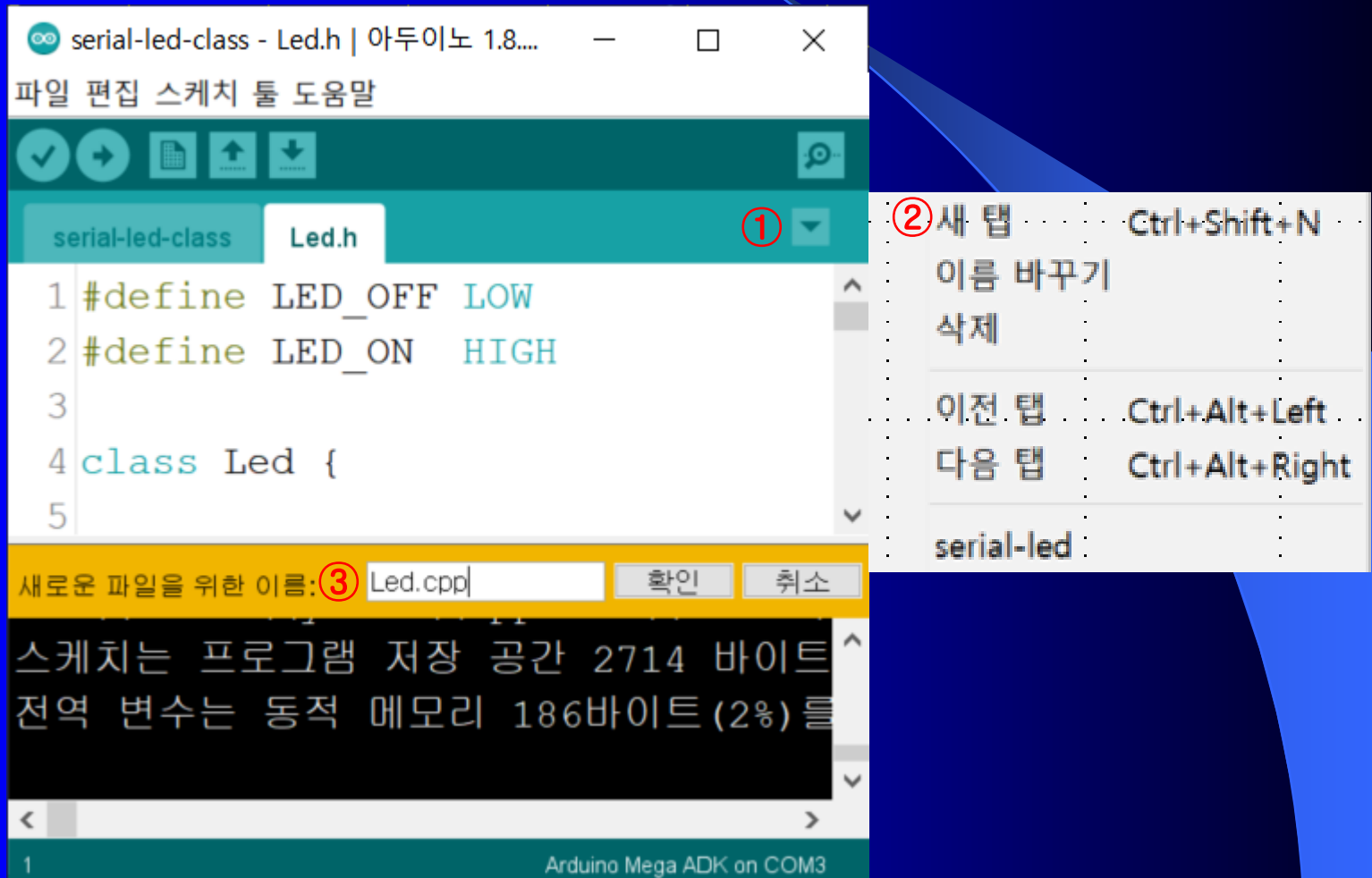
1 Arduino Mega ADK on COM3

- class정의 부
 - 클래스 명 : Led
 - public부
 - 클래스 프로토타입 : Led(int pin)
 - 메소드 : void onLed()
void offLed()
 - private부
 - 변수 : pin
(Led I/O포트 번호)

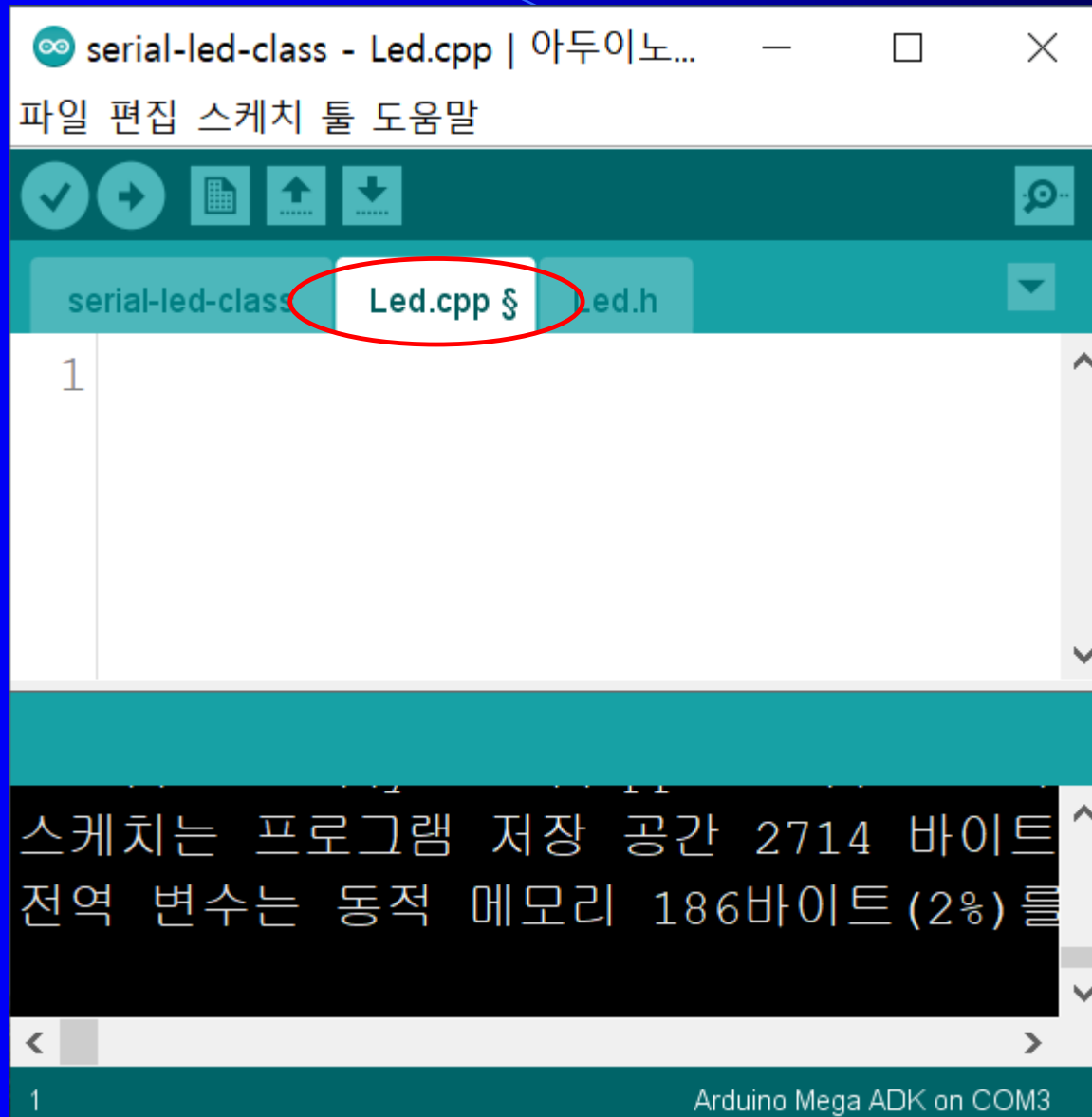
```
serial-led-class  Led.h
1 #define LED_OFF LOW
2 #define LED_ON  HIGH
3
4 class Led {
5
6     public:
7
8         Led(int pin);
9
10        void  onLed();
11        void  offLed();
12
13    private:
14
15        int pin;
16
17};
```

③ 새 파일 작성-Led.cpp

- 탭 -> 새 탭 -> 새로운 파일을 위한 이름
- 새로운 파일 : Led.cpp



- 새 파일 작성-Led.cpp



- 메소드 부

- #include <Arduino.h>
//.cpp파일은 별도로 컴파일 됨
//아두이노 기본 라이브러리 필요

- #include <Led.h>
//.cpp파일은 별도로 컴파일 됨
//클래스 선언 정보가 필요

- 메소드 :

- Led::Led()
//생성자(클래스 이름과 동일한 메소드)
//클래스 형지정자는 없음

- void Led::onLed()
//Led On메소드

- void Led::onLed()
//Led On메소드

```
serial-led-class  Led.cpp  Led.h
1 #include <Arduino.h>
2 #include "Led.h"
3
4 Led::Led(int INpin) {
5
6     pin = INpin;
7
8     pinMode(pin, OUTPUT);
9 }
10
11 void Led::onLed() {
12     digitalWrite(pin, LED_ON);
13 }
14
15 void Led::offLed() {
16     digitalWrite(pin, LED_OFF);
17 }
```

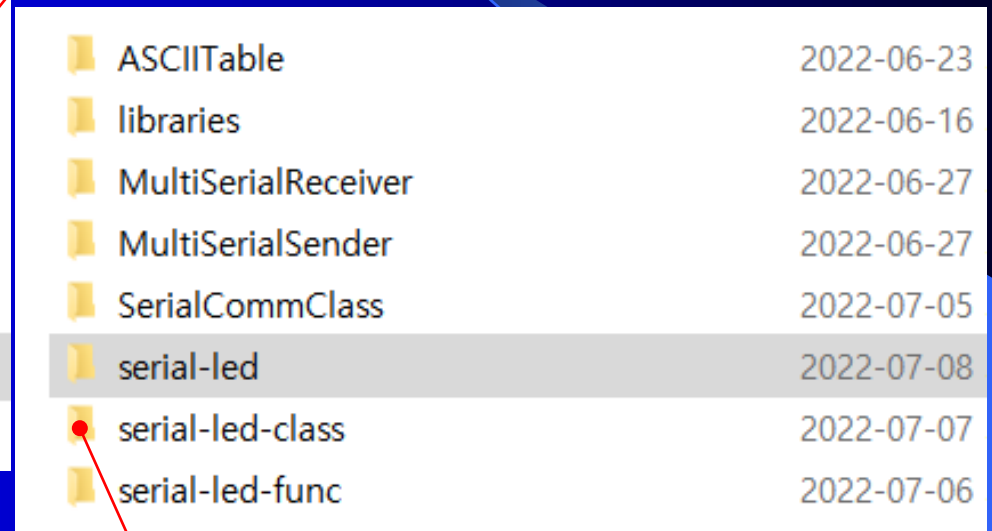
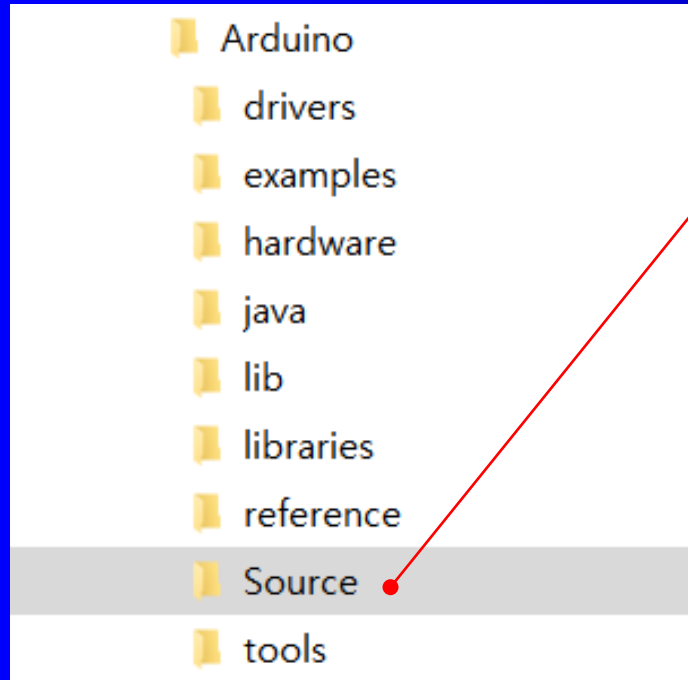
④ 새 파일 작성-serial-led-class.ino

- define LED_PIN LED_BUILTIN
 - 사용 LED I/O포트 설정
- include "Led.h"
 - 메인 프로그램에서 사용하는 Class파일 지정
- Led led(LED_PIN)
 - 클래스기반 객체 생성
- led.onLed() 및 led.offLed()
 - 메소드로 만들어진 프로그램

```
serial-led-class  Led.h
1 #define LED_PIN LED_BUILTIN
2 #include "Led.h"
3
4 Led led(LED_PIN);
5
6 void setup() {
7     Serial.begin(9600);
8 }
9
10 void loop() {
11
12     led.onLed();
13     delay(1000);
14     led.offLed();
15     delay(1000);
16
17 }
```

• serial-led-class 스케치-폴더구조

Arduino 메인
Source폴더 생성



IDE환경에서

- > Serial-led-class폴더 생성
- > Serial-led-class.ino파일생성 (메인파일)
- > Led.h파일생성(클래스파일)
- > Led.cpp파일생성(라이브러리파일)



▪ 과제

아두이노 폴더(라이브러리)의 파일 구성을 확인해 보세요

2. 클래스(Class) & 시리얼통신

□ 클래스(Class)

▪ 클래스(Class)

- 객체를 생성하기 위한 **변수** 와 **메소드(함수)**를 정의하는 틀
- 기본 클래스
 - Serial, String클래스
 - 클래스를 사용하기 위하여 별도의 헤더파일 불필요
 - Serial클래스
 - 시리얼통신을 위해 사용
 - String클래스
 - 문자열 처리를 위해 사용

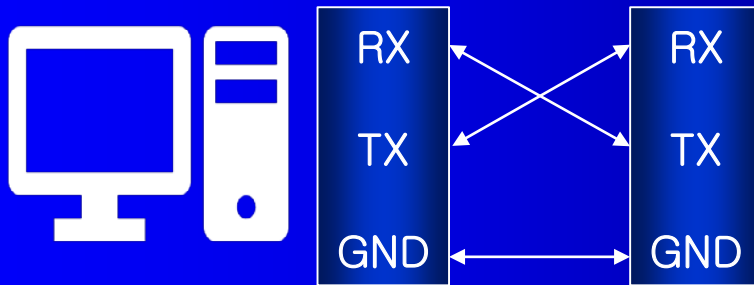
▪ 라이브러리

- 아두이노에 연결된 주변장치 제어
- C++의 객체지향방식으로 **클래스**를 사용하여 작성

□ 시리얼 통신

▪ 아두이노 시리얼 통신

- UART(Universal Asynchronous Receiver Transmitter)
- RS-232C 프로토콜
- RX, TX, GND 단자 통신



S-USB 커넥터 UART 커넥터



- 아두이노 메가
 - 4개의 시리얼포트

< 아두이노 메가, 시리얼 통신 >

채널	연결핀	객체	설명
0	0(RX), 1(TX)	Serial	우노와 동일
1	19(RX), 18(TX)	Serial1	
2	17(RX), 16(TX)	Serial2	
3	15(RX), 14(TX)	Serial3	

- USB
 - 스케치 다운로드
 - 컴퓨터와 시리얼 통신

※ 뒤장 그림으로 설명

- USB
 - 스케치 다운로드
 - 컴퓨터와 시리얼 통신



스케치 업로드



시리얼 통신

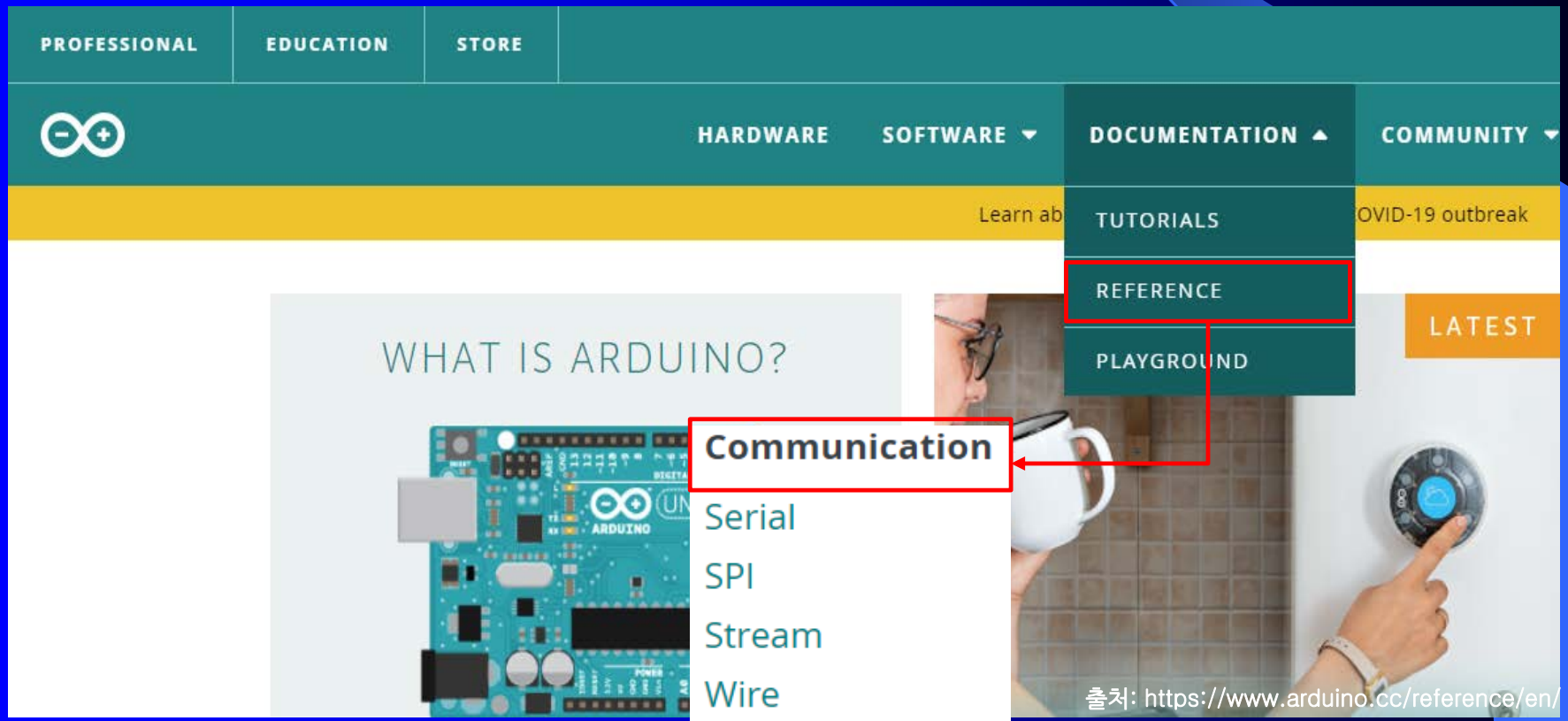


< 콘솔 창 : 시리얼모니터 >

3. 기본 클래스

▪ Serial 클래스

- 아두이노 홈페이지, Reference → Function → Communication
- 아두이노 통신클래스
 - Serial
 - Stream



■ Serial 클래스

- ATmega 2560, 아두이노와 **시리얼 장치**의 데이터 통신
- 종류
 - Serial, Serial1, Serial2, Serial3
 - Serial : 시리얼모니터
 - Serial1, Serial2, Serial3 : 시리얼장치

• 멤버함수

- print(), println(), write()
 - 시리얼 장치에 대해서 문자열기반 출력
 - 시리얼 장치에 대해서 2진데이터기반 출력
- available(), peek(), read()
 - 시리얼 장치로부터 데이터 입력

Functions

if(Serial)
available()
availableForWrite()
begin()
end()
find()
findUntil()
flush()
parseFloat()
parseInt()
peek()
print()
println()
read()
readBytes()
readBytesUntil()
readString()
readStringUntil()
setTimeout()
write()
serialEvent()

■ 맴버함수-Serial.print(), Serial.println()

- 문자열 기반의 데이터 출력
- 문법

- size_t Serial.print(value, format)
- size_t Serial.println(value, format)

- value : 출력값(char, char배열, String, 정수, 실수 등)
- format : 출력형식(DEC, HEX, BIN)
- 리턴값 : 쓰여진 바이트 수(형식 : size_t)
- size_t : 모든 객체의 크기를 바이트 단위로 나타낼 수 있는 데이터 형식

• 사례

- Serial.print("Test"); → Test
- Serial.print(123); → 123
- Serial.print(1.23); → 1.23
- Serial.print(1.23, 0); → 0

■ 멤버함수-Serial.write()

- 2진 데이터 기반의 데이터 출력
- 문법
 - `size_t Serial.write(val)`
 - `size_t Serial.write(str)`
 - `size_t Serial.write(buf, len)`
 - `val` : 단일 바이트 출력값
 - `str` : 연속 바이트의 문자열
 - `buf` : 연속 바이트의 배열
 - `len` : 배열에서 보내지는 바이트 수
- 사례
 - `Serial.write(65);` → “A”

■ 멤버함수-Serial.available()

- 시리얼 통신으로 데이터를 아두이노 보드로 수신
- 데이터의 수신 여부 확인 함수
- 문법

- int Serial.available()

- int : 저장된 데이터의 바이트 수

• 사례

- if(Serial.available()>0); → 만일 시리얼 데이터가 수신되면 ?

- 멤버함수-Serial.read(), Serial.peek()

- 시리얼 통신으로 수신버퍼에 저장된 데이터를 읽기

- 문법

- int Serial.read()

- int Serial.peek()

- 매개변수 : 없음

- int : 시리얼 통신 수신버퍼의 첫번째 문자데이터 또는 -1, read()
읽어온 데이터를 수신버퍼에서 삭제

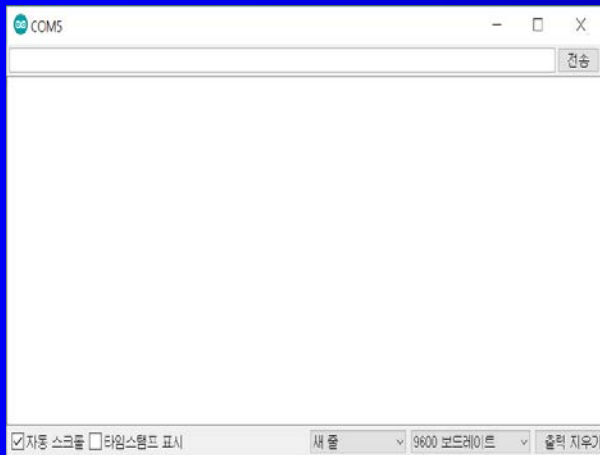
- int : 시리얼 통신 수신버퍼의 첫번째 바이트 데이터 또는 -1, peek()
읽어온 데이터를 수신버퍼에서 보존

- 사례

- if(Serial.available())>0; → 만일 시리얼 데이터가 수신되면 ?
 byte data = Serial.read();

■ 응용(1)

- 임의의 정수, 실수의 출력(시리얼모니터)
- 출력장치 : 시리얼모니터



< 출력장치 : 시리얼모니터 >

■ 소스코드

- 임의의 수 : $n=123$,
 $m=3.141592$
- 멤버함수
Serial.print
Serial.println 등



The screenshot shows the Arduino IDE interface. At the top, it says 'k1 | 아두이노 1.8.12'. Below that is a menu bar with '파일', '편집', '스케치', '툴', and '도움말'. A toolbar with icons for checkmark, undo, redo, and upload is visible. The main editor area shows the following C++ code:

```
k1

void setup() {
  // put your setup code here, to run once:

  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  int n=123;
  float m=3.141592;

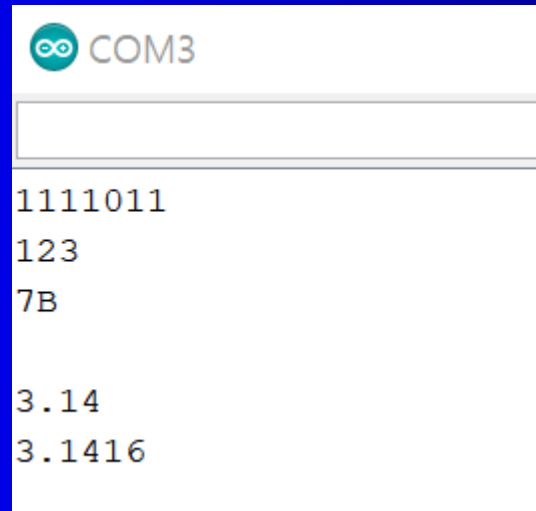
  Serial.println(n, BIN);
  Serial.println(n, DEC);
  Serial.println(n, HEX);

  Serial.println();
  Serial.println(m);
  Serial.println(m, 4);

  while(true);
}
```

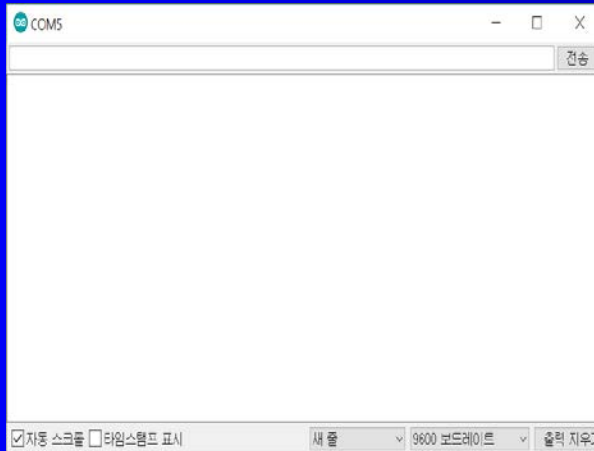
- 실행

- 출력창 : 시리얼모니터

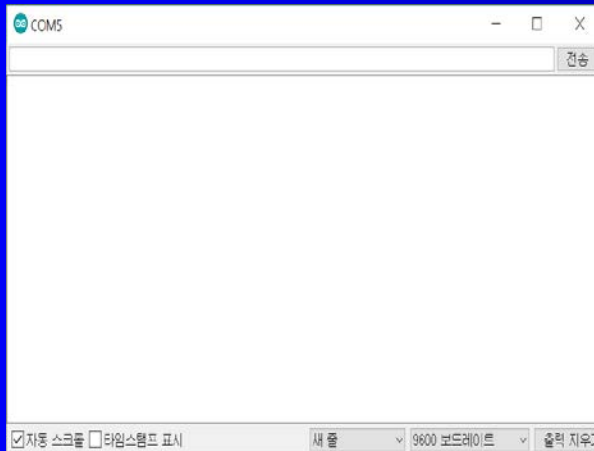


■ 응용(2)

- 문자입력(입력장치, 시리얼모니터) 및 문자출력(출력장치, 시리얼모니터)
- 프로그램 : 입력(문자 또는 문자열), 출력(문자 또는 문자열)



< 입력장치 : 시리얼모니터 >



< 출력장치 : 시리얼모니터 >

■ 소스코드

- 입력장치 및 출력장치 : 시리얼모니터
- 임의의 문자열 처리



```

k3 | 아두이노 1.8.12
파일 편집 스케치 툴 도움말

[Icons: Checkmark, Arrow, File, Upload, Download]

k3

void setup() {
  // put your setup code here, to run once:

  Serial.begin(9600);

}

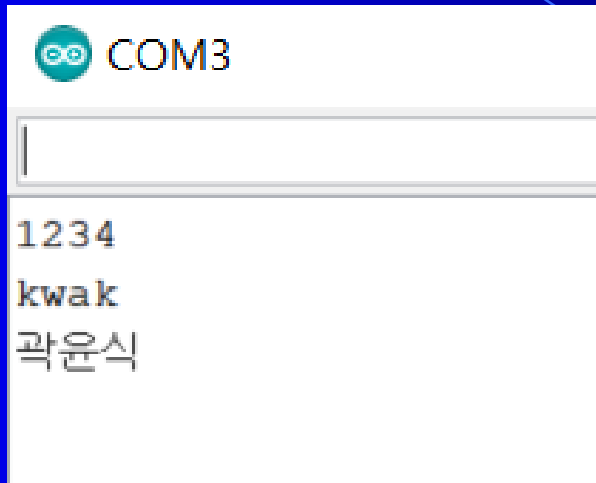
void loop() {
  // put your main code here, to run repeatedly:

  if(Serial.available() > 0){
  |
    Serial.write(Serial.read());
  }

}
```

- 실행

- 출력장치 : 시리얼모니터



□ String 클래스

▪ 기능

- 아두이노에서 문자열을 위한 클래스
 - 문자열처리
 - 숫자를 문자열처리(실수는 문자열로 변환 불가)

※ C에서는 문자열처리 -> **문자배열** 처리방식 사용

변수형	용도	범위	크기 (바이트)
bool	참과 거짓을 표현하는 정수	0,1	1
int	카운트용 정수	-32768~32767	2
unsigned long	가장큰 정수, 시간표시용	0~약42억	4
float	소수점포함	유효 6자리	4
char	문자한개	-128~127	1
String	문자열,클래스	연속된 문자묶음	가변

▪ String 클래스

- 기능

- string클래스의 개체(인스턴스)를 생성
(임의의 데이터형에 대해서 string변환)

- 구문-개체생성

String 개체;

String 개체 = 초기값;

- 구문-다른 변수형을 스트링으로 변환

String(val)

String(val, base)

String(val, decimalPlaces)

- val : string으로 변환할 변수

(데이터형 : string, char, byte, int, long, unsigned int,
unsigned long, float, double)

- base : (옵션)정수값 변환의 베이스
(HEX, DEC, BIN등)

- decimalPlaces : val이 float 또는 double인 경우

- 사례

- `String stringOne = "Hello String";` // using a constant
- `String stringOne = String('a');` // converting a constant char into a `String`
- `String stringTwo = String("This is a string");` // converting a constant string into a `String` object
- `String stringOne = String(stringTwo + " with more");`
// concatenating two strings
- `String stringOne = String(13);` // using a constant integer
- `String stringOne = String(analogRead(0), DEC);`
// using an int and a base
- `String stringOne = String(45, HEX);` // using an int and a base (hexadecimal)
- `String stringOne = String(255, BIN);` // using an int and a base (binary)
- `String stringOne = String(millis(), DEC);` // using a long and a base
- `String stringOne = String(5.698, 3);` // using a float and the decimal places

▪ string 멤버 함수

- string 클래스를 활용한 다양한 기능의 함수 제공
 - compareTo() : 사전순으로 문자열 정렬
 - equals(), equalsIgnoreCase() : en 문자열 비교

Operators

LANGUAGE [] (element access)
LANGUAGE + (concatenation)
LANGUAGE += (append)
LANGUAGE == (comparison)
LANGUAGE > (greater than)
LANGUAGE >= (greater than or equal to)
LANGUAGE < (less than)
LANGUAGE <= (less than or equal to)
LANGUAGE != (different from)
EXAMPLE String Tutorials

Functions

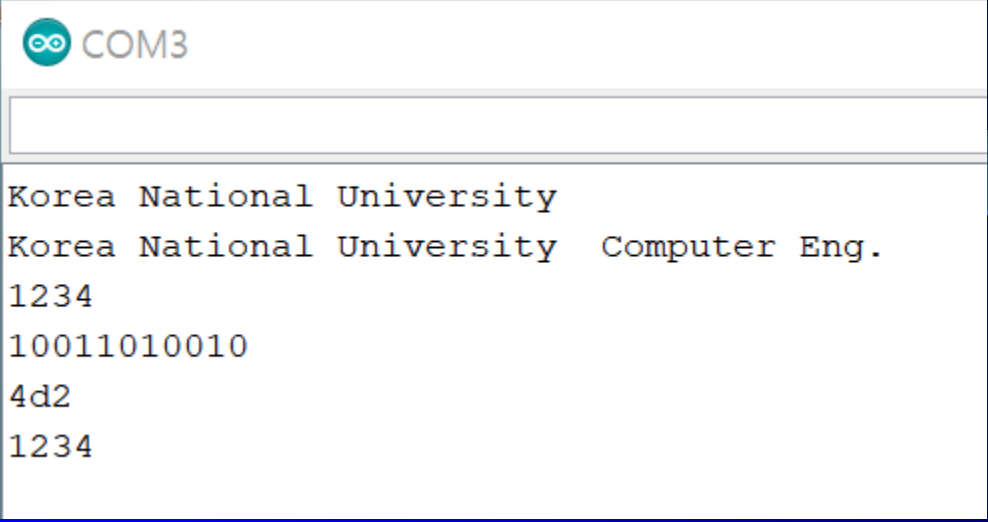
LANGUAGE charAt()
LANGUAGE compareTo()
LANGUAGE concat()
LANGUAGE c_str()
LANGUAGE endsWith()
LANGUAGE equals()
LANGUAGE equalsIgnoreCase()
LANGUAGE getBytes()
LANGUAGE indexOf()
LANGUAGE lastIndexOf()
LANGUAGE length()
LANGUAGE remove()
LANGUAGE replace()
LANGUAGE reserve()
LANGUAGE setCharAt()
LANGUAGE startsWith()
LANGUAGE substring()
LANGUAGE toCharArray()
LANGUAGE toDouble()
LANGUAGE toInt()

■ 응용(3)

- 스트링 클래스 기반 문자열처리
- 문자열처리
 - String
- 데이터형 변환
 - BIN, HEX, DEC

```
void loop() {  
    // put your main code here, to run repeatedly:  
  
    String  str1="Korea National University", str2="Computer Eng.";  
    int     k1=1234;  
  
    Serial.println(str1);  
    Serial.println(str1+" " +str2);  
  
    Serial.println(String(k1));  
    Serial.println(String(k1, BIN));  
    Serial.println(String(k1, HEX));  
    Serial.println(String(k1, DEC));  
  
    while(true);  
}
```

■ 실행



```
COM3  
  
Korea National University  
Korea National University  Computer Eng.  
1234  
10011010010  
4d2  
1234
```

The image shows a screenshot of a terminal window titled 'COM3'. The window has a white background and a thin grey border. Inside the window, the text is displayed in a monospaced font. The text consists of several lines: 'Korea National University', 'Korea National University Computer Eng.', '1234', '10011010010', '4d2', and '1234'. The text is left-aligned and appears to be the output of a serial communication.

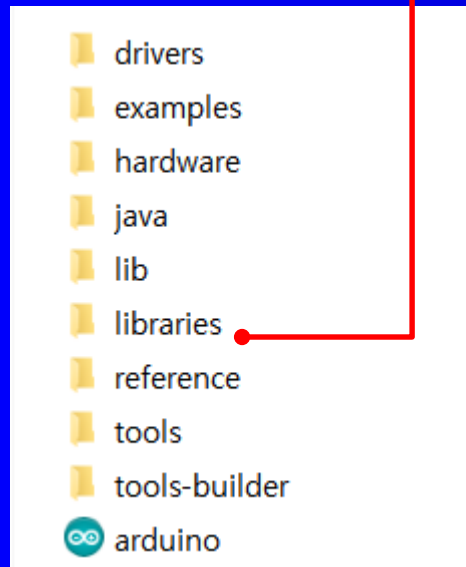
- 과제

시리얼 모니터로부터 임의의 문자열을 입력 받아
영문자를 소문자로 변환하는 프로그램을 작성하시요

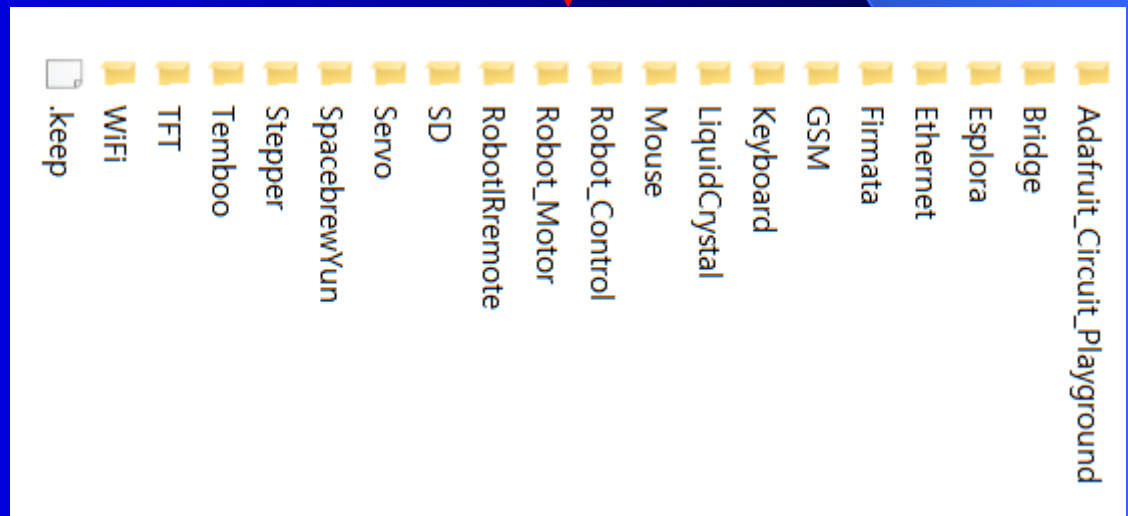
조료
○

별첨. 라이브러리 폴더생성

- 아두이노 라이브러리
 - 주변장치의 활용을 지원하는 라이브러리
 - 형식 : 클래스 형식으로 제공
 - 종류
 - 기본라이브러리 : 아두이노 공식 라이브러리
 - 확장라이브러리 : 사용자 라이브러리



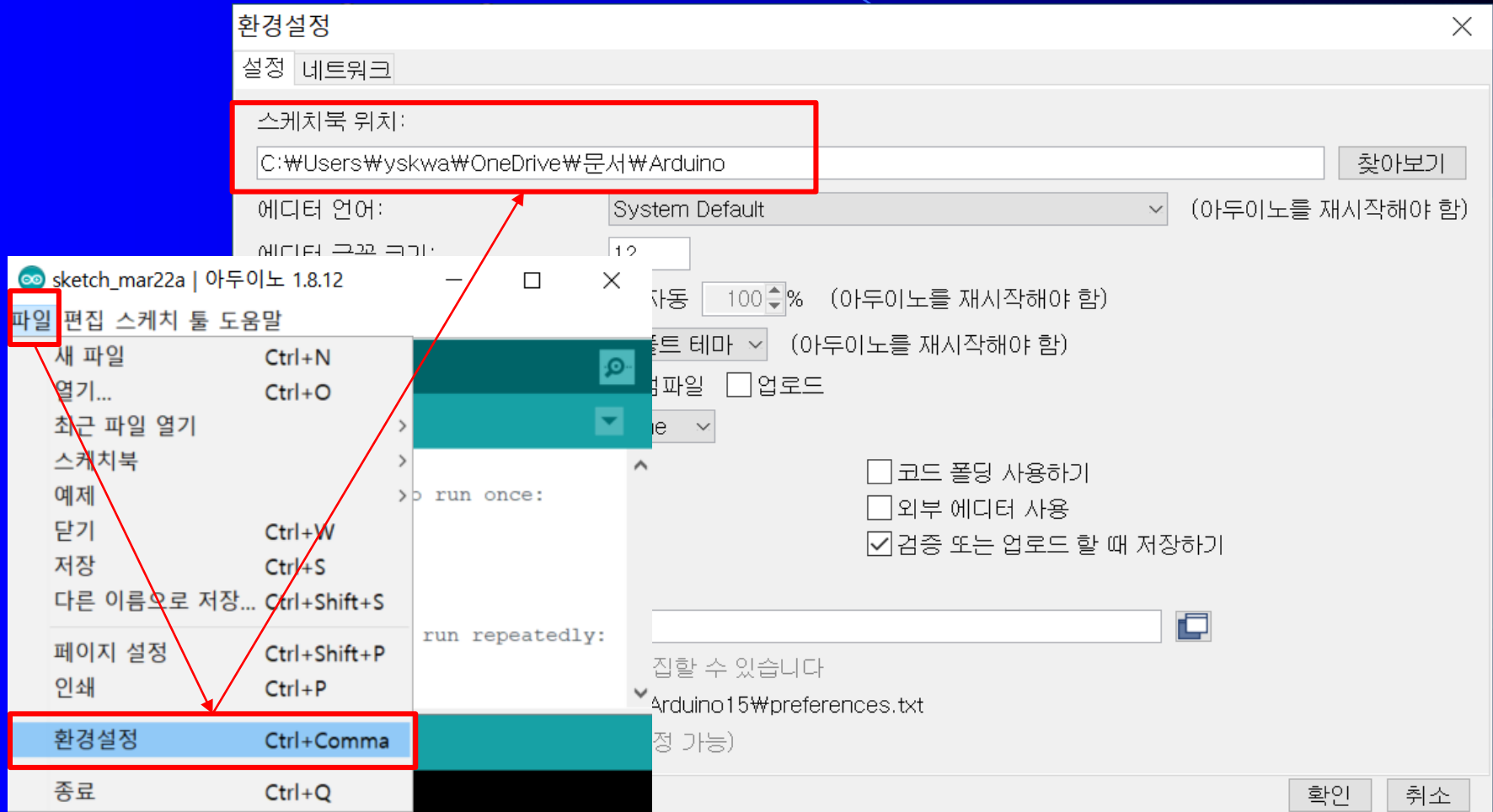
< 아두이노 설치폴더 >



< 라이브러리 폴더 >

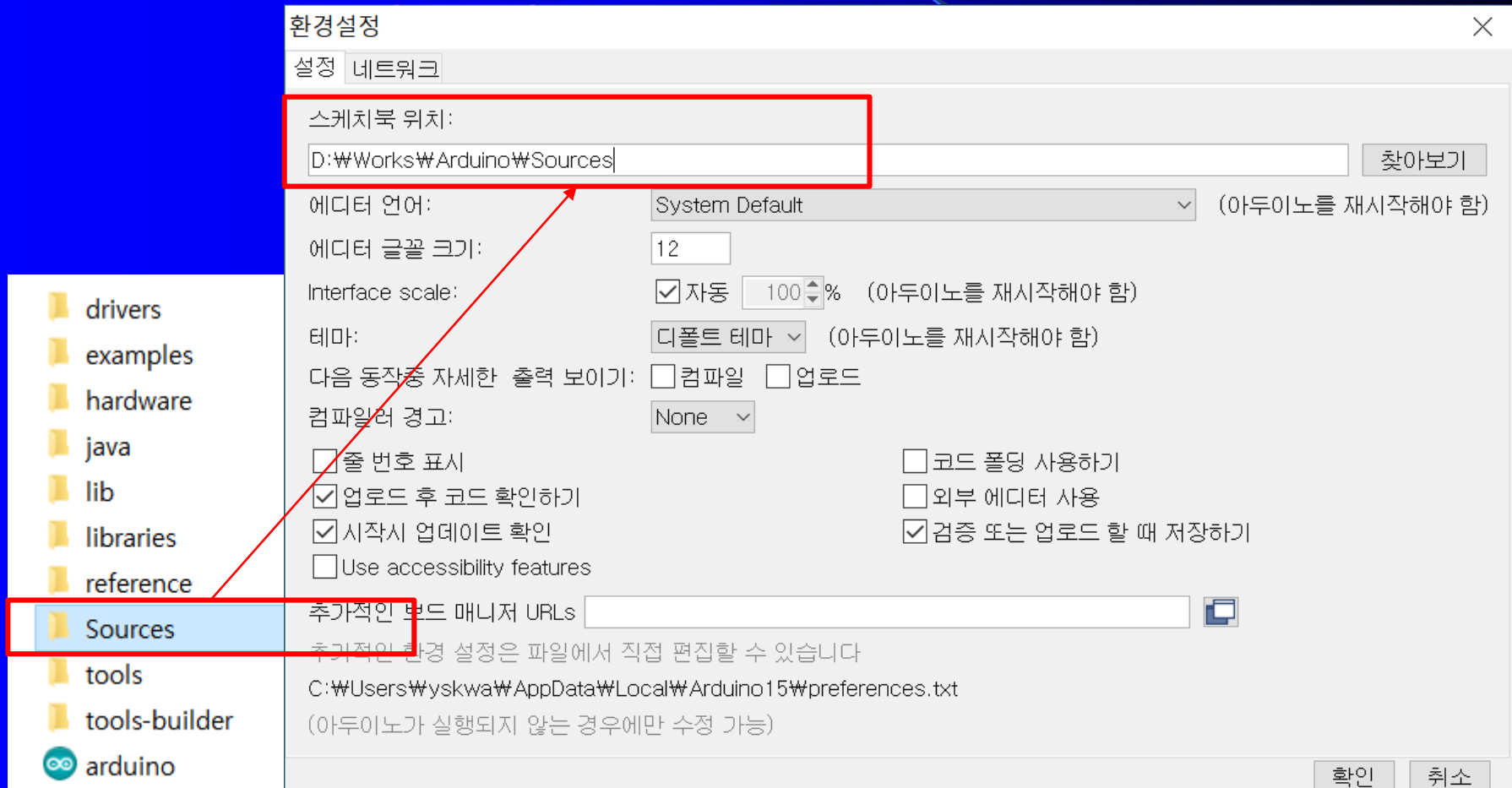
■ 확장 라이브러리 폴더 설정

- 현재, 스케치북 위치 확인 : 파일-> 환경설정



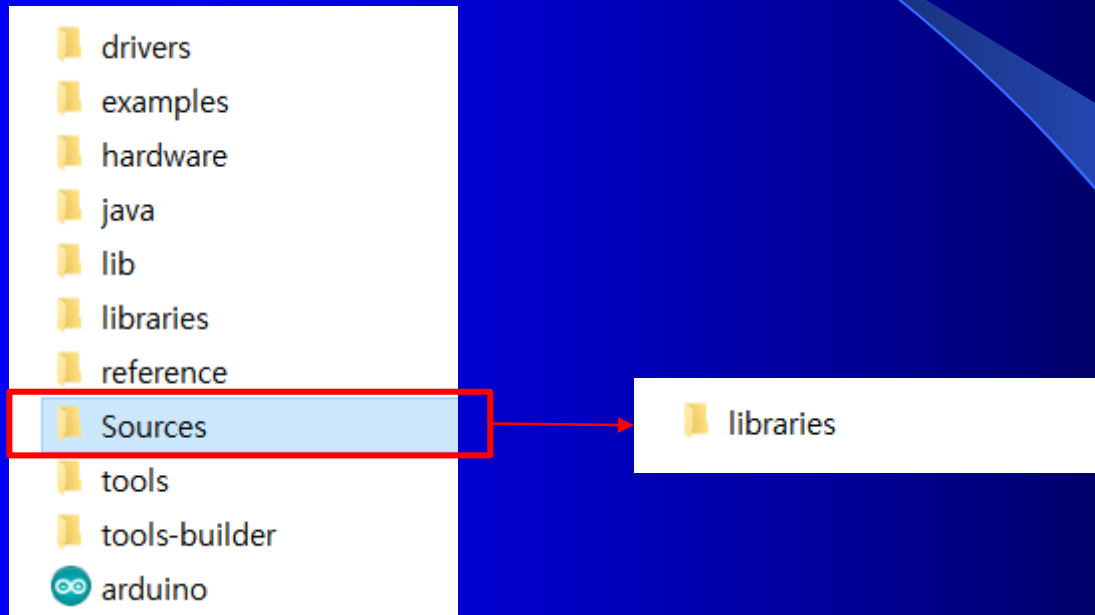
① 스케치북 디렉토리 설정

· D:\Works\Arduino\Sources



② 확장 라이브러리 디렉토리 설정

- D:\Works\Arduino\Sources\Libraries
- 스케치북 디렉토리 아래에 **별도의 라이브러리 디렉토리**



조료
○