

UART 통신

- 기초 학습
 - Serial
- UART 통신
 - 개 요
 - 결 선
 - 예 제 (1)~(3)
- 응용 학습



엣지아이랩

Basic learning

기초 학습

Serial

- Serial
 - **Arduino 보드나 컴퓨터와 기타 장치 사이의 통신에 사용**
 - 모든 Arduino 보드는 하나의 직렬 포트 Serial(UART 또는 USART) 제공
 - USB외에도 디지털 핀 0(RX), 1(TX)으로도 컴퓨터와 통신
 - 이 함수를 사용하는 경우 **디지털 입출력 핀 0과 1을 사용 불가**
- 시리얼 모니터 기능
 - Arduino IDE에서 도구 모음의 시리얼 모니터 버튼을 클릭
 - Serial.begin()에서 설정한 것과 동일한 통신 속도를 선택
- Arduino Uno
 - 1 개의 추가 시리얼 포트
 - 핀 0(RX)와 1(TX)
 - Uno 보드의 USB 시리얼 어댑터가 연결되지 않은 장치와 통신
 - 외부 TTL 시리얼 장치(3.3V 시리얼 장치)와 통신하기 위해 장치의 접지와 Arduino 보드의 접지, 장치의 RX 단자와 Arduino 보드의 TX 단자, 장치의 TX단자와 Arduino 보드의 RX 단자를 각각 연결
 - RS232 시리얼 포트에 직접 핀을 연결하지 말 것
 - +/- 12V에서 동작하므로 Arduino 보드가 손상 가능

Serial

^{①②③} SERIAL_8N1

- if(Serial)
 - 지정된 시리얼 포트가 준비되어 있는지를 확인
 - 준비되었으면 true, 그렇지 않으면 false를 반환
- available()
 - 시리얼 포트에서 읽을 수 있는 바이트(문자) 수를 반환
 - 수신 버퍼에 저장되어 있는 데이터
 - 수신된 데이터는 64바이트까지만 수신 버퍼에 저장
- begin()
 - 시리얼 데이터 전송을 위한 전송 속도와 옵션을 설정
 - begin(unsigned long baud) 또는 begin(unsigned long baud, byte config) 형태로 사용
 - Baud는 전송 속도
 - 컴퓨터와 통신하기 위해서 일반적으로 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, 115200의 속도 중 하나로 설정
 - config는 데이터 비트 수, 패리티 비트, 정지 비트 등을 설정
 - config의 디폴트 값은 SERIAL_8N1로 설정

① Data bit : 5, 6, 7, 8

② Parity bit: N(None), E(Even), O(Odd)

③ Stop bit: 1, 2

Serial

- `end()`
 - 시리얼 통신을 종료
 - 종료되면 디지털 핀 0(RX) 및 핀 1(TX)로 사용된 핀은 입출력 핀으로 사용 가능
 - 시리얼 통신을 다시 시작하기 위해서는 `begin` 함수를 호출
- `bool find(char *target)`
 - `target` 문자열을 찾을 때까지 시리얼 버퍼에서 데이터를 읽음
 - `target` 문자열이 있으면 `true`, 시간이 초과하면 `false`를 반환
- `bool findUntil(char *target, char *terminator)`
 - `target` 문자열이나 `terminator` 종료 문자열을 찾을 때까지 시리얼 버퍼에서 데이터를 읽어냄
 - `target` 문자열이나 `terminator` 종료 문자열이 있으면 `true`
 - 시간이 초과하면 `false`를 반환

Serial

- flush()
 - 시리얼 통신의 송신 버퍼에 있는 데이터가 전송이 완료할 때까지 대기
- parseFloat()
 - 시리얼 버퍼에서 첫 번째 유효한 실수를 반환
 - 현재 수신 버퍼가 비어있거나 실수가 발견되지 않으면 0을 반환
- parseInt()
 - 시리얼 버퍼에서 첫 번째 유효한 정수를 반환
 - 현재 수신 버퍼가 비어있거나 정수가 발견되지 않으면 0을 반환
- peek()
 - 수신 버퍼에 저장된 첫 번째 바이트 데이터를 반환
 - 데이터는 수신 버퍼에서 제거 되지 않으므로 여러 번 peek 함수를 호출하여도 똑같은 문자 반환
 - 수신 버퍼가 비어있는 경우 -1을 반환

Serial

- print(value) 함수
 - 사람이 읽을 수 있는 ASCII 텍스트 데이터를 시리얼 포트에 출력
 - 숫자는 각 자리의 ASCII 문자를 사용하여 출력
 - 실수(float)도 기본 소수점 2자리까지 ASCII 숫자로 출력
 - 바이트는 단일 문자로 전송
 - 문자와 문자열은 그대로 전송

```
Serial.print(78)           // "78"  
Serial.print(1.23456)      // "1.23"  
Serial.print('N')         // "N"  
Serial.print("Hello world") // "Hello world"
```

Serial

- print(value, format) 함수
 - 두 번째 매개 변수는 사용하는 format을 지정
 - 정수의 경우 출력형식 지정 가능
 - BIN(2진법)
 - OCT(8진법)
 - DEC(10진법)
 - HEX(16진법)
 - 실수의 경우 소수점 이하 자릿수를 지정 가능

```
Serial.print(78, BIN) -> "1001110"  
Serial.print(78, OCT) -> "116"  
Serial.print(78, DEC) -> "78"  
Serial.print(78, HEX) -> "4E"  
Serial.print(1.23456, 0) -> "1"  
Serial.print(1.23456, 2) -> "1.23"  
Serial.print(1.23456, 4) -> "1.2345"
```


Serial

- `println()`
 - `print()` 함수와 동일하게 사용
 - 문자출력 후 캐리지 리턴(ASCII 13 또는 `\r`)과 개행 문자(ASCII 10 또는 `\n`)를 추가로 출력
- `read()`
 - 수신된 시리얼 데이터를 읽음
 - 수신 버퍼에서 첫 번째 문자를 읽어 반환
 - 수신버퍼가 비어 있으면 -1을 반환
- `readBytes(*buffer, length)`
 - 수신된 시리얼 데이터를 읽어 버퍼에 저장
 - `length`에 지정된 바이트 수의 문자를 읽었거나 시간 초과가 발생하면 종료
 - 입력 받은 문자의 수를 반환
- `readBytesUntil(character, buffer, length)`
 - 수신된 시리얼 데이터를 읽어 버퍼에 저장
 - `length`에 지정된 바이트 수의 문자를 읽었거나 시간 초과가 발생한 경우와 `character` 문자가 발견하면 종료
 - 입력 받은 문자의 수를 반환

UART Communication

UART 통신

개요

- UART(Universal Asynchronous Receiver/Transmitter)
 - 보통 컴퓨터의 직렬 장치를 말하며 서로 데이터를 주고 받는 장치
 - UART 통신은 시리얼 기반의 통신 방식
 - 일반적으로 RS232C 프로토콜을 통해 원격지와 통신을 지원하는 방식
 - UART는 컴퓨터에게 RS-232C DTE 인터페이스를 제공
 - 모뎀이나 기타 다른 직렬장치들과 통신하거나 데이터 송수신 가능
- 데이터 전송
 - 컴퓨터로부터 병렬 회로를 통해 받은 바이트들을, 외부에 전달하기 위해 하나의 단일 직렬 비트 스트림으로 변환
 - 외부 전송을 위해 패리티 비트를 추가
 - 시작 비트와 정지 비트 추가

개요

- 데이터 수신
 - 내부로 전송할 때에는, 직렬 비트 스트림을 컴퓨터가 처리할 수 있도록 바이트로 변환
 - 수신되는 바이트들의 패리티를 확인
 - 패리티 비트 제거
 - 시작비트 정지비트 제거
- 직렬 디바이스 제어
 - 키보드나 마우스로부터 들어오는 인터럽트 처리
 - 진보된 UART는 일정량의 데이터 버퍼링 제공함으로써, 컴퓨터와 직렬 장치들의 데이터 스트림이 대응

RS232 통신

- 1969년 미국의 EIA(Electric Industries Association)에 의해서 정해진 표준 인터페이스
 - 직렬 2진 데이터를 교환하는 데이터 터미널 장비(DTE)와 데이터 통신장비(DCE) 간의 인터페이스의 제반 사항을 규정하는 것
 - RS: Recommended Standard
 - 232: 특정규격의 식별번호
 - C: 버전
- 직렬전송을 위한 규격
 - 병렬전송에서는 8비트가 동시에 8개의 데이터 선을 통해 전송되는 반면, 직렬전송은 한번에 한 비트씩 전송
 - CCITT에서는 RS-232C와 유사한 규격을 V.24라는 이름의 권고안으로 채택
 - ISO에서도 RS-232C와 호환성이 있는 ISO 2110 채택

RS232 통신 - 동작 원리

- 시리얼 포트
 - 외부 장치와 데이터를 주고받기 위한 인터페이스로 많이 사용
 - 통신 최소 단위: 1바이트 (= 8비트)
- RS-232C 동작 원리
 - 병렬을 직렬로 직렬을 병렬로 바꾸어 주는 작업 수행
 - 컴퓨터 내부에서는 데이터가 병렬 → 전화선은 직렬
- 스타트 비트와 스톱비트
 - 다른 데이터와의 혼란을 방지하기 위하여 앞뒤 한 비트씩 첨가
 - 10비트를 1바이트로 전송, 2400bps라 하면 초당 2400비트 전송 가능
- 기본 인터페이스
 - RXD, TXD 라인을 통해 신호 송수신
 - RS232 Transceiver를 통해 전송 전압을 끌어 올려 장거리 전송

RS232 통신 - 프로토콜

- 통신 프로토콜
 - RS-232C를 이용한 비동기식 전송 규약
- 통신 속도
 - 시간당 데이터를 전송할 수 있는 양
 - baud rate : 1초당 전송되는 변조된 신호의 수
- 스톱 비트
 - 데이터의 시작과 끝을 알리는 스타트와 스톱 비트
 - 전송을 시작할 경우 논리 1
 - 8비트를 전송한 후 스톱비트 전
 - 스타트 비트는 고정
 - 스톱 비트는 1과 1.5, 2비트 중 하나를 선택(일반적으로 1 사용)

RS232 통신 - 프로토콜

- 패리티 비트
 - 외적인 요인에 의한 오류 검출 목적 사용
 - 종류 : 짝수, 홀수, 사용하지 않음
 - 데이터 길이가 7인 경우에 8번째 비트를 패리티 비트로 이용
 - 2바이트 한글은 8비트 모두를 사용하여야 하기 때문에 패리티 사용불가
 - 패리티가 고정되어 있는 장비에서는 한글 전송 불가
- 자료 길이
 - 하나의 데이터를 전송하는데 필요한 데이터 길이(비트 수)
 - 종류 : 7과 8 비트
 - 2 바이트 한글 전송에서는 데이터 길이를 8로 설정
 - 일반적으로 8 사용

결선

- Arduino Uno의 업로드 포트(USB)와 PC의 USB 포트를 연결



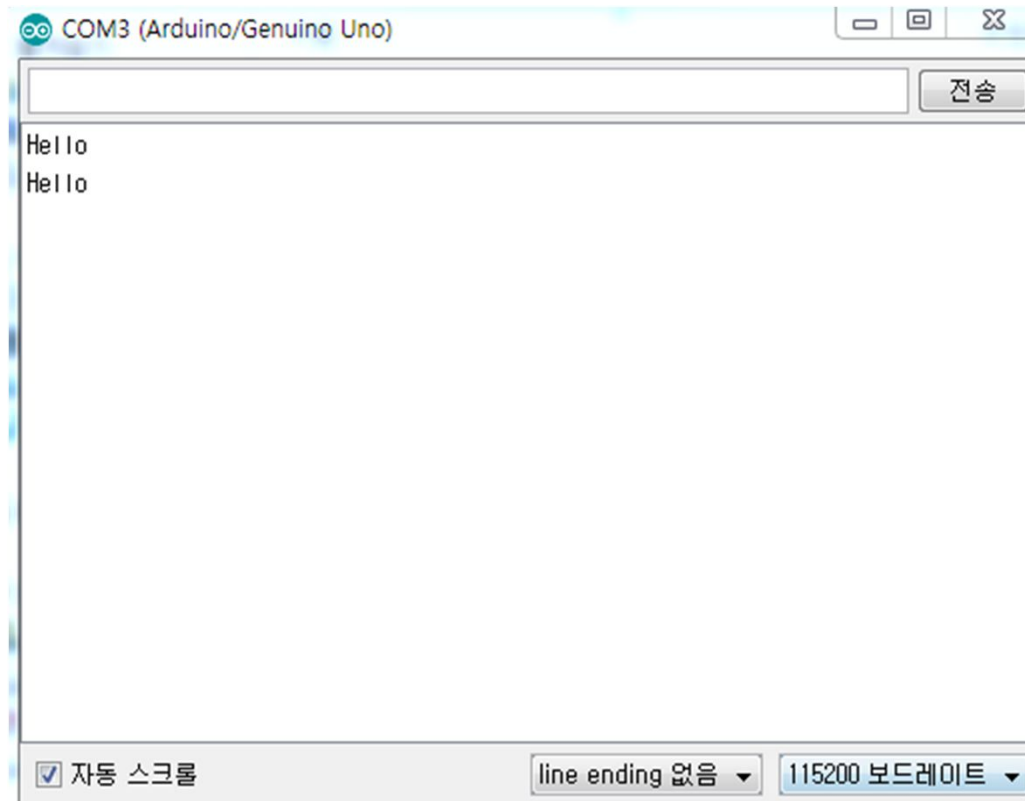
PC 연결(USB A type)



Uno 연결(USB B type)

예제(1)

- 예제
 - 시리얼 모니터에 "Hello"가 출력되게 하시오.



예제(1)

- 프로그램 설명

- Arduino IDE의 시리얼 모니터 프로그램을 사용하여 시리얼 통신

- 초기화 구문

Serial port(UART 0)를 전송속도 115200, 데이터 비트 8, 패리티 없음, 스톱비트 1로 설정

```
// UART setup baud 115200, data bit 8, parity None, stop bit 1
Serial.begin(115200); // same Serial.begin(115200, SERIAL_8N1)

Serial.write('H');
Serial.write('e');
Serial.write('l');
Serial.write('l');
Serial.write('o');
Serial.println();
Serial.write("Hello");
```

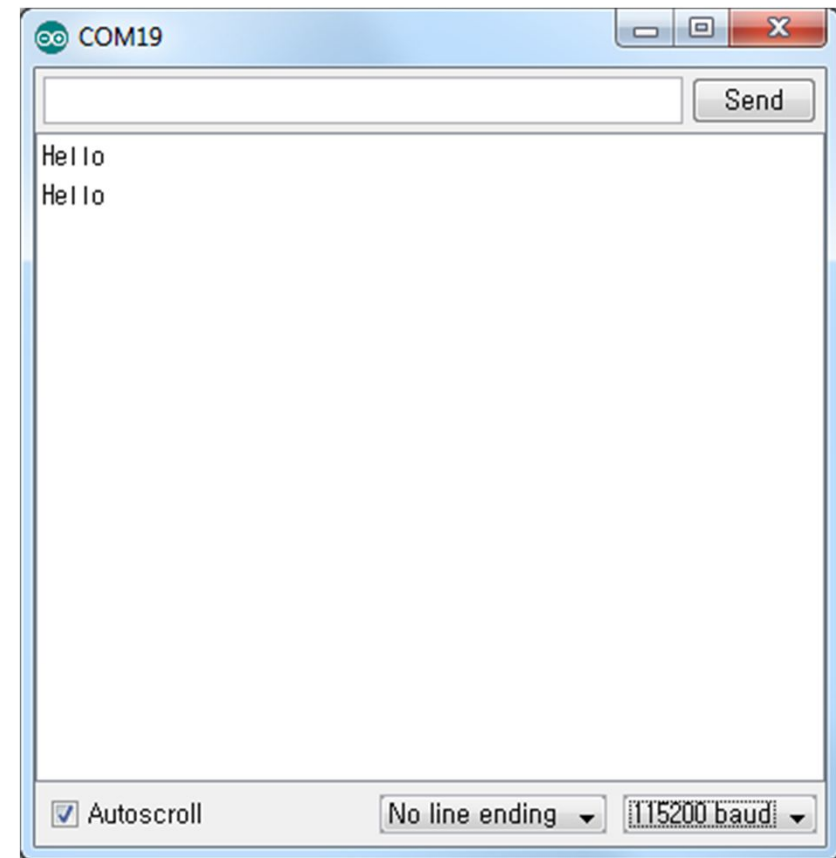
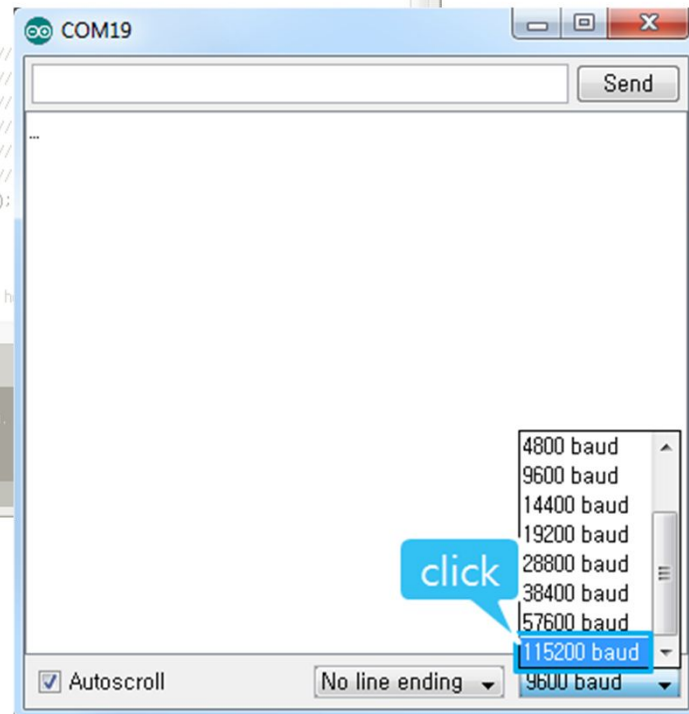
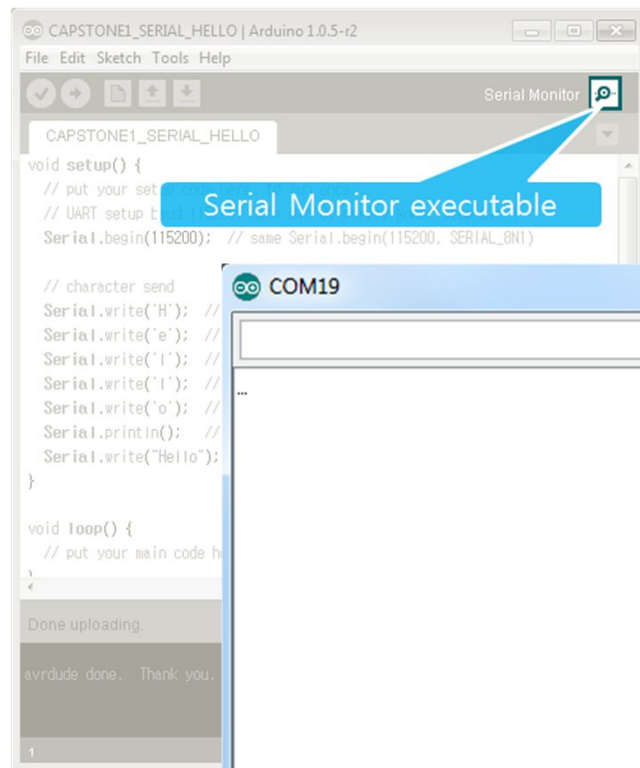
예제(1)

- 전체 소스코드

```
1. void setup() {  
2.   Serial.begin(115200);  
3.   Serial.write('H');  
4.   Serial.write('e');  
5.   Serial.write('l');  
6.   Serial.write('l');  
7.   Serial.write('o');  
8.   Serial.println();  
9.   Serial.write("Hello");  
10.}  
11.  
12. void loop() {  
13.}
```

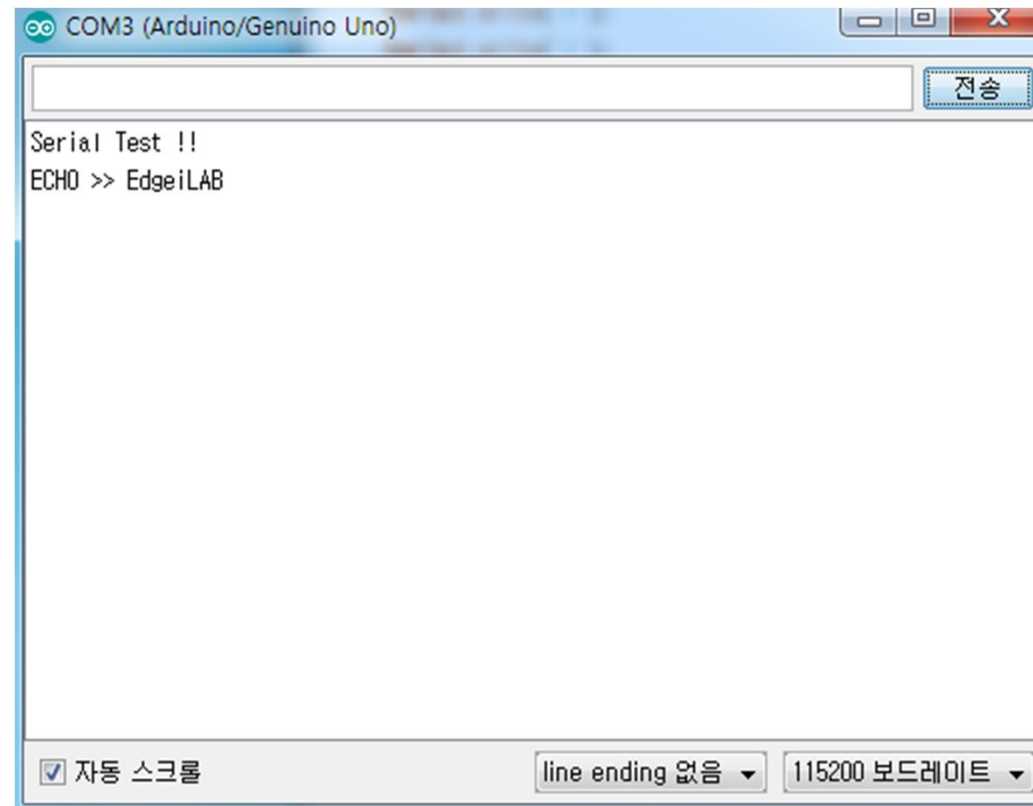
예제(1)

- 업로드 후 시리얼 모니터 실행
 - 속도 115200 선택
 - 다음과 같이 Hello 메시지 출력



예제(2)

- 예제
 - 시리얼 모니터에서 "Serial Test !!"와 "ECHO >>"를 출력된 후 "EdgeiLAB"을 전송하여 출력하시오.



예제(2)

- 프로그램 설명

- 초기화 구문

Serial port(UART 0)를 전송속도 115200, 데이터 비트 8, 패리티 없음, 스톱비트 1로 설정

```
// UART setup baud 115200, data bit 8, parity None, stop bit 1
Serial.begin(115200); // same Serial.begin(115200, SERIAL_8N1)
Serial.println("Serial Test !!");
Serial.print("ECHO >> ");
```

- loop 구문

수신 버퍼에 데이터가 수신되었는지 확인

데이터가 수신되어 1byte의 시리얼 데이터를 읽어 data에 저장, 저장된 데이터를 다시 시리얼로 전송

```
char data;
if(Serial.available() > 0)
{
    data = Serial.read();           // 1바이트 수신
    Serial.write(data);             // 1바이트 송신
}
```

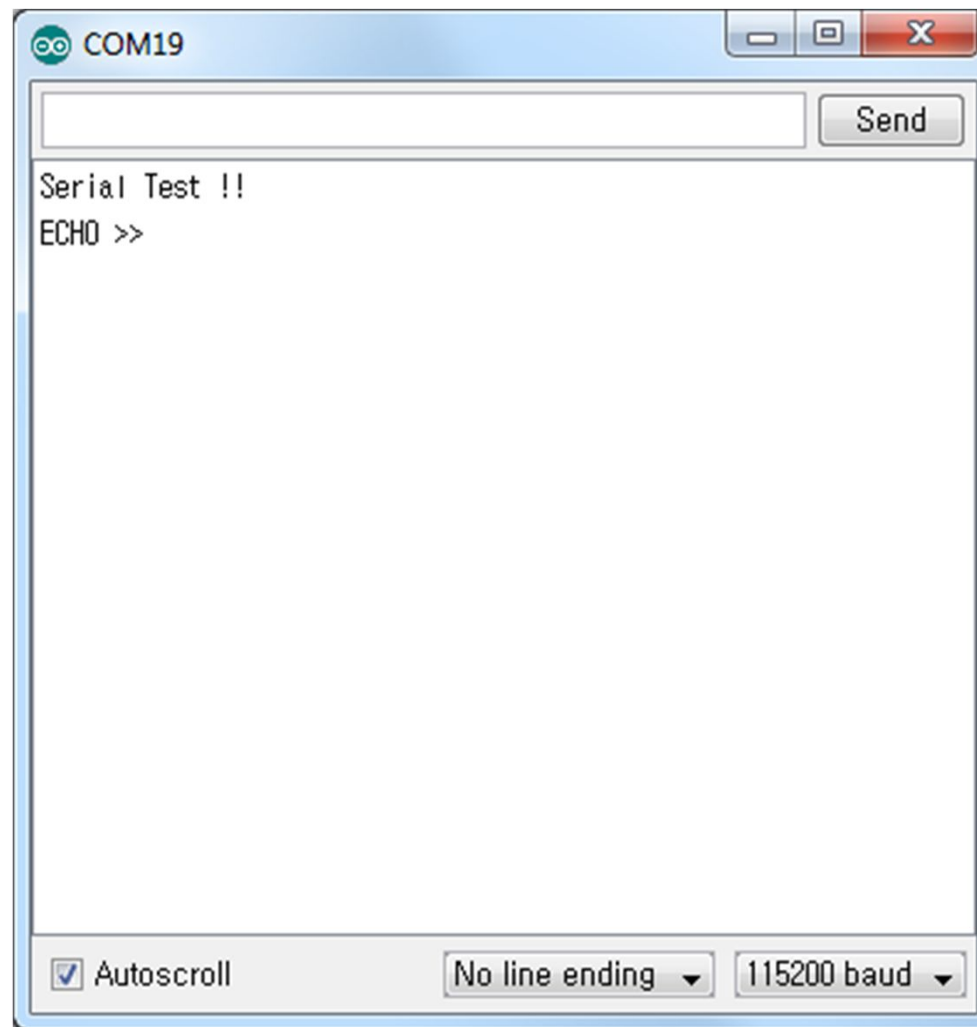
예제(2)

- 전체 소스코드

```
1. void setup() {  
2.   Serial.begin(115200);  
3.   Serial.println("Serial Test !!");  
4.   Serial.print("ECHO >> ");  
5. }  
6.  
7. void loop() {  
8.   char data;  
9.   if(Serial.available() > 0)  
10.  {  
11.    data = Serial.read();    // 1byte read  
12.    Serial.write(data);     // 1byte send  
13.  }  
14. }
```

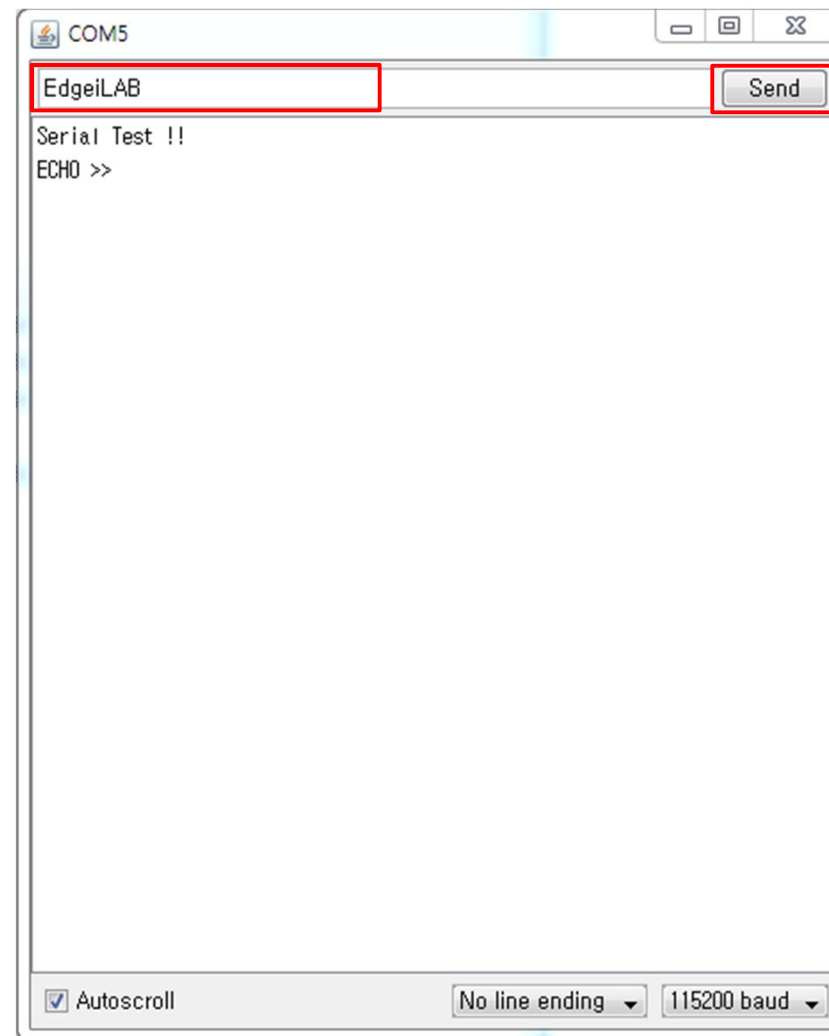

예제(2)

- 다음 그림과 같이 "Serial Test !!", "ECHO >>"가 출력



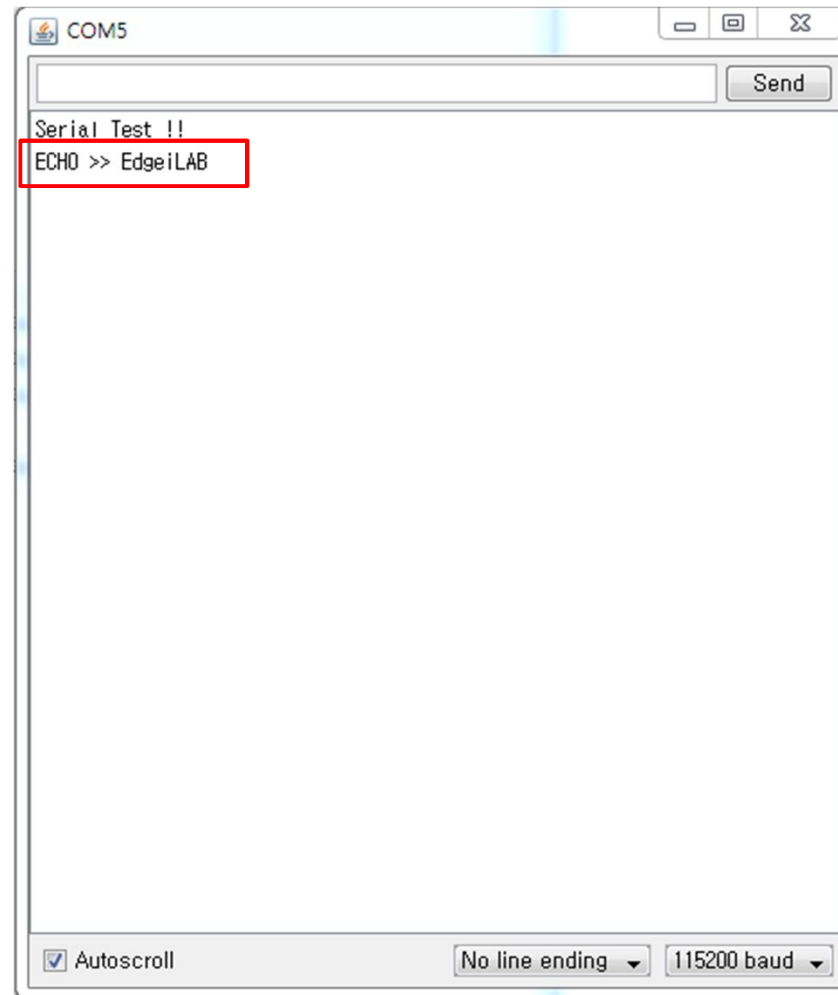
예제(2)

- 다음 그림과 같이 "EdgeiLAB"을 입력하고 Send를 클릭



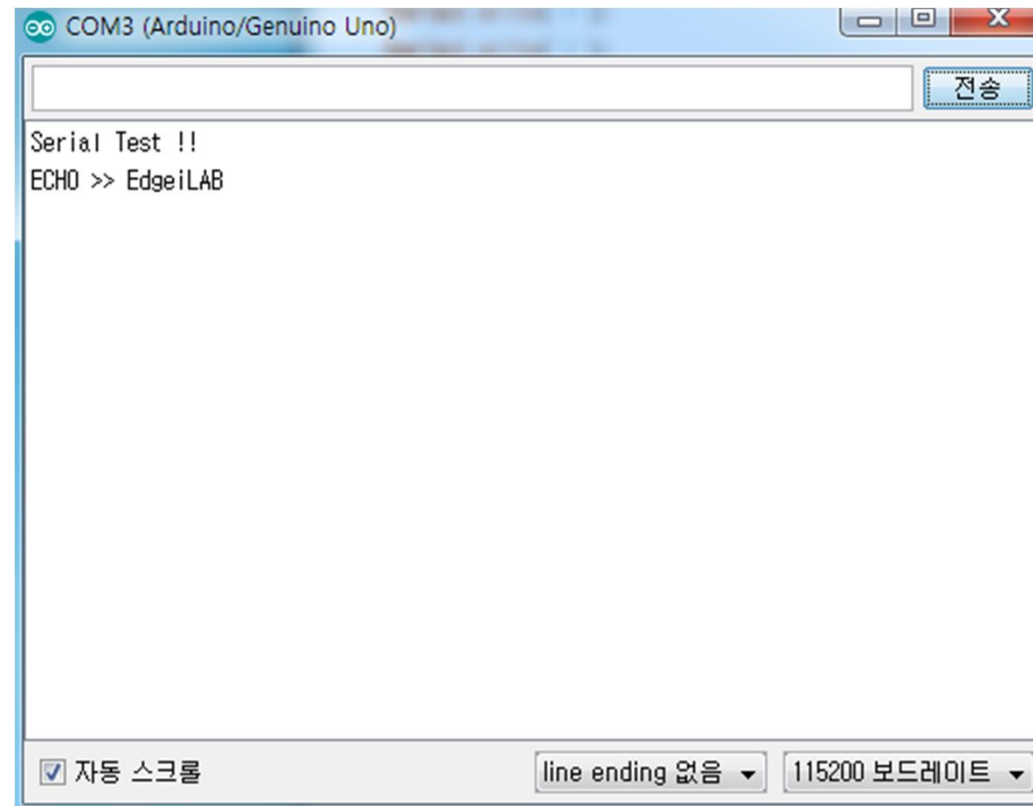
예제(2)

- 다음 그림과 같이 "EdgeiLAB"이라는 문자가 출력



예제(3)

- 예제
 - 시리얼 모니터에서 "Serial Test !!"와 "ECHO >>"를 출력된 후 "EdgeiLAB"을 전송하여 출력하시오. (단, serialEvent 함수를 사용)



예제(3)

- 프로그램 설명

- 초기화 구문

Serial port(UART 0)를 전송속도 115200, 데이터 비트 8, 패리티 없음, 스톱비트 1로 설정

```
// UART setup baud 115200, data bit 8, parity None, stop bit 1
Serial.begin(115200); // same Serial.begin(115200, SERIAL_8N1)
Serial.println("Serial Test !!");
Serial.print("ECHO >> ");
```

- serialEvent 함수

데이터가 수신되면 호출되는 함수

데이터가 수신되어 1byte의 시리얼 데이터를 읽어 data에 저장, 저장된 데이터를 다시 시리얼 모니터에 출력

```
void serialEvent(void)
{
    char data;
    data = Serial.read();
    Serial.print(data);
}
```

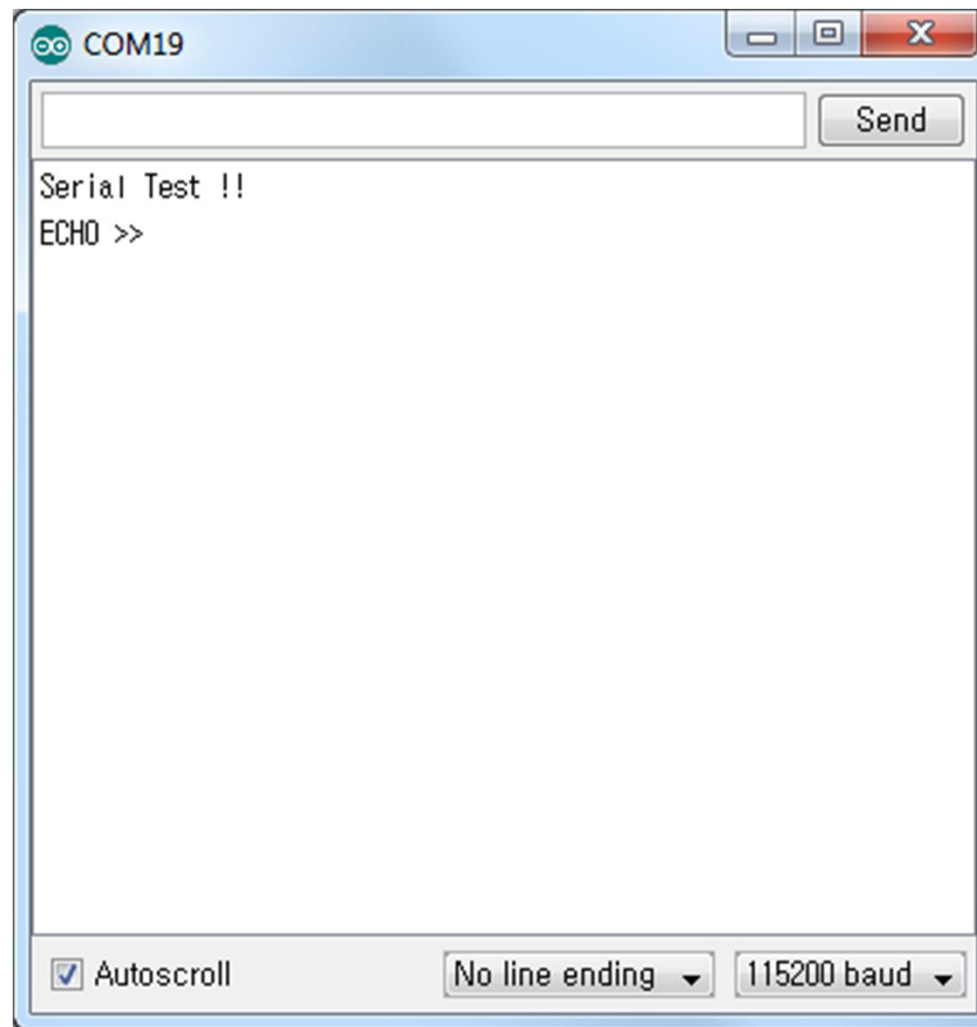
예제(3)

- 전체 소스코드

```
1. void setup() {  
2.   Serial.begin(115200);  
3.   Serial.println("Serial Test !!");  
4.   Serial.print("ECHO >> ");  
5. }  
6.  
7. void loop() {  
8. }  
9.  
10. void serialEvent(void) {  
11.   char data;  
12.   data = Serial.read();  
13.   Serial.print(data);  
14. }
```

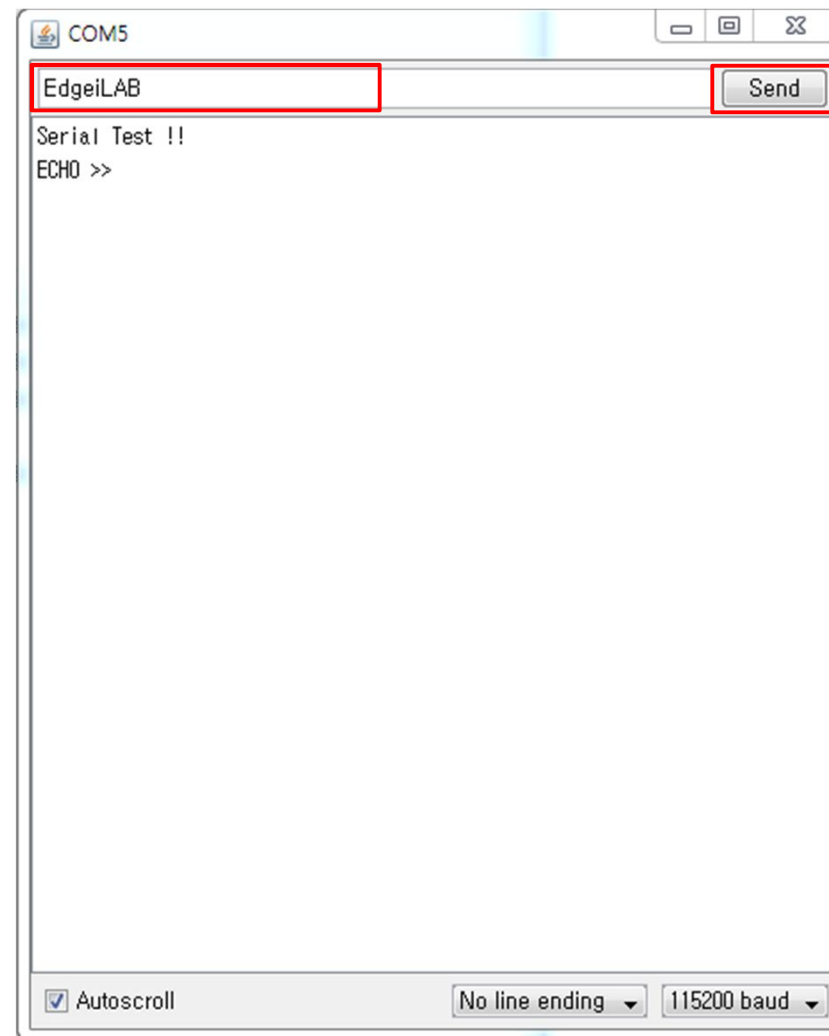
예제(3)

- 다음 그림과 같이 "Serial Test !!", "ECHO >>"가 출력



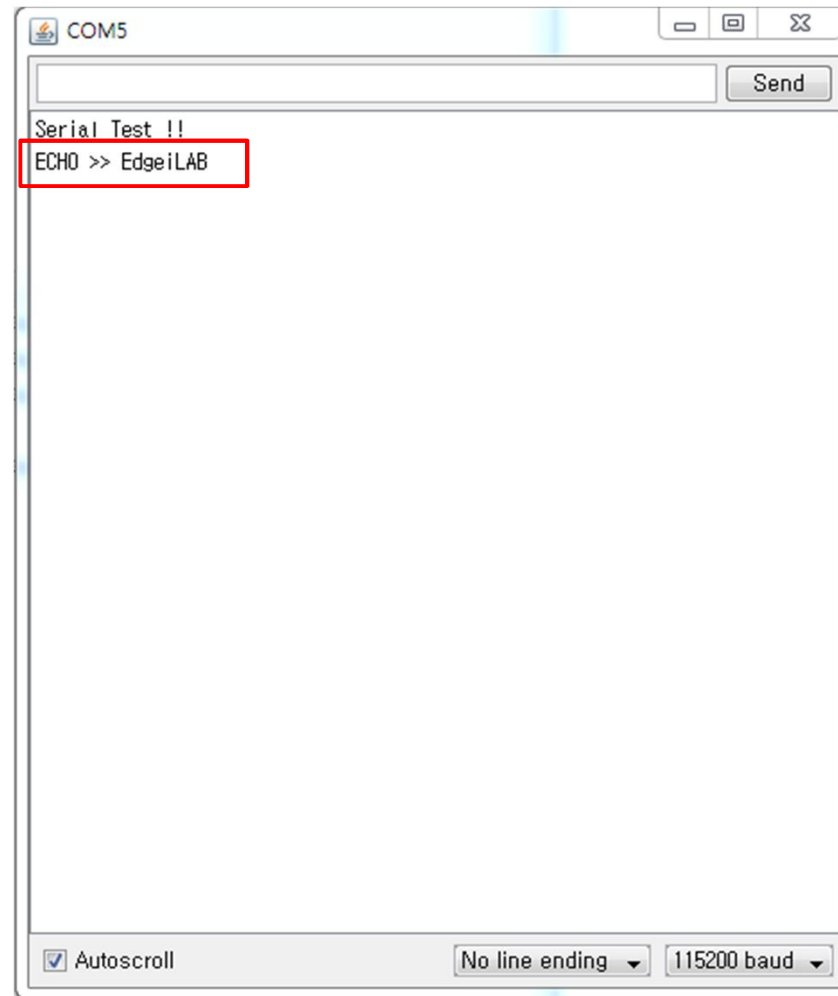
예제(3)

- 다음 그림과 같이 "EdgeiLAB"을 입력하고 Send를 클릭



예제(3)

- 다음 그림과 같이 "EdgeiLAB"이라는 문자가 출력



Application practice

응용 실습

응용 실습

- 응용 문제
 - 시리얼 모니터에서 'O'와 'X'를 전송해서 LED를 On/Off 하시오.
- 구성
 - Arduino Uno
 - LED/Switch 모듈

