

# Serial 함수와 명령어 (1)

---

 [kocoafab.cc/tutorial/view/501](http://kocoafab.cc/tutorial/view/501)

2015-07-17 15:55:37

## 개요

---

기본적으로 Serial함수에 대해 알고 계시고, 자주 사용하는 함수는 Serial.begin(), Serial.print()/Serial.println(), Serial.available(), Serial.read() 함수 일 것입니다.

저 위의 함수들만 사용할 수 있으면 기본적으로 시리얼 통신을 이용하는데 전혀 문제가 없지만, 밑의 사진을 보시면 아두이노에서 Serial을 사용하는데 다양한 도움을 주는 함수들을 많이 제공합니다.

밑의 함수들을 보시면 시리얼 통신할 때 알아두면 유용한 함수들이 많은데요, 이번 컨텐츠에서는 한국형 아두이노 오렌지보드에서 사용할 수 있는 다양한 Serial함수들에 대해 알아보고 어떻게 동작하는지 알아 보겠습니다.

위의 함수들을 컨텐츠에서 전부 다루지는 못하지만 알아두시면 매우 유용한 함수들 위주로 하나씩 소개하도록 하겠습니다.

## Functions

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

## 1. Serial.Write()

---

### 사용 방법

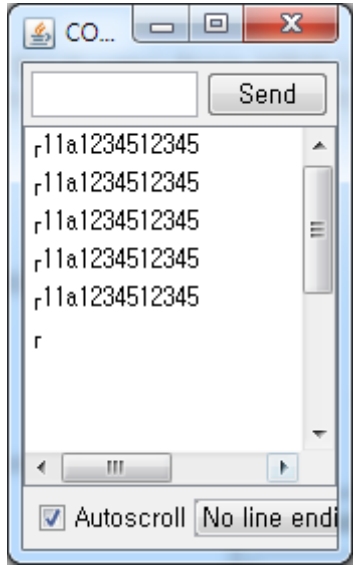
---

- `Serial.write(val)`
- `Serial.write(str)`
- `Serial.write(buf, len)`

## 변수

- val : byte형 데이터
- str : string형 문자열
- buf : byte형 배열
- len : 배열의 길이

## 예제 소스 코드

소스 코드	결과 화면
<pre>void setup(){   Serial.begin(9600); }  void loop(){   const uint8_t temp[5] = {'1', '2', '3', '4', '5'};   Serial.write(1); // write()함수로 1 전송   delay(500);   Serial.write(49); // write()함수로 49 전송   delay(500);   Serial.print(1); // print()함수로 1 전송   delay(500);   Serial.write('a'); // write()함수로 'a' 전송   delay(500);   Serial.write(temp, 5); // write()함수로 temp배열을 5만큼 전송   delay(500);   Serial.write("12345"); // write()함수로 string값 전송   delay(500);   Serial.write("\n"); // 줄바꿈   delay(500); }</pre>	

## 설명

Serial.write()함수는 Serial.print() 함수와 마찬가지로 데이터 값을 시리얼 통신으로 송신하는 기능을 합니다.

위의 소스코드 결과 화면을 보시면 write()함수와 print() 함수가 모두 1 을 전송하는데(첫번째와 세번째) write()함수는 이상한 값을 출력합니다. write(49)를 출력해야 제대로 값이 1이 나오는것을 확인 할 수 있습니다.

print()함수는 1값을 아스키 코드로 해석해서 데이터를 전송하지만, write()함수는 1값을 그대로 전송해서 시리얼 모니터에서 아스키 코드로 해석해서 보여주므로 다르게 보이는 것입니다.

아스키 코드값 49가 문자 1이어서 49를 write()함수로 전송해야 시리얼 모니터에서 아스키 코드 문자 1로 해석하고 문자 1을 보여주는 것입니다.

write()함수는 기본적으로 시리얼 모니터에 표시할 때는 사용하지 않지만 블루투스나 지그비 간 통신을 할때 문자 외에 직접 데이터 값을 보내야 할 경우 사용합니다.

## 2. Serial.end()

---

### 사용방법

---

- Serial.end();

### 설명

---

Serial.end는 Serial통신을 사용하지 않도록 하는 함수입니다. Serial.begin()이 Serial 통신을 사용하기 위해 사용되는 것이라면, Serial.end()는 Serial에 사용되는 핀(디지털 0, 1번핀)을 일반 디지털 핀으로 사용하게 하는 함수입니다.

## 3. Serial.parseInt()

---

### 사용방법

---

- Serial.parseInt();

#### 반환값

- Long타입 숫자 반환(입력 받은 값이 숫자일 경우)
- 숫자 값이 아닐 경우 0을 반환

### 예제 소스 코드

---

소스 코드	결과 화면
-------	-------

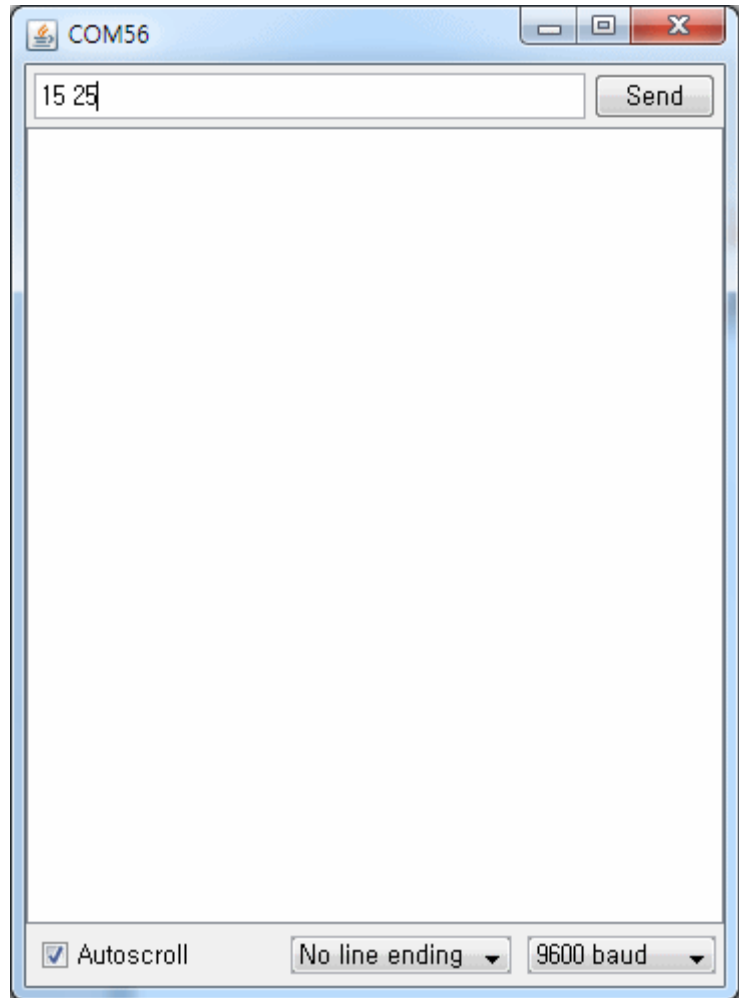
```

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

void loop(){
  if(Serial.available()){
    long value1 =
    Serial.parseInt();
    long value2 =
    Serial.parseInt();

    Serial.print("value 1 :
");
    Serial.println(value1);
    Serial.print("value 2 : ");
    Serial.println(value2);
    Serial.print("value1 + value2 =
");
    Serial.println(value1 + value2);
  }
}

```



## 설명

Serial.parseInt()함수는 Serial.read()함수와 기능은 비슷합니다. Serial.read()함수는 보내준 데이터 값을 그대로 읽어오지만 parseInt()함수는 보낸 값을 long 형태의 숫자값으로 바꿔서 저장하게 됩니다.

위의 예제 결과 화면을 보시면 모니터 창에 15 25를 입력 하면 그 값을 parseInt()로 받고 그 값을 숫자로 바꿔서 저장하고, 저장한 2개의 값을 더해서 결과 값을 보여주게 됩니다.

위에 15 값을 read()로 받게 되면 '1' '5' 두개의 문자를 받지만 parseInt()로 받아서 long값 15를 받게 되었습니다. 참고로 숫자 값이 아닐 경우 0을 반환하게 되니 이를 이용하여 숫자와 문자를 구분하는 것으로도 이용할 수 있습니다.

## 4. Serial.parseFloat()

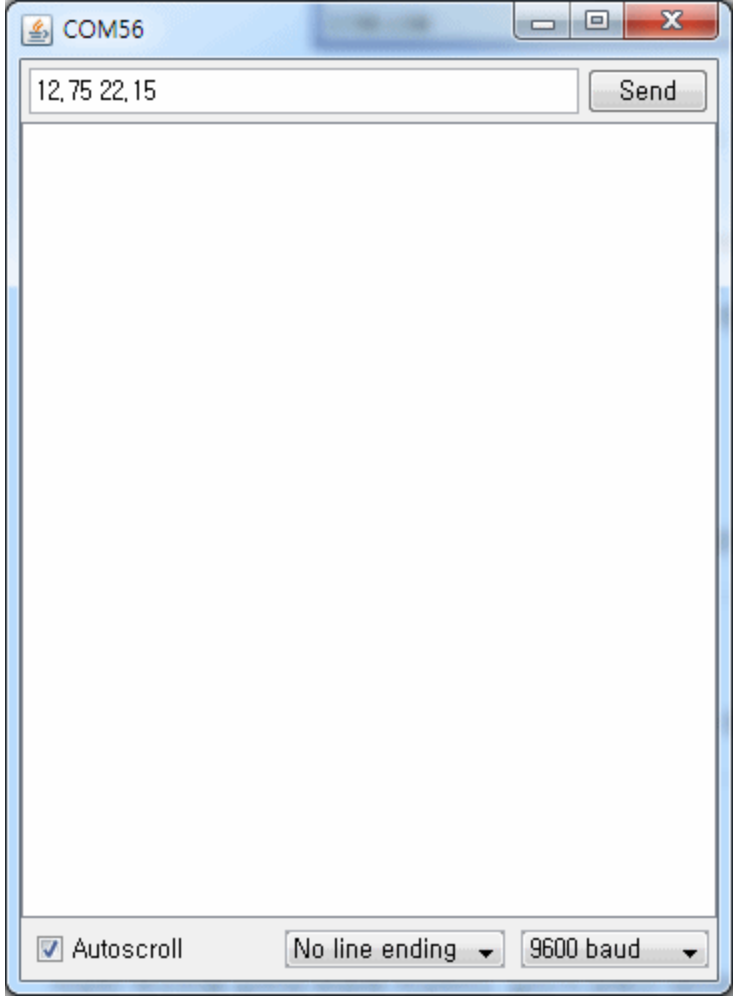
### 사용방법

- Serial.parseFloat()

## 반환값

- Float형 숫자 반환(입력 값이 숫자일 경우)
- 숫자 값이 아닐 경우 0을 반환

## 예제 소스 코드

소스 코드	결과 화면
<pre>void setup(){   Serial.begin(9600); }  void loop(){   if(Serial.available()){     float value1 =     Serial.parseFloat();     float value2 =     Serial.parseFloat();      Serial.print("value 1 : ");     Serial.println(value1);     Serial.print("value 2 : ");     Serial.println(value2);     Serial.print("value1 + value2 = ");     Serial.println(value1 + value2);   } }</pre>	

## 설명

위에서는 정수형태의 숫자만 처리했다면, 이번엔 소숫점까지 처리 할 수 있는 함수입니다.

parseInt()와 같은 기능이지만 반환값이 long값이 아닌 float값 입니다. 간단하게 설명하면 parseInt()는 정수형태의 값으로 변환하고, parseFloat()는 소수형태 값으로 변환합니다.

(두개의 예제 소스 코드 결과화면을 비교해 보시면 간단하게 이해하실 수 있습니다.)

## 5. Serial.readBytes()

## 사용방법

- Serial.readBytes(buffer, length)

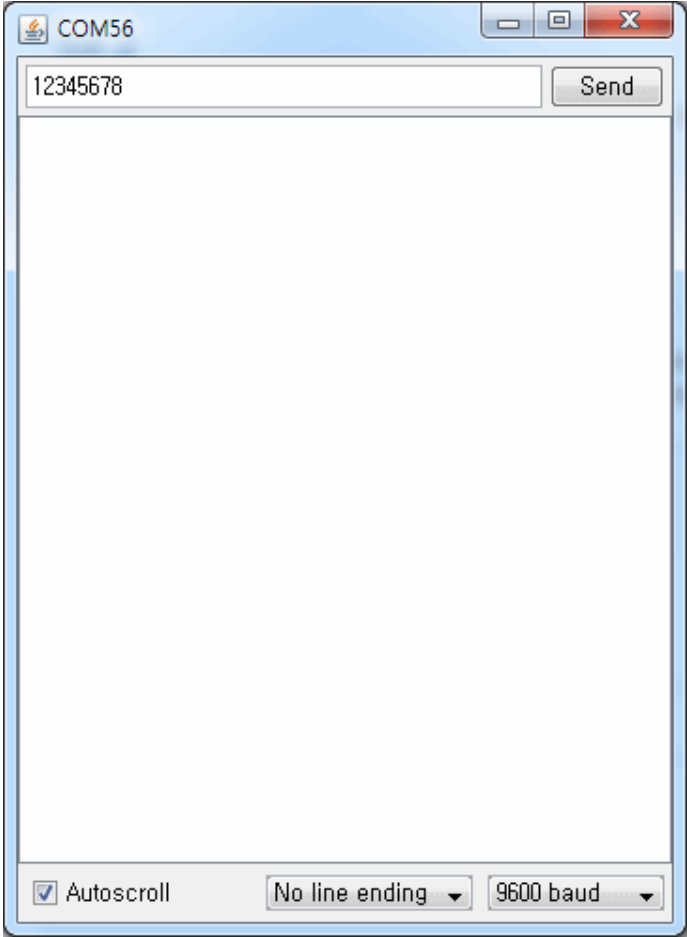
### 변수

- buffer : 입력받은 데이터를 저장할 buffer
- length : 입력받은 문자의 길이

### 반환

- 읽어온 데이터의 길이 byte값

## 예제 소스 코드

소스 코드	결과 화면
<pre>void setup(){   Serial.begin(9600); }  void loop(){   char temp[100];   if(Serial.available()){     byte leng =     Serial.readBytes(temp, 20);      Serial.print("Input data Lenght : ");     Serial.println(leng);      for(int i = 0; i &lt; leng; i++){       Serial.print(temp[i]);     }     Serial.println();   } }</pre>	

## 설명

Serial.readBytes() 함수는 지정한 숫자만큼이나 일정 시간동안 읽은 데이터를 buffer에 저장해주는 함수입니다.(일정 시간은 밑의 serial.setTimeout() 함수를 참조하시면 됩니다.)

지정한 숫자만큼 데이터가 입력되지 않으면 입력받은 갯수만큼만 저장하고, 지정한 숫자 이상으로 데이터가 들어 올 경우 지정된 숫자만큼 buffer에 저장하고, 그 이후에 들어온 값을 따로 buffer에 저장합니다.

함수의 반환값은 버퍼에 저장한 문자의 갯수를 byte형태로 반환됩니다.

## 6. Serial.readBytesUntil()

---

### 사용방법

---

- Serial.readBytesUntil(char, buffer, length)

#### 변수

- char : 종료 문자열
- buffer : 입력받은 데이터를 저장할 buffer
- length : 입력받을 문자의 길이

#### 반환

- 읽어온 데이터의 길이 byte값

### 예제 소스 코드

---

소스 코드	결과 화면(기준값 'k')
-------	----------------



```

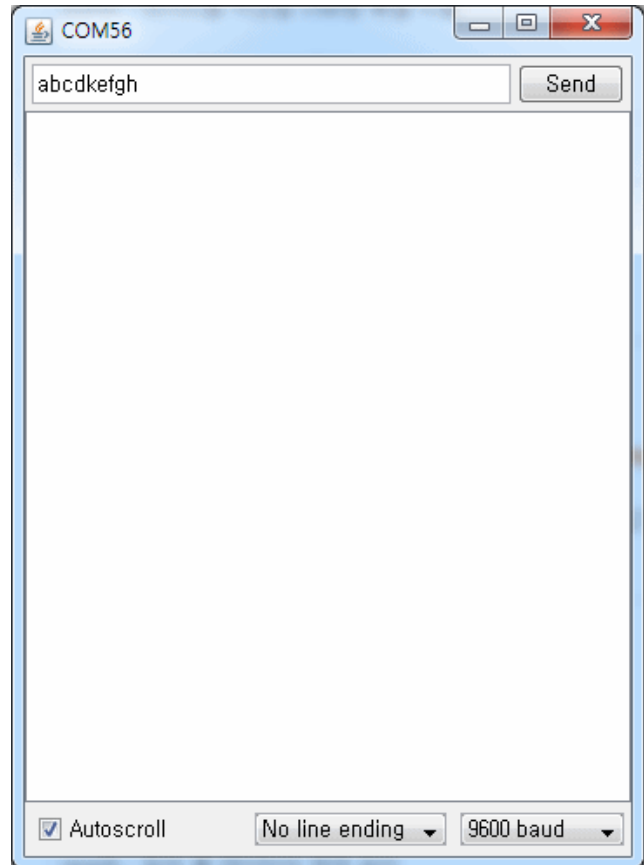
void setup(){
  Serial.begin(9600);
}

void loop(){
  char temp[100];
  if(Serial.available()){
    byte leng = Serial.readBytesUntil('k',
temp, 20);

    Serial.print("Input data Lenght : ");
    Serial.println(leng);

    for(int i = 0; i < leng; i++){
      Serial.print(temp[i]);
    }
    Serial.println();
  }
}

```



## 설명

Serial.readBytesUntil() Serial.readBytes() 함수와 같이 지정한 숫자만큼이나 일정 시간동안 읽은 데이터를 buffer에 저장해주는 함수입니다.(일정 시간은 밑의 serial.setTimeout() 함수를 참조하시면 됩니다.)

다른 점은 맨 처음 변수에 저장한 특정 문자가 들어올때까지 데이터를 읽어 오게 됩니다.(혹은 지정한 숫자만큼 데이터가 입력이 되거나)

기본은 readBytes()함수와 같습니다. 지정한 숫자보다 적게 들어 올경우 입력받은 갯수 만큼만 저장되고, 초과 될 경우 나눠서 저장이 됩니다. 결과 화면에 보이듯이 특정 문자가 들어 올경우는 특정 문자 앞/뒤로 나뉘어서 버퍼에 저장되게 됩니다.

(위에 예제는 'k'값이 특정 문자로 지정해서 실행한 예제 입니다. 입력값은 abcd**k**efgh 입니다.)

함수의 반환값은 버퍼에 저장한 문자의 갯수를 byte형태로 반환됩니다.

**\* 주의할 점은 특정 문자로 지정한 문자는 버퍼에 따로 저장되지 않습니다.**



