Serial 통신 때 String 사용 (아두이노) — Steemit

steemit.com/kr-arduino/@codingman/serial-string

- 온라인 가상시뮬레이터 : https://www.tinkercad.com
- 참고 :
  Serial 통신 제어 (아두이노)
  Bluetooth 통신 제어(아두이노)
- 공개회로도 : https://www.tinkercad.com/things/1xmY92OZYdn

오늘은 간단히 Serial 통신을 할 때 String으로 접근하는 방법을 알아보고자 합니다. 지금까지는 간단히 Sensor의 값을 Serial 통신을 통해 값을 전송할 때는 하나의 값만 주고 받아 왔습니다. 조이스틱에서는 두개의 x,y값을 만들어 내지만 이것 역시 간단히 하나씩 키 값으로 해서 전송해 왔는데 이제는 여러개의 값을 한번에 전송하고 그 값을 수신하는 쪽에서 분리해 내는 과정을 설명하면 좋을 것 같아서 오늘 Post 주제로 결정했습니다. 대부분 실험이 한개의 데이터만 Serial 통신에 주고 받기 때문에 필요 없을 수 도 있지만 한번은 접해 놓으셔야 나중에 필요하실 때 기억해 내서 활용 할 수 있으니깐 한번은 접해 보셨으면 합니다. 이제부터서 String에 대해 이야기를 하겠습니다.

1. Serial 통신 함수

[시리얼 통신]

- Serial.begin(9600) : 시리얼 통신 시작(9600 통신속도)
- Serial.println(값) : 시리얼모니터 출력
- Serial.available() : 시리얼통신으로 데이터가 들어왔는지 상태 확인
- Serial.read() : Int형으로 데이터를 읽음

[Bluetooth(SoftwareSerial 통신)]

```
#include <SoftwareSerial.h>
```

- SoftwareSerial mySerial (rx, tx) : 소프트시리얼 객체선언(rx(수신), tx(전송))
- mySerial.begin(9600) : 시리얼 통신 시작(예로 9600 통식속도를 사용해 봤네요.)
- mySerial.println(값) : 데이터 전송
- mySerial.read() : Int형으로 데이터를 읽음

[기본소스]

Serial 통신

```
void setup() {
  Serial.begin(9600);    //시리얼 통신 9600 통신속도로 시작
}
void loop() {

  if (Serial.available() > 0) { //데이터가 수신되는지 확인
    char ch = Serial.read(); //1byte 읽음
    Serial.println(ch); //1byte 읽은거 출력
  }
}
```

SoftwareSerial 통신

```
#include <SoftwareSerial.h>

const int rx = 2; //Bluetooth TX 핀
const int rx = 3; //Bluetooth RX 핀
SoftwareSerial mySerial (rx, tx) : 소프트시리얼 객체선언(rx(수신), tx(전송))

void setup() {
  mySerial.begin(9600);    //시리얼 통신 9600 통신속도로 시작
}
void loop() {

  if (mySerial.available() > 0) { //데이터가 수신되는지 확인
    char ch =mySerial.read(); //1byte 읽음
    mySerial.println(ch); //1byte 읽은거 출력
  }
}
```

기본 동작은 Serial.read() 함수로 1byte씩 읽어와 다시 시리얼모니터로 1byte을 출력하는 방법은 예전에 post로 설명을 했었습니다. 복습차원으로 다시 한번 살펴 봐 주세요. 그리고 Bluetooth의 SoftwareSerial 통신도 Serial에 mySerial로 작명한 Serial 객체변수명으로 접근하는데 기본 함수명과 동작은 동일합니다. Serial 통신을 할 수 있으면 SoftwareSerial로 Bluetooth 통신도 할 수 있겠죠.

복습이 끝났다면 본격적으로 사용할 String 대해서 살펴보도록 하죠.

2. String 함수

> Stiing 레퍼런스 :
> https://www.arduino.cc/reference/ko/language/variables/data-types/stringobject/
> https://www.arduino.cc/reference/ko/language/functions/communication/serial/
> https://www.arduino.cc/reference/ko/language/functions/communication/stream/

아두이노 공식 홈페이지에 가면 String 함수들이 나열 되어 있습니다. 링크된 곳에 가셔서 한번씩 테스트 해보셨으면 합니다.
오늘 post에 사용 할 몇개 String 함수에 대해서 살펴보도록 하겠습니다.

stream : https://www.arduino.cc/reference/ko/language/functions/communication/stream/

위 링크쪽을 가시면 Serial 통신 시 관련 함수들이 있습니다. 문자열을 Serial 통신을 통해 읽을때 readString(), readStringUntil() 두개의 함수가 있습니다. 둘 다 사용해도 되지만 readStringUntil()함수를 사용합니다.

```
if(Serial.available()){
   String inString = Serial.readStringUntil('\n');
}
```

문자열 변수 선언

String inString : 문자열 객체변수 선언

문자열 Serial 통신 읽기

Serial.readStringUntil('\n') : '\n' 문자를 만날때까지 문자열을 읽는 함수

문자열 공백 제거

String.trim() : 문자열변수안에 공백을 제거

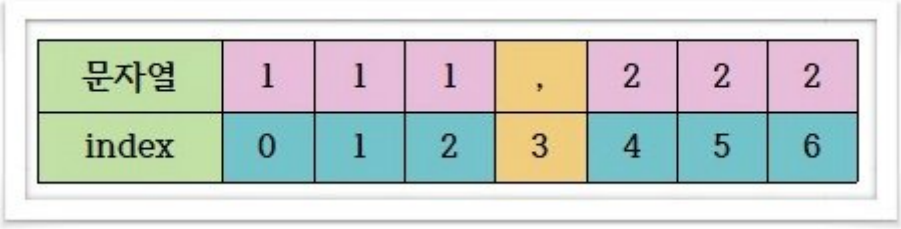
문자열 분리에 사용할 함수

- **String.indexOf('찾을문자')** : 문자열에 '찾을문자'가 있는 위치(index) 값을 반환
- **String.indexOf('찾을문자',시작위치)** : 문자열의 시작위치에서 시작하고 '찾을문자'가 있는 위치(index) 값을 반환
- **String.substring(시작위치, 종료위치)** : 시작~종료위치까지의 문자열을 반환한다. 전체 문자열에서 부분 문자열 추출.

3. 문자열 분리

```
String inString = "111,222";
int index1 = inString.indexOf(',');
Serial.println(index1);
```

결과 : **3**



문자열	1	1	1	,	2	2	2
index	0	1	2	3	4	5	6

이렇게 하면 inString에 저장된 "111,222" 문자열에서 ','가 있는 위치값 '3'을 반환합니다. 위치는 0부터 시작하니깐 네번째인 3이 반환됩니다. 혼동하지 마세요. 여기서 ','를 기준으로 "111"과 "222"의 문자열을 분리해 낼려면 substring()함수를 사용하게 됩니다.

```
String inString = "111,222";
int index1 = inString.indexOf(',');
int index2 = inString.length();
String inString1 = inString.substring(0, index1);
String inString2 = inString.substring(index1+1,index2);
Serial.println(inString1);
Serial.println(inString2);
```

결과 :
111
222

여기서, 문자열이 숫자면 숫자형으로 변환 시켜야 합니다. 그냥 사용 한다면 문자열이지 숫자형이 아닙니다. 문자열을 숫자형으로 변환 하기 위해서는 다음과 같은 함수를 사용해야 합니다.

String.toInt() : 문자열을 정수형으로 변환

```
String inString = "111,222";
int index1 = inString.indexOf(',');
int index2 = inString.length();
int inString1 = inString.substring(0, index1).toInt();
int inString2 = inString.substring(index1+1,index2).toInt();
Serial.print(inString1);
Serial.print('+');
Serial.print(inString2);
Serial.print('=');
Serial.println(inString2+inString2);
```

결과 : **111+222=333**

int inString1 = inString.substring(0, index1).toInt();
자료형 변수 = 전체문자열.부분문자열추출.정수형변환;

inString의 문자열 "111,222" 값에서 substing(0,3)함수로 "111"문자열이 추출되고 이 값을 toInt()로 정수형으로 변환 합니다.

substring(from, to)함수의 혼동 주의를 하세요. substring(인자1, 인자2)에서 두 인자는 위치를 가리키지만 정확히 말하면 인자1과 인자2의 의미는 좀 다른 것 같습니다. 문자열이 0부터 시작한다고 했죠. 그러면, 시작 위치가 0부터 하셔야 첫번째 문자 '1'의 값을 가리키게 됩니다. 그런데 뒤에 콤마(,) 문자의 위치를 가리키면 위치값이 3이 됩니다. substring(0,3)의 문자열을 추출하게 됩니다. 이럴 때 3이니깐 4번째 콤마(,)까지 추출되어 나오는 거 아냐 하실 수 있는데 그 전 문자열 index(2)까지의 부분 문자열을 추출합니다. 간혹, 콤마(,)의 위치가 3이니깐 필요한 문자열이 "111"로 index(0~2)의 문자 3개가 필요하니깐 콤마(,) index 값에서 -1을 해서 부분문자열을 추출하려고 잘못된 코딩을 하실 수 있습니다. 처음에는 이런 문제로 시행착오를 거치실 수 있지만 금방 문제의 원인을 찾아서 수정하실 수 있을 꺼에요. 하지만, 실제 실험을 안하고 글을 읽고 상상코딩을 하신다면 이부분을 고려하시고 상상코딩을 하셔야 합니다.

substring(인자1, 인자2)함수가 좀 그렇더군요. 명확하게 인자2를 문자열의 정확한 추출위치 인자 index로 해 놓았다면 좋았을 것을 왜! 이런식으로 표현했는지 아쉬움이 남는 함수입니다. 마지막 널문자를 나타내기 위한 의도인지 아니면 뒤에 인자2는 몇번째 인지를 나타내는 숫자인지 참 모호한 인자인 것 같아요. 나중에 문자열을 좀 더 많은 데이터를 쪼갤때 혼동하실 수 있으니깐 잘 기억해 두세요.

예) **String inString = "1,2,3,4,5";**

콤마(,)인덱스는

int index1 = inString.indexOf(',');
int index2 = inString.indexOf(',',index1+1);
int index3 = inString.indexOf(',',index2+1);
...

이렇게 각 콤마(,) 문자의 위치를 얻고 이것을 분리해 낼때는
int inString1 = inString.substring(0, index1).toInt();
int inString2 = inString.substring(index1+1, index2).toInt();
int inString3 = inString.substring(index2+1, index3).toInt();
...
이렇게 해야하는데 인덱스 위치를 혼동해서 잘못 위치를 지정하게 될 수 있으니 주의하세요. 혼동을 피하기 위해서는 인자1은 배열 index 위치라고 생각하고 인자2은 문자열의 몇번째 위치라고 생각하시면 될 듯 싶네요. 이게 더 혼동이 될려나 모르겠군요.

indexOf(0,3) 이면 배열[0]에서 문자열 3번째 배열[2]까지의 부분 문자열을 추출한다. 이렇게 전 정리했네요. 아무튼 요상한 함수입니다. 이게 원래는 배열[0]~배열[3]까지 부분 문자열을 추출하는데 마지막 배열[3]은 문자열 끝을 나타내는 널문자가 들어가기 때문에 실제로 문자열 배열[0]~배열[2]까지의 글자가 출력되는지 모르겠지만요. 좀 혼동되는 함수이니깐 주의해서 코딩해 주세요.

이야기가 삼천포로 빠졌지만 계속 이야기를 이여 가겠습니다.

자이로센서와 같은 센서를 사용할 경우 값이 실수형으로 표현될 경우 문자열을 실수형 값으로 전송하게 된다면 다음과 같이 변경하시면 됩니다.

String.toFloat() : 문자열을 실수형으로 변환

예) **String inString = "111.11,-222.22";**

이와 같이 문자열이 주어졌다면

float inString1 = inString.substring(0, index1).toFloat();

결과 : **111.11**

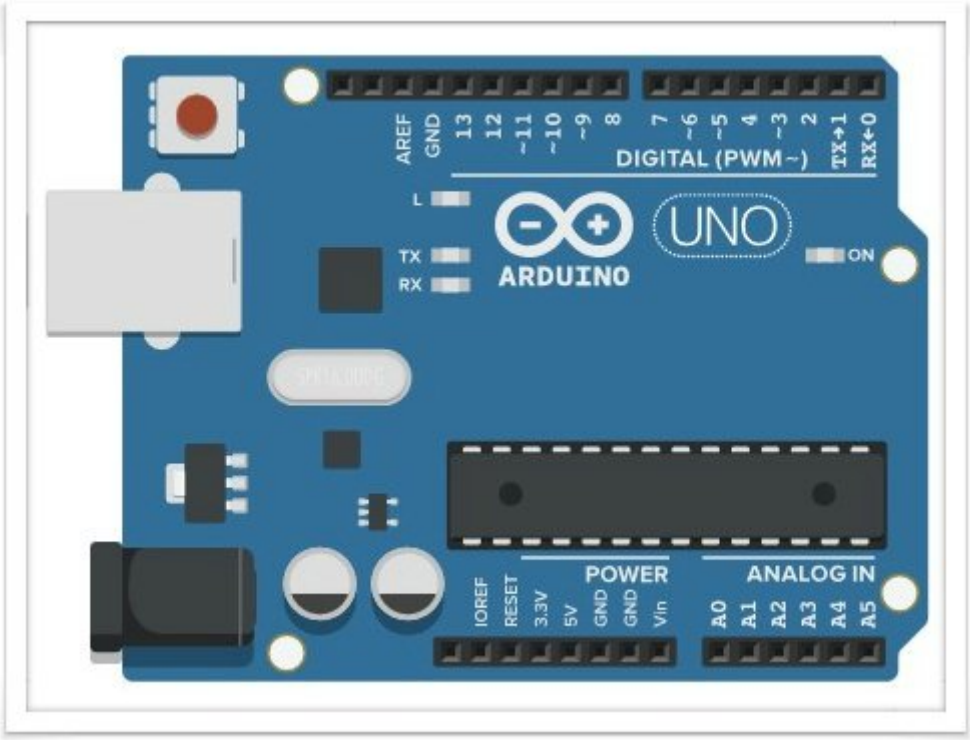
이렇게 문자열을 실수형으로 변환하고 그 값을 실수자료형 변수에 저장하게 됩니다.

4. Serial 통신으로 테스트

간단히, 시리얼 모니터로 두개의 실수형 데이터 x,y 값을 Serial 통신을 통해 문자열로 한번에 전송한다고 가정하고 이 두값을 구분하는 문자 콤마(,)로 구분자를 만들어 보낸다고 설정 했습니다. Serial 통신을 통해 문자열을 읽고 그 문자열을 다시 x,y값으로 분리해 내서 각 데이터를 실수형 변수에 저장하고 정상적으로 분리가 되었는지 시리얼모니터로 출력하는 소스입니다.

분리된 문자열이 실수형으로 정확히 변환이 되었는지 확인하기 위해서 간단히 두 실수값을 더한 값을 시리얼 모니터로 출력하여 확인합니다.

[소스]



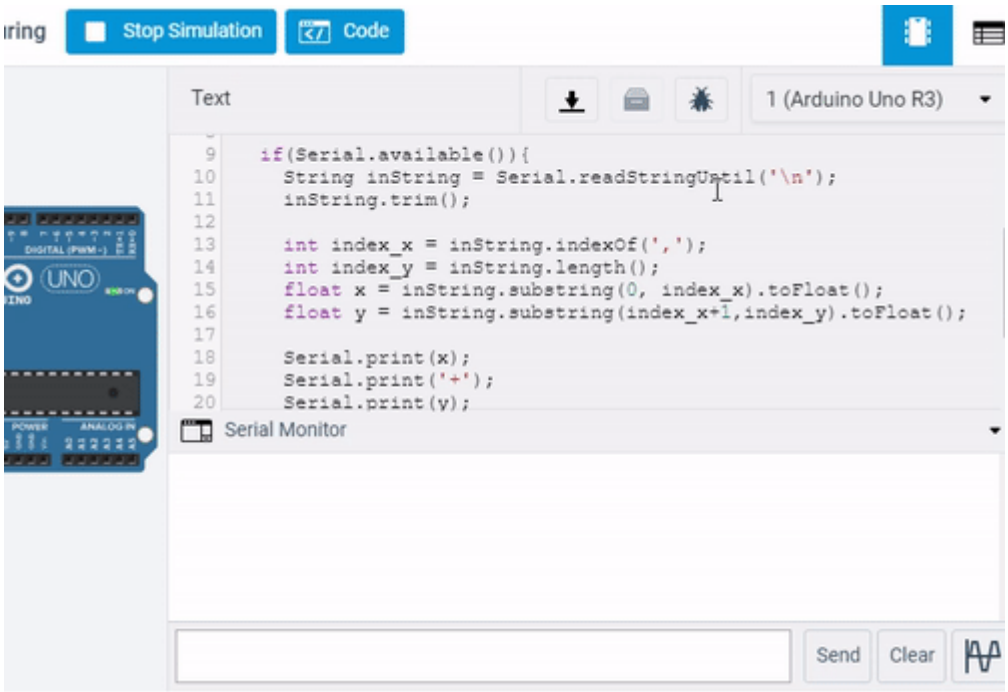
```
void setup()
{
  Serial.begin(9600);
}

void loop()
{
  if(Serial.available()){
    String inString = Serial.readStringUntil('\n');

    int index_x = inString.indexOf(',');
    int index_y = inString.length();
    float x = inString.substring(0, index_x).toFloat();
    float y = inString.substring(index_x+1,index_y).toFloat();

    Serial.print(x);
    Serial.print('+');
    Serial.print(y);
    Serial.print('=');
    Serial.println(x+y);
  }
}
```

[결과]



마무리

오늘은 간단히 Serial 통신으로 여러개의 데이터를 하나의 문자열로 보내지면 그 문자열을 다시 여러개의 데이터로 분리해내는 과정을 실험 하였네요.

이 원리는 다양한 데이터를 측정할 때 그 값을 한번에 전송하는데 사용하면 좋습니다. 예를 들어, MPU6050 자이로센서의 경우는 가속도 x,y,z 온도, 각속도 x,y,z 값으로 총 7개의 데이터를 측정하게 됩니다. 이것을 하나씩 개별적으로 보낸다면 불편하겠죠. 이 데이터 7개를 한번에 보내고 형식에 맞춰 분리하여 원하는 동작 제어를 한다면 편하게 제어 할 수 있게 됩니다.

하나씩 전송하게 되면은 데이터를 읽을 때 x값인지 y값인지 구분해서 읽는 코딩은 좀 복잡해집니다. 하지만 이렇게 문자열로 보내고 문자열로 읽고 해당 x, y값의 위치 문자열에서 분리해 내서 읽게 되면 좀 더 편하게 코딩을 할 수 있습니다. 사실 문자열 Serial 통신을 사용할 경우는 극히 드물지만 참고로 이런게 있다는 것만 알아만 두세요

#jjangjjangman #kr-dev #kr #blog
3년 전 in #kr-arduino by **codingman (64)**
댓글 달기 2