

HW Programming

wk10 :

Arduino Coding II.

Coding using IDE

Basic HW coding using Arduino and EV3 (RP3)

HIT, INJE University

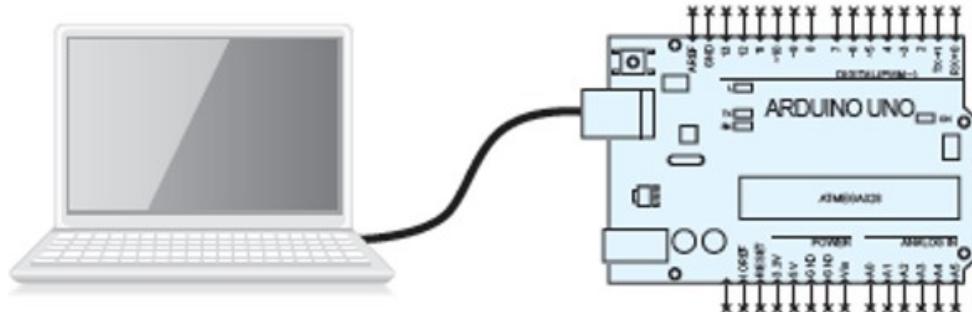
1st semester, 2017

Email : yish@inje.ac.kr

Weekly schedule of HP– 1st semester, 2017



- **wk01 :**
- **wk02 :**
- **wk03 :**
- **wk04 :**
- **wk05 :**
- **wk06 : Introduction to class and enrollment in cyber class, and installing SW**
- **wk07 : Basic HW : Arduino I. – circuit**
- **wk08 : Mid-term exam. → Practice of various circuits**
- **wk09 : Basic HW : Arduino II. – coding by Arduino IDE**
- **wk10 : Arduino Coding II. – Serial monitor**
- **wk11 :**
- **wk12 :**
- **wk13 :**
- **wk14 :**
- **wk15 : Final exam.**





Arduino SW

fritzing.org Fritzing Fritzing

fritzing

electronics
made easy

Projects Parts Download Learning Services Contribute FORUM FAB

fritzing APP

Download the free Fritzing App and start building immediately!

Fritzing is an open-source hardware initiative that makes electronics accessible as a creative material for anyone. We offer a software tool, a community website and services in the spirit of Processing and Arduino, fostering a creative ecosystem that allows users to document their prototypes, share them with others, teach electronics in a classroom, and layout and manufacture professional pcbs.

Download and Start

Download our latest version 0.9.3b released on June 2, 2016 and start right away.

Produce your own board

With Fritzing Fab you can easily and inexpensively turn your circuit into a real, custom-made PCB. Try it out now!

Participate

Fritzing can only act as a creative platform if many

<http://fritzing.org/home/>



Arduino

Starter Kit



Arduino Starter Kit





Arduino Starter Kit : contents

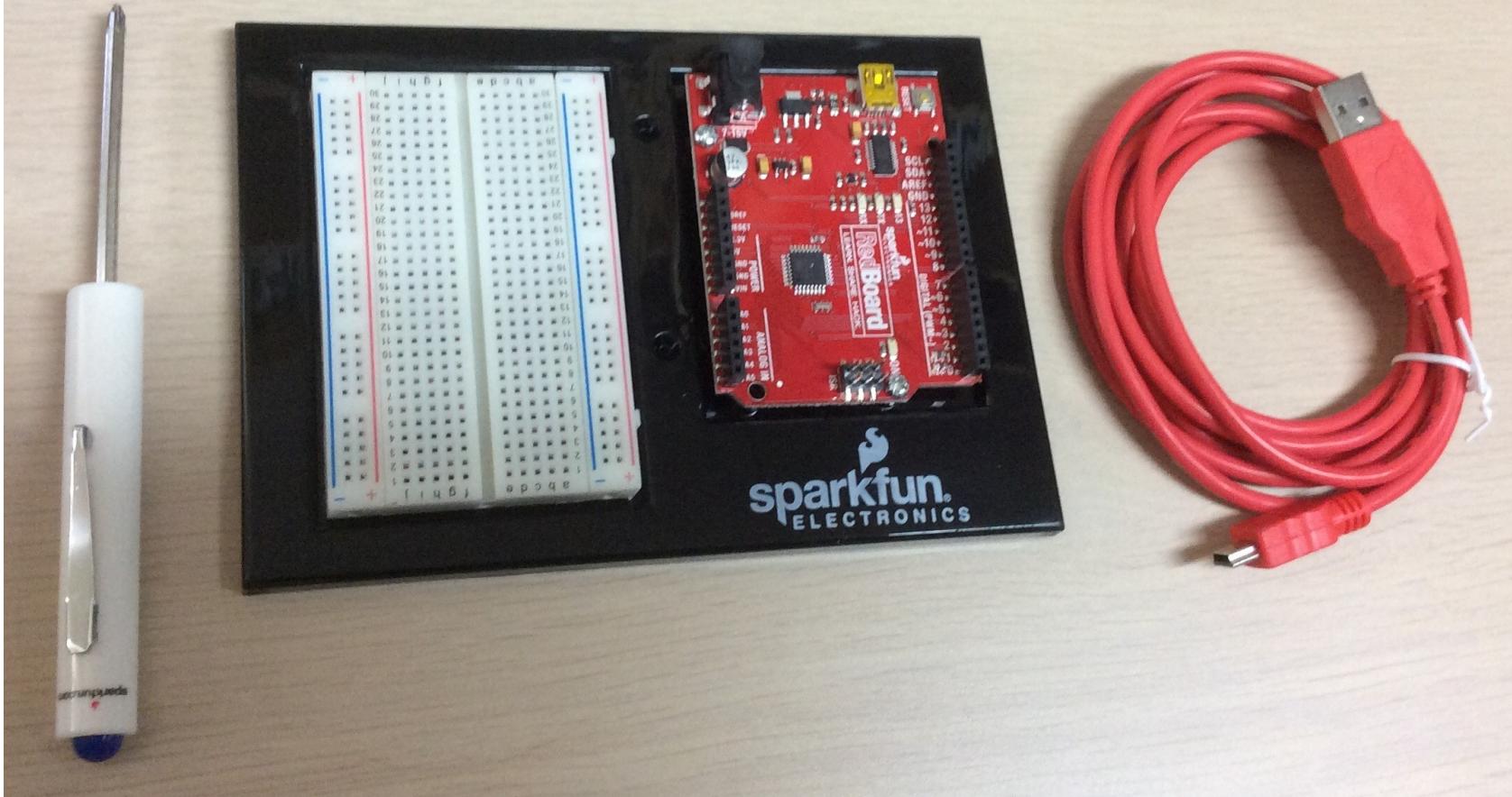
Arduino set

- **SparkFun 레드보드(아두이노 uno 호환)**
- 아두이노, 레드보드 고정판
- 한글판 가이드 북
- **400옴 브레드보드**
- 부품케이스
- **16x2 White on Black LCD**
- **74HC595** 쉬프트레지스터
- **2N2222** 트랜지스터
- **1N4148** 다이오드 (2)
- **기어달린 DC 모터**
- 서보모터
- **SPDT 5V 릴레이**
- **TMP36 온도센서**

- **플렉스 센서(Flex sensor)**
- **소프트 포텐시오미터(Softpot)**
- **SparkFun USB Cable**
- 점퍼케이블 (65)
- **Photocell**
- **RGB LED**
- **적색, 황색, 청색, 초록색 LED (5)**
- **10K 포텐시오미터 (가변저항, Trimpot)?**
- 피에조 부저
- **풀사이클 푸시버튼 (4)**
- **330, 10K 저항 (20)**

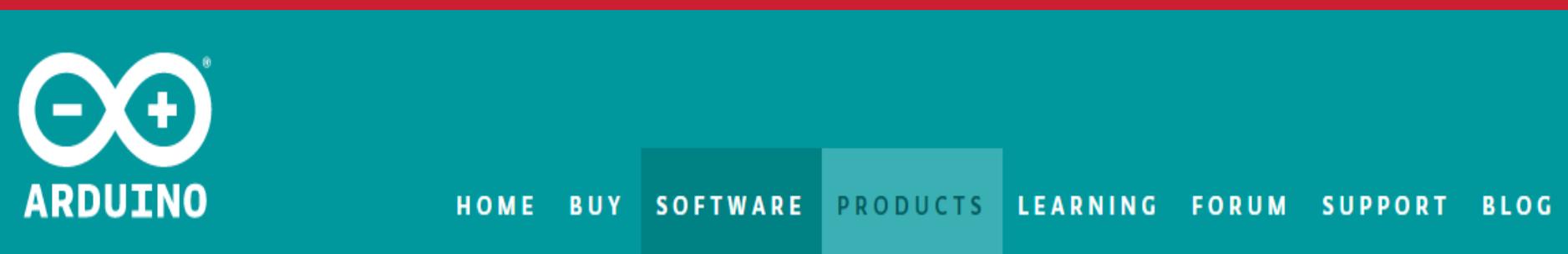


Arduino Starter Kit (Board setting)





1. Arduino SW: IDE



The screenshot shows the top navigation bar of the Arduino website. On the left is the Arduino logo. To its right are several menu items: HOME, BUY, SOFTWARE (which is highlighted in a darker teal color), PRODUCTS, LEARNING, FORUM, SUPPORT, and BLOG. The background of the header is a teal gradient.

<https://www.arduino.cc/>



1.5 Arduino IDE

sketch_may01a | 아두이노 1.8.2

파일 편집 스케치 툴 도움말

- 새 파일 Ctrl+N
- 열기... Ctrl+O
- 최근 파일 열기 >
- 스케치북 >
- 예제 >
- 닫기 Ctrl+W
- 저장 Ctrl+S
- 다른 이름으로 저장... Ctrl+Shift+S
- 페이지 설정 Ctrl+Shift+P
- 인쇄 Ctrl+P
- 환경설정 Ctrl+Comma
- 종료 Ctrl+Q

환경설정

설정 네트워크

스케치북 위치: C:\Users\yish-HC\Documents\Arduino

에디터 언어: 시스템 기본설정 (마두이노를 재시작해야 함)

에디터 글꼴 크기: 14

Interface scale: 자동 100 % (마두이노를 재시작해야 함)

다음 동작중 자세한 출력 보기: 컴파일 업로드

컴파일러 경고: None

줄 번호 표시

코드 풀딩 사용하기

업로드 후 코드 확인하기

외부 에디터 사용

Aggressively cache compiled core

시작시 업데이트 확인

스케치 파일을 저장할 때 새로운 확장자(.pde -> .ino)로 업데이트

검증 또는 업로드 할 때 저장하기

추가적인 보드 매니저 URLs

추가적인 환경 설정은 파일에서 직접 편집할 수 있습니다
C:\Users\yish-HC\AppData\Local\Arduino15\preferences.txt
(마두이노가 실행되지 않는 경우에만 수정 가능)

확인 취소

Arduino coding II.



1. Arduino SW : IDE

2. Blink a LED

3. Serial monitor



3. Serial monitor & plotter



3. Serial monitor & plotter

COM3 (Arduino/Genuino Uno)

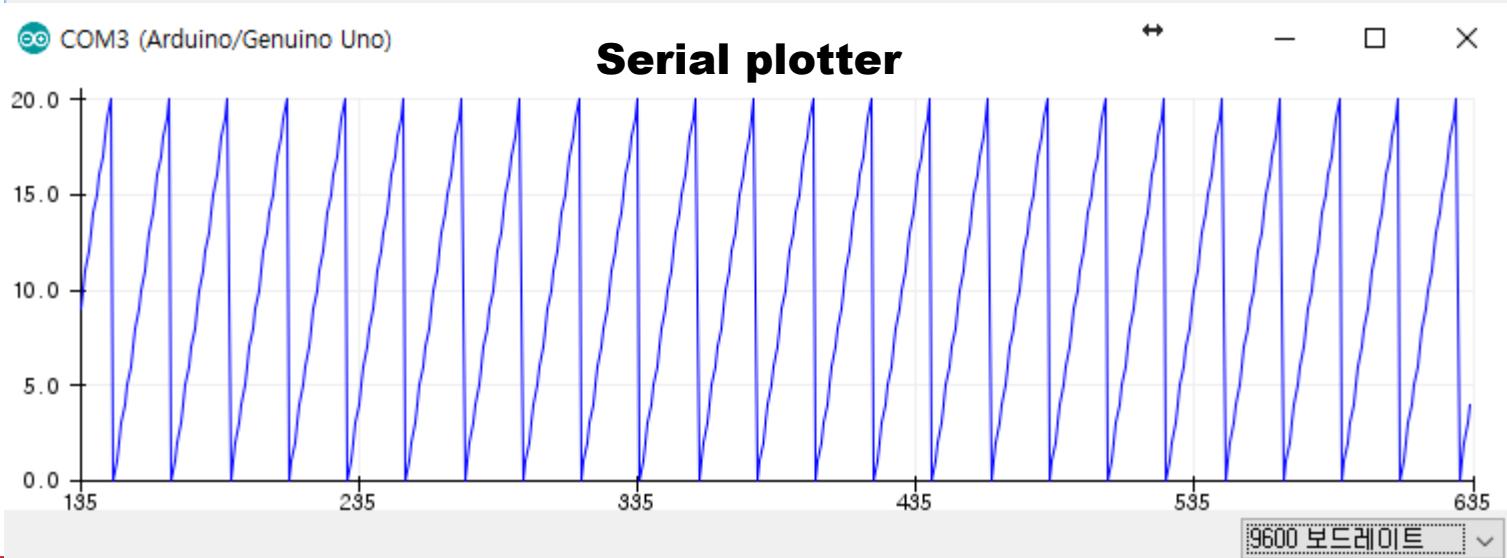
14 sec
15 sec
16 sec
17 sec
18 sec
19 sec
20 sec
0 sec
1 sec
2 sec
3 sec
4 sec
5 sec
6 sec

자동 스크롤

line ending 없음

9600 보드레이트

Serial moniter





3. Serial monitor & plotter

- 3.1 Arduino에서 컴퓨터로 데이터 전송하기
- 3.2 변수 유형별로 컴퓨터에 전송하기
- 3.3 Arduino에서シリ얼 통신을 이용하여
데이터 수신하기

3. 시리얼 통신 (Serial communication)

시리얼 통신

UART (Universal Asynchronous Receiver/Transmitter)

RS-232

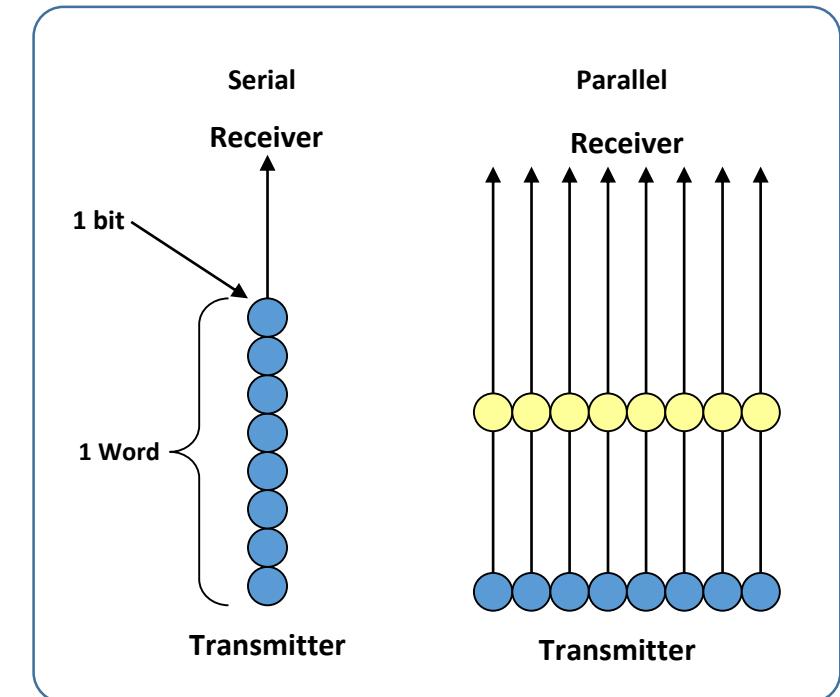
RS-422

RS-485

Arduino에서는 다음과 같은 목적으로 사용

Debugging : 프로그램의 오류를 수정하는 작업

데이터 통신 : Arduino와 컴퓨터 혹은 다른 장치와의
통신



3.1.4 Arduino에서 컴퓨터로 데이터 전송하기

EX 2.1

Arduino에서 컴퓨터로 변수와 문자열 전송하기 (3/3)

Sketch 구성 1. 시리얼통신을 시작한다. ‘Serial.begin()’ 명령어로 할 수 있다.

2. 변수를 전송하기 위해 ‘Serial.print()’ 명령어를 사용하고,
문자열 전송 후 줄 바꿈을 하기 위해서 ‘Serial.println()’ 명령어를 사용한다.
3. 루프를 1초마다 실행하기 위해서 ‘delay()’명령어로 시간지연을 시켜준다.

실습 결과 IDE의 시리얼 모니터를 실행시켜 Arduino에
서 전송되는 메시지를 확인할 수 있다.

응용 문제 1. 2초, 5초 단위로 시간을 변경해 보자.

2. 자신만의 메시지를 출력해보자.

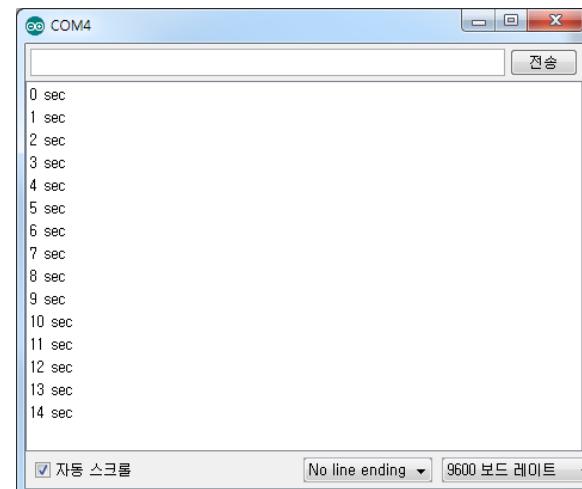
3. [DIY1]

delay를 0.2초로 설정후

5초 마다 number를 초기화하여

시리얼플로터로 톱니파를 발생.

시간은 ms로 계산해서 출력

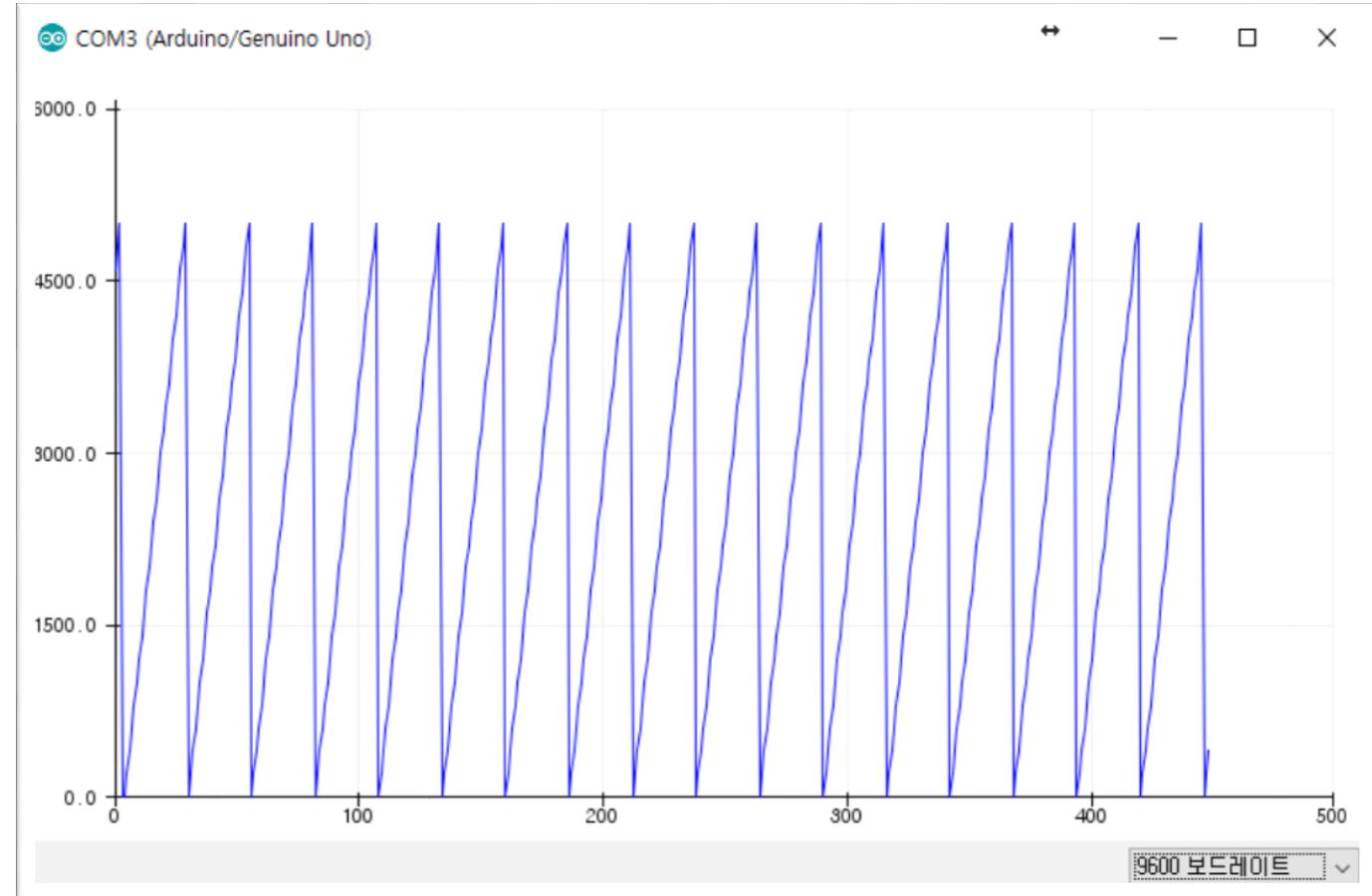




DIY2. sawtooth signal

COM3 (Arduino/Genuino Uno)

4600 msec
4800 msec
5000 msec
0 msec
200 msec
400 msec
600 msec
800 msec
1000 msec
1200 msec
1400 msec
1600 msec
1800 msec
2000 msec
2200 msec
2400 msec
2600 msec
2800 msec
3000 msec
3200 msec



HPnn_sawtooth.png



DIY2. sawtooth signal : Code-1

hp00_diy2_sawtooth

```
1 /*  
2 예제 2.1 : hp00_sawtooth  
3 Arduino에서 컴퓨터로 변수와 문자열 전송하기  
4 */  
5  
6 int number = 0;           // -32768~32767 범위의 변수 number 설정, 초기값은 0  
7  
8 void setup() {  
9   Serial.begin(9600);     // 9600bps로 시리얼 통신 설정  
10 }  
11  
12 void loop() {  
13   Serial.print(number*200); // number 변수값 출력  
14   Serial.println(" msec"); // " msec"를 출력 후 줄 바꿈  
15   delay(200);            // 0.2초동안 지연시킨다.  
16   number++;              // number 변수값을 하나 증가시킨다.  
17  
18   if (number > 25) {  
19     number = 0;  
20   }  
21 }
```



DIY2. sawtooth signal : Code-2

hp00_diy2_sawtooth2

```
1 /*
2 예제 2.1 : hp00_sawtooth
3 Arduino에서 컴퓨터로 변수와 문자열 전송하기
4 */
5
6 int number = 0;           // -32768~32767 범위의 변수 number 설정, 초기값은 0
7
8 void setup() {
9     Serial.begin(9600);    // 9600bps로 시리얼 통신 설정
10 }
11
12 void loop() {
13     Serial.print(number); // number 변수값 출력
14     Serial.println(" msec"); // " msec"를 출력 후 줄 바꿈
15     delay(200);           // 0.2초동안 지연시킨다.
16     //number++;           // number 변수값을 하나 증가시킨다.
17
18     if (number < 5000) {
19         number += 200;
20     }else {
21         number = 0;
22     }
23 }
```



DIY2. sawtooth signal : Code-3

```
hp00_diy2_sawtooth3
1 /*
2 예제 2.1 : hp00_sawtooth
3 Arduino에서 컴퓨터로 변수와 문자열 전송하기
4 */
5
6 int number = 0;           // -32768~32767 범위의 변수 number 설정, 초기값은 0
7
8 void setup() {
9     Serial.begin(9600);    // 9600bps로 시리얼 통신 설정
10 }
11
12 void loop() {
13     Serial.print(number); // number 변수값 출력
14     Serial.println(" msec"); // " msec"를 출력 후 줄 바꿈
15     delay(200);           // 0.2초동안 지연시킨다.
16     number += 200;         // number 변수값을 200 증가시킨다.
17
18     if (number > 5000) {   // 5초(5000ms) 경과하면 number 초기화
19         number = 0;
20     }
21 }
```



3.2 Serial monitor & plotter

3.2

변수 유형별로 컴퓨터에 전송하기

```
*** Hello Arduino ***
```

```
*** char Value ***
```

```
Binary:1000001
```

```
Decimal:65
```

```
Hexadecimal:41
```

```
ASCII:A
```

```
*** int Value ***
```

```
int Value:65
```

```
char(intValue):A
```

```
*** float Value ***
```

```
float Value:65.00
```



3.2.1 변수 유형별로 컴퓨터에 전송하기

- ✓ 사용 목적에 따라 다양한 변수 유형 중 선택하여 사용
- ✓ C 언어와 유사함

표 2.1 Arduino 변수 유형

변수 유형	바이트	변수 범위	용도
boolean	1	true(1) 혹은 false(0)	참(1) 아니면 거짓(0) 값을 나타냄.
char	1	-128~127 혹은 아스키 코드 값과 매칭되는 문자	부호가 있는 1바이트 숫자를 나타냄
unsigned char	1	0~255	0~255의 숫자를 나타낼 때 사용함
byte	1	0~255	unsigned char과 동일
int	2	-32,768~32,767	부호가 있는 2바이트 숫자를 나타냄
short	2	-32,768~32,767	int와 동일
unsigned int	2	0~65,535	부호가 없는 2바이트 숫자를 나타냄
word	2	0~65,535	unsigned int와 동일
long	4	-2,147,483,648 ~2,147,483,647	부호가 있는 4바이트 숫자를 나타냄
unsigned long	4	0~4,294,967,295	부호가 없는 4바이트 숫자를 나타냄
float	4	$-3.4028235 \times 10^{38}$ $\sim 3.4028235 \times 10^{38}$	소수점 있는 숫자를 나타냄
double ¹⁾	4 or 8	$-3.4028235 \times 10^{38}$ $\sim 3.4028235 \times 10^{38}$ Or $-1.79 \times 10^{308} \sim 1.79 \times 10^{308}$	소수점 있는 숫자를 나타냄. 4바이트의 경우 float과 동일 8바이트의 경우 변수 범위가 커짐
String	-	문자열에 따라 다름	문자열을 나타냄.



3.2.2 변수 유형별로 컴퓨터에 전송하기

✓ 실제 전송은 아스키코드 (ASCII Code)를 전송함

ASCII CODE TABLE

10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자
0	0x00	NULL	22	0x16	STN	44	0x2C	.	66	0x42	B	88	0x58	X	110	0x6E	n
1	0x01	SOH	23	0x17	ETB	45	0x2D	-	67	0x43	C	89	0x59	Y	111	0x6F	o
2	0x02	STX	24	0x18	CAN	46	0x2E	.	68	0x44	D	90	0x5A	Z	112	0x70	p
3	0x03	ETX	25	0x19	EM	47	0x2F	/	69	0x45	E	91	0x5B	[113	0x71	q
4	0x04	EOT	26	0x1A	SUB	48	0x30	0	70	0x46	F	92	0x5C	₩	114	0x72	r
5	0x05	ENQ	27	0x1B	ESC	49	0x31	1	71	0x47	G	93	0x5D]	115	0x73	s
6	0x06	ACK	28	0x1C	FS	50	0x32	2	72	0x48	H	94	0x5E	^	116	0x74	t
7	0x07	BEL	29	0x1D	GS	51	0x33	3	73	0x49	I	95	0x5F	_	117	0x75	u
8	0x08	BS	30	0x1E	RS	52	0x34	4	74	0x4A	J	96	0x60	.	118	0x76	v
9	0x09	HT	31	0x1F	US	53	0x35	5	75	0x4B	K	97	0x61	a	119	0x77	w
10	0x0A	LF	32	0x20	SP	54	0x36	6	76	0x4C	L	98	0x62	b	120	0x78	x
11	0x0B	VT	33	0x21	!	55	0x37	7	77	0x4D	M	99	0x63	c	121	0x79	y
12	0x0C	FF	34	0x22	"	56	0x38	8	78	0x4E	N	100	0x64	d	1222	0x7A	z
13	0x0D	CR	35	0x23	#	57	0x39	9	79	0x4F	O	101	0x65	e	123	0x7B	{
14	0x0E	SO	36	0x24	\$	58	0x3A	:	80	0x50	P	102	0x66	f	124	0x7C	
15	0x0F	SI	37	0x25	%	59	0x3B	;	81	0x51	Q	103	0x67	g	125	0x7D	}
16	0x10	DEL	38	0x26	&	60	0x3C	<	82	0x52	R	104	0x68	h	126	0x7E	~
17	0x11	DC1	39	0x27	'	61	0x3D	=	83	0x53	S	105	0x69	i	127	0x7F	DEL
18	0x12	DC2	40	0x28	(62	0x3E)	84	0x54	T	106	0x6A	j			
19	0x13	DC3	41	0x29)	63	0x3F	?	85	0x55	U	107	0x6B	k			
20	0x14	DC4	42	0x2A	*	64	0x40	@	86	0x56	V	108	0x6C	l			
21	0x15	NAK	43	0x2B	+	65	0x41	A	87	0x57	W	109	0x6D	m			

그림 2.2 ASCII 코드표

3.2.3 변수 유형별로 컴퓨터에 전송하기

EX 2.2

변수 유형별 Arduino에서 컴퓨터로 전송하기 (1/4)

- 실습목표
1. Arduino에서 컴퓨터로 데이터를 전송할 때 변수 유형별로 출력한다.
 2. char로 선언된 변수, int로 선언된 변수, float로 선언된 변수를 ‘Serial.print’ 명령어를 이용하여 PC로 전송하자.
 3. ‘Serial.print’ 명령어의 출력 옵션을 변경하여 전송해 보자.
 4. ‘Serial.write’ 명령어로 문자열을 출력해 보자.
 5. 각 변수 유형별 출력되는 차이를 비교해 보자.

Hardware

Arduino와 PC를 USB 케이블로 연결한다.

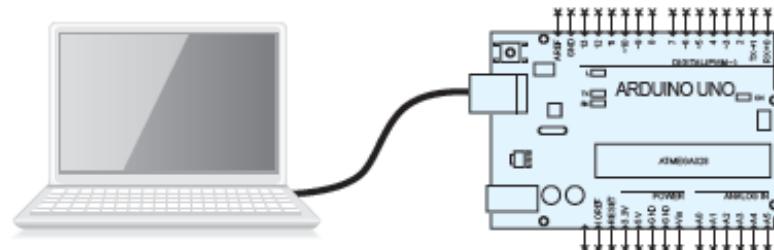


그림 2.1 Arduino와 PC와의 연결



3.2.4 변수 유형별로 컴퓨터에 전송하기

EX 2.2

변수 유형별 Arduino에서 컴퓨터로 전송하기 (2/4)

Commands

- `Serial.write(char 변수);`

char변수에 해당하는 ASCII 코드값의 문자를 출력한다.

- `Serial.print(변수, BIN);`

변수를 2진수(Binary)로 표시한다.

- `Serial.print(변수, DEC);`

변수를 10진수(Binary)로 표시한다.

- `Serial.print(변수, HEX);`

정해진 변수를 16진수(Hexadecimal)로 표시한다.



3.2.5 변수 유형별로 컴퓨터에 전송하기

EX 2.2

변수 유형별 Arduino에서 컴퓨터로 전송하기 (3/4)

- Sketch 구성
1. ‘65’란 숫자를 char형, int형, float형 변수에 각각 저장한다.
 2. ‘Binary:’, ‘Decimal:’, ‘Hexadecimal:’, ‘ASCII:’ 등 네 가지 문자열을 저장하여 호출하여 사용한다.
 3. 변수에 저장된 숫자를 2진수형, 10진수형, 16진수형, ASCII 코드형 등 Serial.print 명령어의 옵션을 변경하여 전송한다.
 4. 옵션이 변경될 때마다 문자열을 호출하여 함께 출력한다.
 5. loop가 반복될 때마다 숫자를 1씩 증가시킨다. float형은 0.1씩 증가시킨다.

3.2.6 변수 유형별로 컴퓨터에 전송하기

EX 2.2

변수 유형별 Arduino에서 컴퓨터로 전송하기 (3/4)

- Sketch 구성
1. ‘65’란 숫자를 char형, int형, float형 변수에 각각 저장한다.
 2. ‘Binary:’, ‘Decimal:’, ‘Hexadecimal:’, ‘ASCII:’ 등 네 가지 문자열을 저장하여 호출하여 사용한다.
 3. 변수에 저장된 숫자를 2진수형, 10진수형, 16진수형, ASCII 코드형 등 Serial.print 명령어의 옵션을 변경하여 전송한다.
 4. 옵션이 변경될 때마다 문자열을 호출하여 함께 출력한다.
 5. loop가 반복될 때마다 숫자를 1씩 증가시킨다. float형은 0.1씩 증가시킨다.



3.2.7 변수 유형별로 컴퓨터에 전송하기

EX 2.2

변수 유형별 Arduino에서 컴퓨터로 전송하기 (4/4)

실습 결과 IDE의 시리얼 모니터를 실행시켜

Arduino에서 전송되는 메시지를
확인할 수 있다.

10초 간격으로 증가된 값에 대하여
출력하게 된다. 이때 2진수, 10 진수,
16진수로 표시되고, 증가된 숫자에
해당하는 아스키코드의 문자가 전송
된다. ‘float’형 변수는 소수점이 함
께 표시된다.

```
*** char Value ***
Binary:10000001
Decimal:65
Hexadecimal:41
ASCII:A
```

```
*** int Value ***
*** float Value ***
float Value:65.00
```

Quiz 1. `Serial.write(floatValue);`

위의 명령이 오류가 나는 이유는?



3.2.8 DIY-1

```
*** char Value ***
```

```
Binary:1000001
```

```
Decimal:65
```

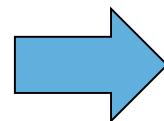
```
Hexadecimal:41
```

```
ASCII:A
```

```
*** int Value ***
```

```
*** float Value ***
```

```
float Value:65.00
```



```
*** Hello Arduino by HPnn***
```

```
*** char Value ***
```

```
Binary:1000001
```

```
Decimal:65
```

```
Hexadecimal:41
```

```
ASCII:A
```

```
*** int Value ***
```

```
int Value:65
```

```
char(intValue):A
```

```
*** float Value ***
```

```
float Value:65.00
```

HPnn_hello.png



DIY2. Escape from loop()

응용 문제 [DIY2] 0~15까지 10진수를 2진수와 16진수로 출력하는 스케치를 작성해보자

```
COM3 (Arduino/Genuino Uno)

Number = 0, Binary:0, Hexadecimal:0
Number = 1, Binary:1, Hexadecimal:1
Number = 2, Binary:10, Hexadecimal:2
Number = 3, Binary:11, Hexadecimal:3
Number = 4, Binary:100, Hexadecimal:4
Number = 5, Binary:101, Hexadecimal:5
Number = 6, Binary:110, Hexadecimal:6
Number = 7, Binary:111, Hexadecimal:7
Number = 8, Binary:1000, Hexadecimal:8
Number = 9, Binary:1001, Hexadecimal:9
Number = 10, Binary:1010, Hexadecimal:A
Number = 11, Binary:1011, Hexadecimal:B
Number = 12, Binary:1100, Hexadecimal:C
Number = 13, Binary:1101, Hexadecimal:D
Number = 14, Binary:1110, Hexadecimal:E
Number = 15, Binary:1111, Hexadecimal:F
Mission completed!
```

[Hint]

1. int number = 0; // starting number
2. loop()에서 1초 간격으로 number를 1씩 증가
3. 옆의 방식으로 결과 출력
4. number가 15를 초과하면 loop() 탈출
`exit(0); // loop 탈출 함수`

HPnn_loop_escape.png



DIY2. Escape from loop()

응용 문제 [DIY2 – hint] 0~15까지 10진수를 2진수와 16진수로 출력하는 스케치를 작성해보자

hp00_diy2

```
1 /*  
2 DIY-2  
3 */  
4  
5 // start number  
6 int number = 0;  
7  
8 // 문자열 세가지를 설정한다.  
9 String stringValue[]={"Binary:", "Hexadecimal:"}; // array  
10  
11 void setup() {  
12 // 9600bps로 시리얼 통신 설정  
13 Serial.begin(9600);  
14 }
```

```
16 void loop() {  
17  
18 // 'char Value'를 출력하고 문자열과 숫자를 변수 유형별로 출력한다.  
19 Serial.print("Number = ");  
20 Serial.print(number);  
21 Serial.print(", ");  
22 Serial.print(stringValue[0]); // stringValue 중 첫 번째 문자열 출력  
23 Serial.print(number,BIN); // 2진수 형태로 출력  
24 Serial.print(", ");  
25 Serial.print(stringValue[1]); // stringValue 중 첫 번째 문자열 출력  
26 Serial.print(number,HEX); // 16진수 형태로 출력  
27 // 줄바꿈  
28 Serial.println();  
29  
30 number++; // number 1 증가  
31  
32  
33  
34 your code !!!  
35  
36  
37  
38 delay(1000); // 1초동안 지연시킨다.  
39 }
```



DIY3. sum from 1 to 100

응용 문제 [DIY3] 1에서 100까지 정수의 합을 계산하는 스케치를 작성해보자

COM3 (Arduino/Genuino Uno)

```
Number = 85, Sum = 3655  
Number = 86, Sum = 3741  
Number = 87, Sum = 3828  
Number = 88, Sum = 3916  
Number = 89, Sum = 4005  
Number = 90, Sum = 4095  
Number = 91, Sum = 4186  
Number = 92, Sum = 4278  
Number = 93, Sum = 4371  
Number = 94, Sum = 4465  
Number = 95, Sum = 4560  
Number = 96, Sum = 4656  
Number = 97, Sum = 4753  
Number = 98, Sum = 4851  
Number = 99, Sum = 4950  
Number = 100, Sum = 5050
```

HPnn: 1 + 2 + ... + 100 =5050

[Hint]

1. int number = 0; // starting number
 2. int sum = 0;
 3. loop()에서 0.1초 간격으로 number를 1씩
증가시키면서 sum에 합한다.
-
- ✓ 옆의 방식으로 결과 출력
 - ✓ startNum 가 100을 초과하면 loop() 탈출
exit(0); // loop 탈출 함수

HPnn_sum100.png

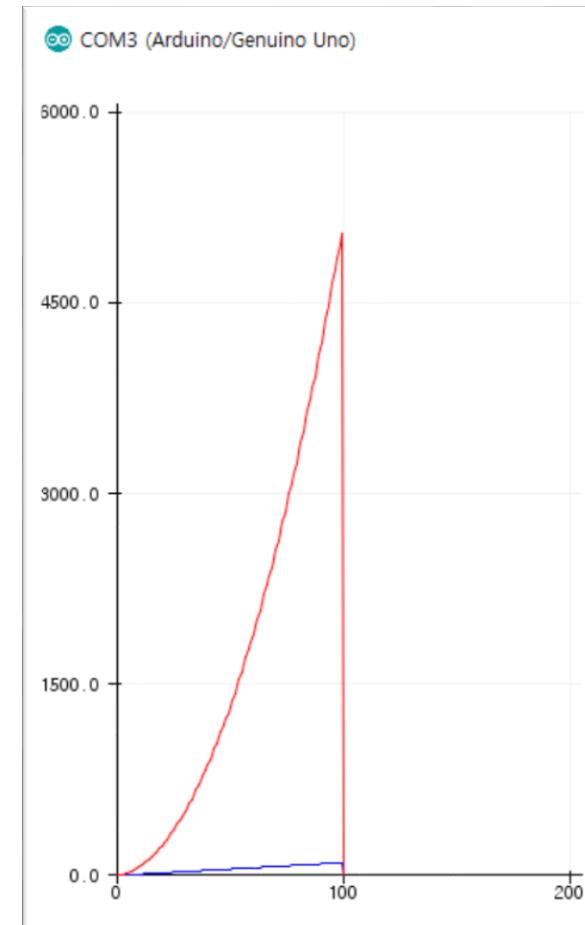
DIY3. sum from 1 to 100

응용 문제 [DIY3] Results on serial monitor and plotter

COM3 (Arduino/Genuino Uno)

```
Number = 82, Sum = 3403
Number = 83, Sum = 3486
Number = 84, Sum = 3570
Number = 85, Sum = 3655
Number = 86, Sum = 3741
Number = 87, Sum = 3828
Number = 88, Sum = 3916
Number = 89, Sum = 4005
Number = 90, Sum = 4095
Number = 91, Sum = 4186
Number = 92, Sum = 4278
Number = 93, Sum = 4371
Number = 94, Sum = 4465
Number = 95, Sum = 4560
Number = 96, Sum = 4656
Number = 97, Sum = 4753
Number = 98, Sum = 4851
Number = 99, Sum = 4950
Number = 100, Sum = 5050

HPnn: 1 + 2 + ... + 100 =5050
```

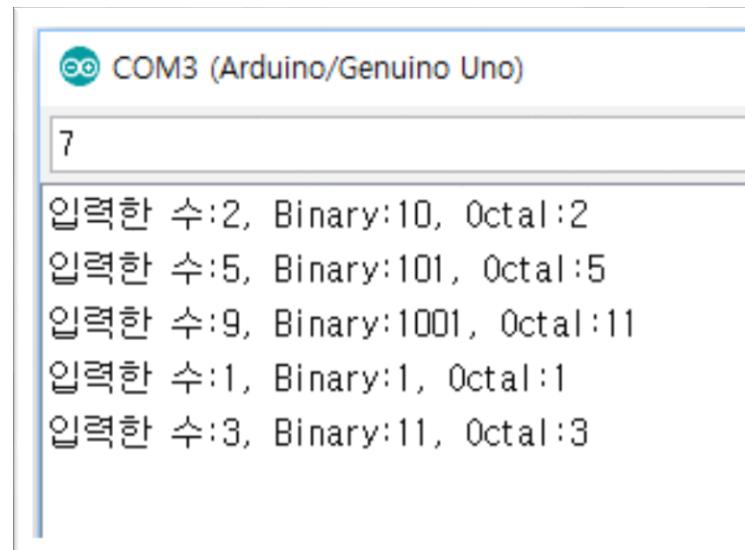




3.3 Serial monitor & plotter

3.3

시리얼 통신을 이용하여 데이터 수신하기





3.3.1 시리얼 통신을 이용하여 데이터 수신하기

EX 2.3

변수 유형별 Arduino에서 컴퓨터로 전송하기 (1/3)

- 실습목표
1. 컴퓨터에서 Arduino로 0~9의 숫자를 전송한다.
 2. Arduino에서는 전송 받은 숫자만큼 Arduino 보드의 LED를 점멸시킨다.

Hardware Arduino와 PC를 USB 케이블로 연결한다.

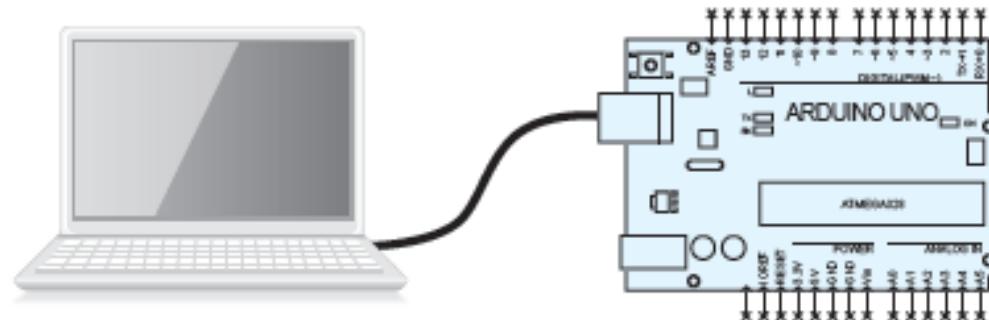


그림 2.1 Arduino와 PC와의 연결



3.3.2 시리얼 통신을 이용하여 데이터 수신하기

EX 2.3

변수 유형별 Arduino에서 컴퓨터로 전송하기 (2/3)

Commands

- `Serial.available()`

시리얼 통신에 수신된 데이터가 있는지 확인한다. 있을 경우 참(true)의 값을 갖는다.

- `Serial.read()`

시리얼 통신을 통하여 수신된 값을 읽는다.

- `isDigit(변수)`

변수의 값이 ASCII 코드의 0~9의 숫자 범위에 있는지 여부를 판단. 범위에 있을 경우 참(true)의 값을 갖는다.

- `pinMode(핀번호, 설정)`

핀의 입출력 모드를 설정한다. '핀번호'에는 설정하고자 하는 핀의 번호와 '설정'에는 입력으로 사용하기 위해선 'INPUT', 출력으로 사용하기 위해선 'OUTPUT', 입력이며 풀업 사용 시 'INPUT_PULLUP'을 적는다.

- `digitalWrite(핀번호, 값)`

핀에 디지털 출력(High or Low)을 한다. '핀번호'에는 출력하고자 하는 핀의 번호를, '값'에는 'HIGH' 혹은 'LOW'를 설정하여 High 혹은 Low 출력을 한다.



3.3.3 시리얼 통신을 이용하여 데이터 수신하기

EX 2.3

변수 유형별 Arduino에서 컴퓨터로 전송하기 (3/3)

- Sketch 구성
1. 13번 핀에 연결된 내장 LED를 이용한다.
 2. 시리얼 통신 상태를 감시한 후 시리얼 통신으로 입력되는 데이터가 있을 때 이를 저장한다.
 3. 전송된 값은 ASCII 코드값이므로 이를 숫자로 변경한다.
 4. 숫자만큼 LED를 0.2초 간격으로 점멸시킨다.
- 실습 결과
- IDE의 시리얼 모니터를 실행시켜 전송란에 0~9의 값을 입력한 후 Arduino의 LED가 입력한 값 만큼 점멸하는지를 확인해 본다..
- 응용 문제
1. 0~9의 입력 값에 따라 점멸 주기가 변화하는 스케치를 작성해 보자.
 2. 0~9의 숫자를 전송하면 전송된 수의 2진수와 16진수를 컴퓨터로 전송하는 스케치를 만들어보자. (hint: 예제 2.2를 참고하자)



DIY4. 점멸 주기가 변화

응용 문제 [DIY4] 0~9의 입력 값에 따라 점멸 주기가 변화하는 스케치를 작성해 보자.

- 시리얼모니터에 입력한 수를 표시
- 입력한 수에 비례해서 LED 켰 상태를 길게 유지.

```
COM3 (Arduino/Genuino Uno)

입력한 수:5
입력한 수:9
입력한 수:1
입력한 수:3
```

완성된 스케치 code를
[**HPnn_period_change.ino**](#)
로 저장해서 제출.



DIY5. 입력된 수를 변환하여 출력

응용 문제 [DIY5] 0~9의 숫자를 전송하면 전송된 수의 2진수와 8진수를 컴퓨터로 전송하는 스케치를 만들어보자. (hint: 예제 2.2를 참고하자).

- 아래 출력 참조.

```
COM3 (Arduino/Genuino Uno)
7
입력한 수:2, Binary:10, Octal:2
입력한 수:5, Binary:101, Octal:5
입력한 수:9, Binary:1001, Octal:11
입력한 수:1, Binary:1, Octal:1
입력한 수:3, Binary:11, Octal:3
```

완성된 스케치 **code**를
HPnn_number_output.ino
로 저장해서 제출.



[Practice]

- ◆ [wk10]
 - Arduino coding II
 - Complete your codes
 - Upload file name : HPnn_Rpt05.zip

◆ [Target of this week]

- Complete your 5th project
- Save your outcome and compress all png and ino files.

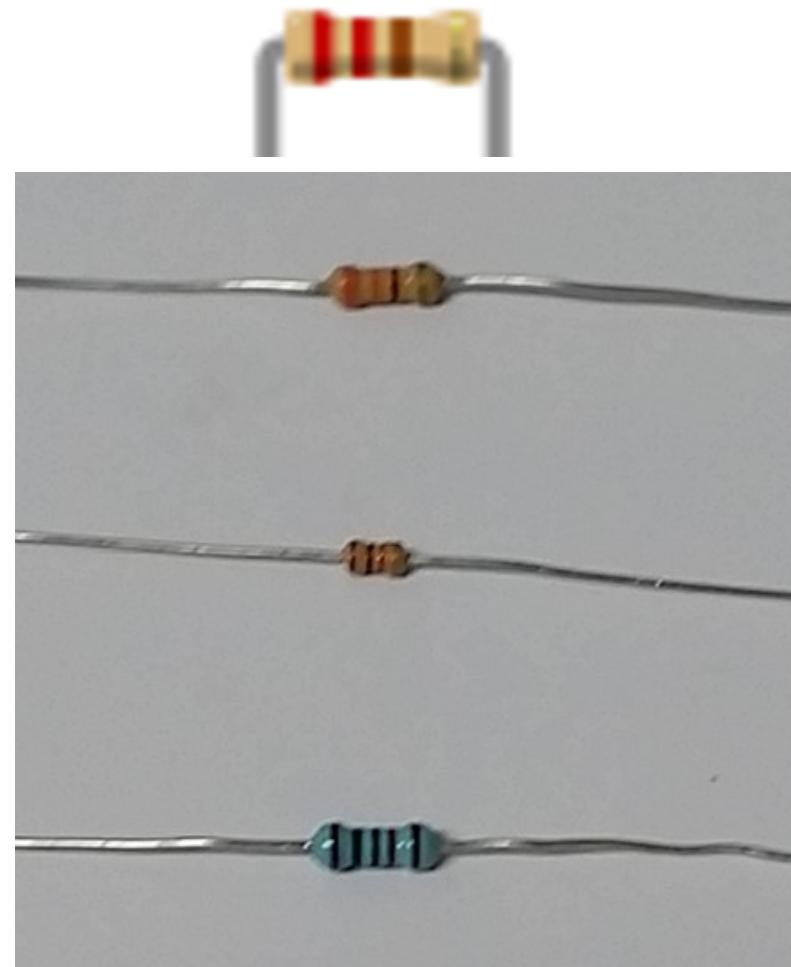
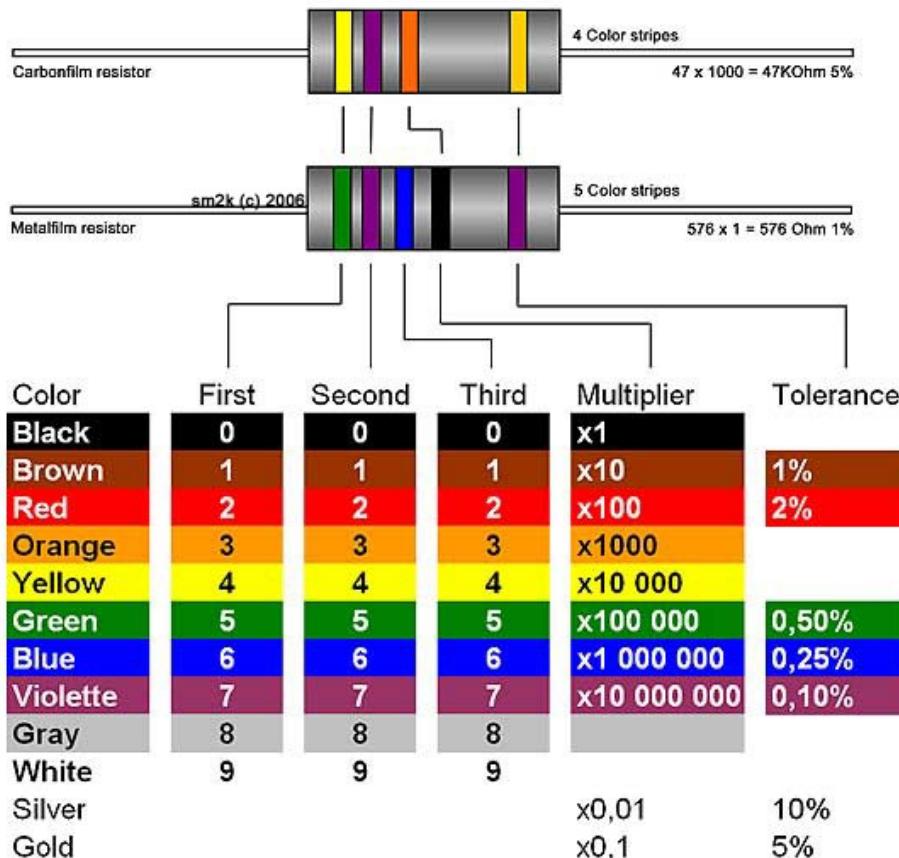
제출파일명 : **HPnn_Rpt05.zip**

- 압축할 파일들

- ① **HPnn_hello.png**
- ② **HPnn_loop_escape.png**
- ③ **HPnn_sum100.png**
- ④ **HPnn_period_change.ino**
- ⑤ **HPnn_number_output.ino**

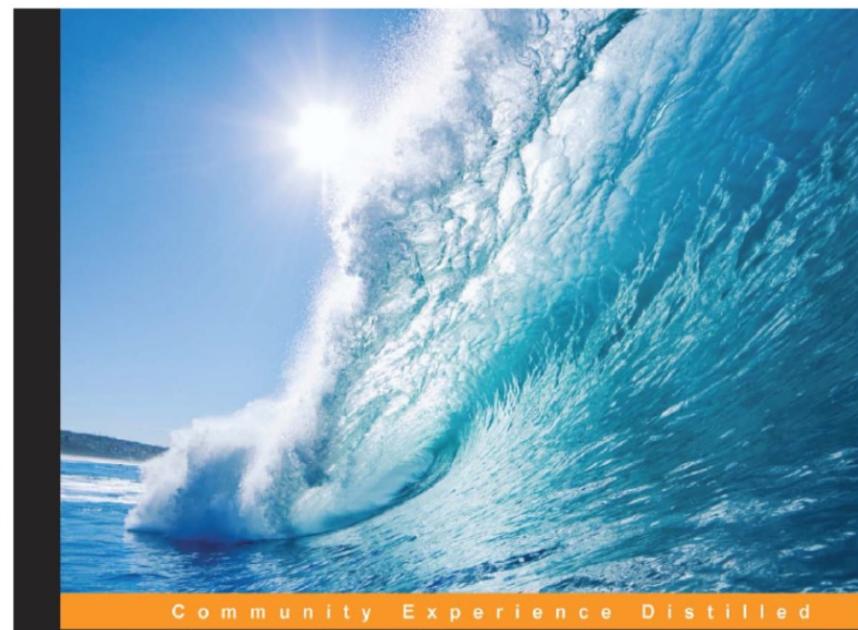


[참고 : 저항 값 읽기]



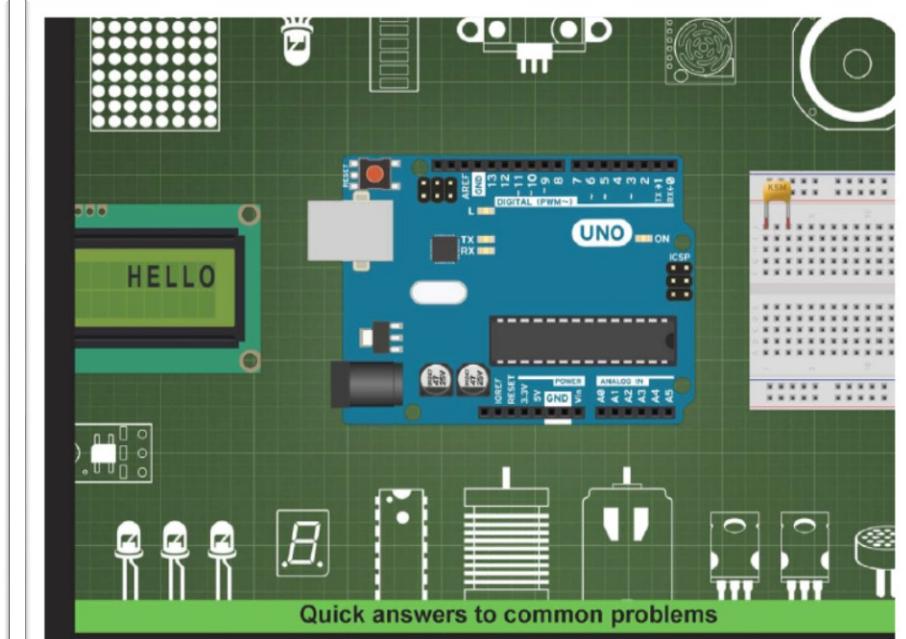


References



Francis Perea

[PACKT]
PUBLISHING



Cornel Amariei

[PACKT] open source*
PUBLISHING