

주니어 임베디드SW 챌린저

-1차 기술 지원 교육-

2018. 07. 14

- Python 개발 환경 구축

- 프로그래밍 기초

- 변수
- 조건문
- 반복문
- 함수

- 로봇 제어 기초

- 모터제어
- 컬러 센서

- 로봇 제어 응용

- 정확한 회전
- 사각형 그리기
- 검은 선 카운트
- 라인트레이싱

- 마무리

Python 개발 환경 구축

• 구축 순서

1. EV3Dev 이미지 파일 다운로드 -> 수정
2. 부팅 SD 카드 제작
3. EV3Dev 부팅
4. Python3 설치
5. Visual Studio Code 설치
6. Visual Studio Code Extension 설치
7. EV3와 PC 연결

• 주의 할 점

- 사용하는 PC의 OS는 WINDOWS 10을 권장합니다.
- Windows 10 미만 버전의 OS 환경에서는 ev3dev와의 호환성에 문제가 있을 수 있습니다.
- 공식적인 기술 지원 세미나와 QNA는 Windows 10을 기준으로 제공됩니다.
- 사용하시는 PC의 OS가 32비트와 64비트 중 어느 것인지 반드시 확인 하시고 알맞은 버전의 프로그램을 설치하시기 바랍니다.
- 이 프레젠테이션은 Windows 10 64비트 버전을 기준으로 작성되어 있습니다.

1. EV3Dev 이미지 파일 다운로드

Python을 이용해서 프로그래밍을 하려면 이미지 파일을 받아 SD카드에 설치하고 SD카드를 이용하여 EV3를 실행해야 한다.

<http://www.ev3dev.org/download/>에 접속하여 EV3Dev를 다운로드한다.

<https://oss.jfrog.org/list/oss-snapshot-local/org/ev3dev/brickstrap/2018-07-15/> : 최신버전

Index of oss-snapshot-local/org/ev3dev/brickstrap/2018-07-15

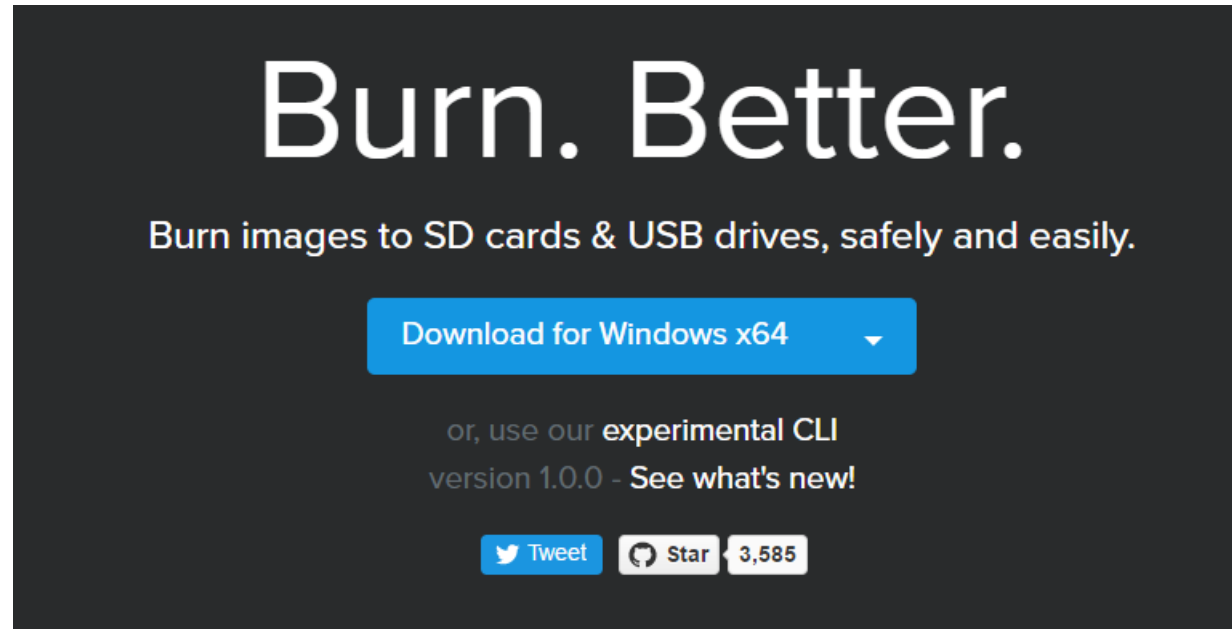
Name	Last modified	Size
../		
snapshot-ev3dev-stretch-bone-generic-2018-07-15.img.xz	15-Jul-2018 21:09	227.15 MB
snapshot-ev3dev-stretch-ev3-generic-2018-07-15.img.xz	15-Jul-2018 21:09	205.33 MB
snapshot-ev3dev-stretch-rpi-generic-2018-07-15.img.xz	15-Jul-2018 21:09	246.04 MB
snapshot-ev3dev-stretch-rpi2-generic-2018-07-15.img.xz	15-Jul-2018 21:08	239.53 MB

Artifactory/6.1.0 Server at oss.jfrog.org Port 443

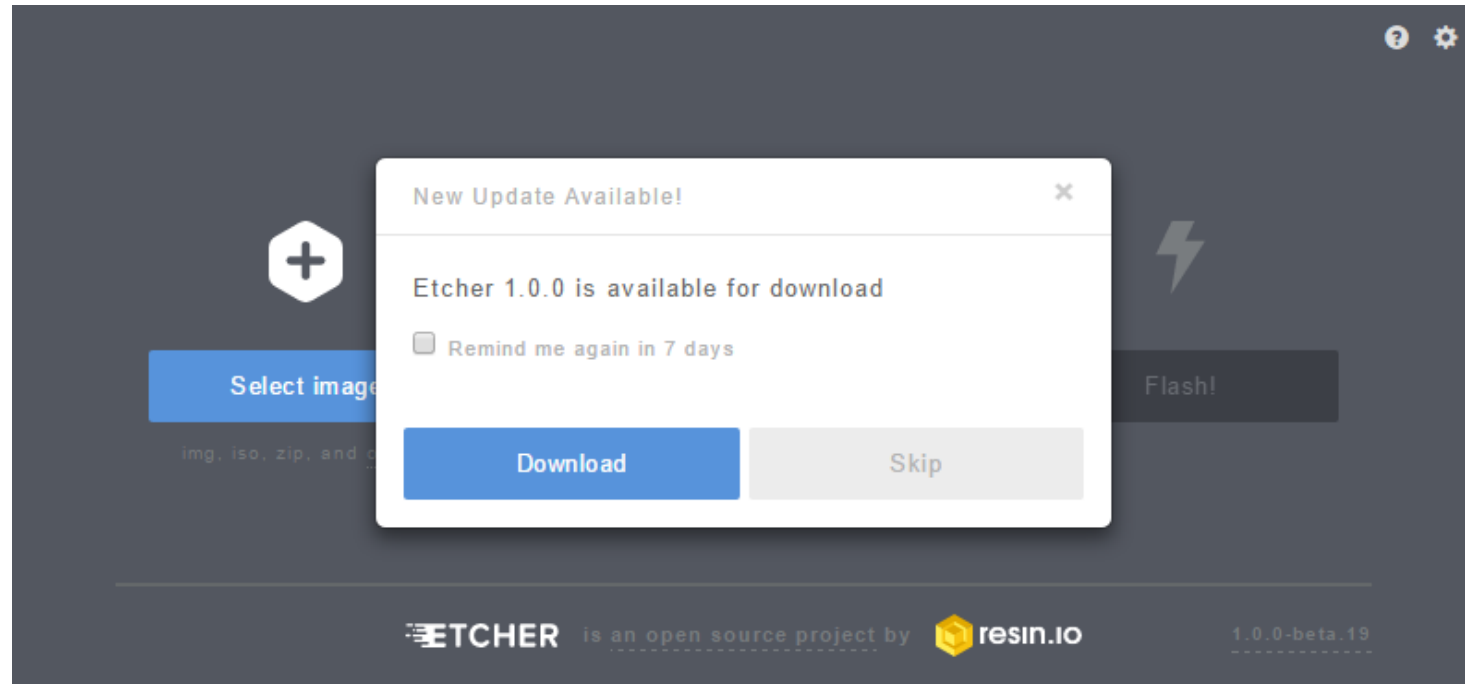
2. 부팅 SD카드 제작

부팅 SD카드를 제작하기 위해선 'Etcher'라는 프로그램을 설치해야한다.

- ① <https://etcher.io/>에 접속하여 다운로드한다.
- ② 다운로드된 Etcher-1.0.0-beta.19-win32-x64.exe를 실행하여 설치를 진행한다.

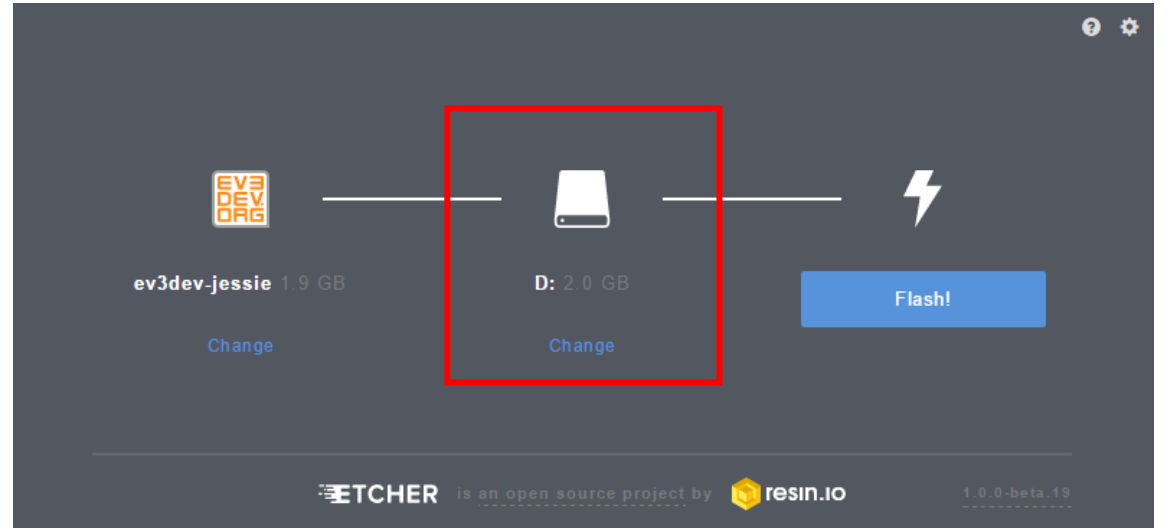
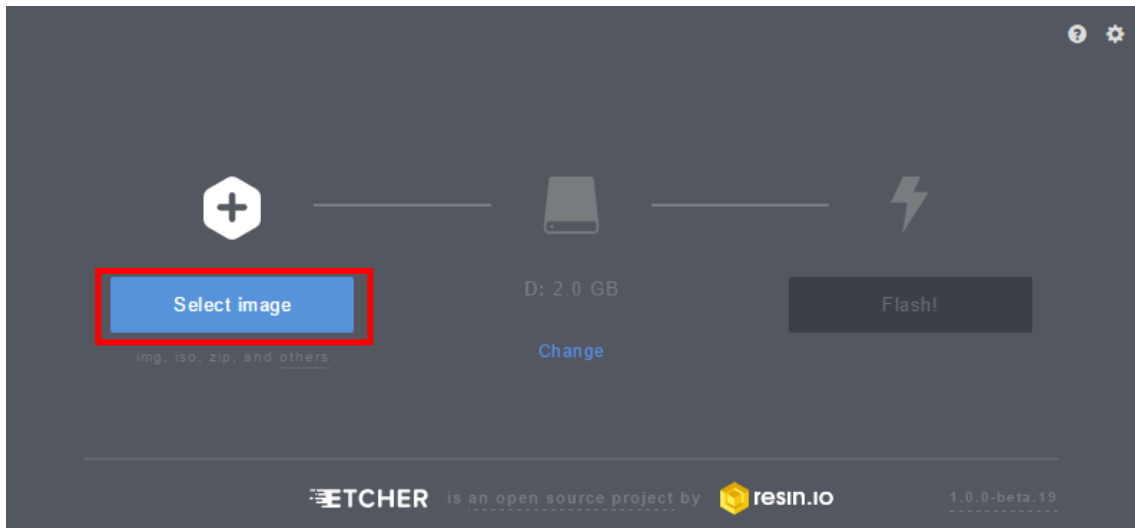


- ③ 설치가 완료된 'Etcher'를 실행한다.
- ④ 버전이 업데이트 되어 아래와 같은 창이 활성화 되었을 경우, Download를 클릭하여 새 버전을 다운로드하거나 Skip을 클릭하여 기존 버전을 계속 사용할 수 있다.

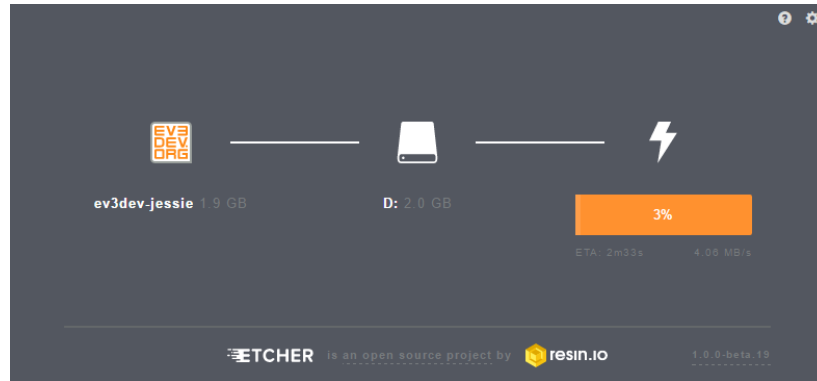
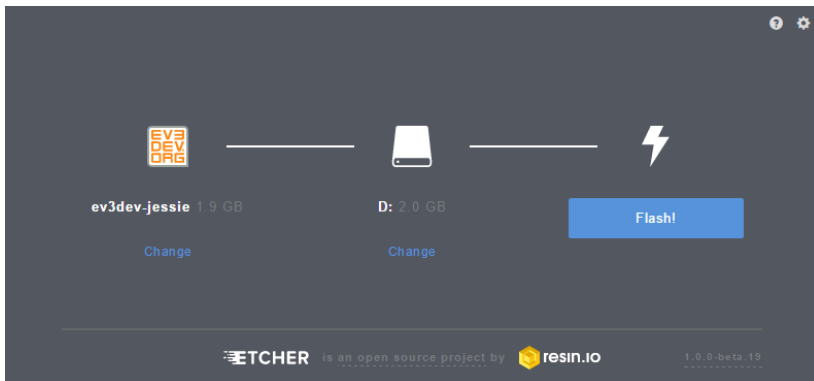
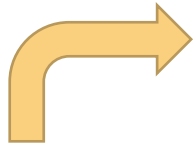


‘Etcher’를 실행하였으면 아래와 같은 순서대로 진행한다.

- ① SD카드 리더기를 이용하여 SD카드(8Gb~32Gb)를 PC에 연결한다.
- ② Select image를 클릭하여 1에서 다운로드한 EV3Dev 압축 파일을 선택한다.
- ③ SD카드의 경로가 제대로 설정 되었는지 확인한다. 경로가 다를 경우 Change를 클릭하여 올바른 SD카드 경로를 설정해준다.



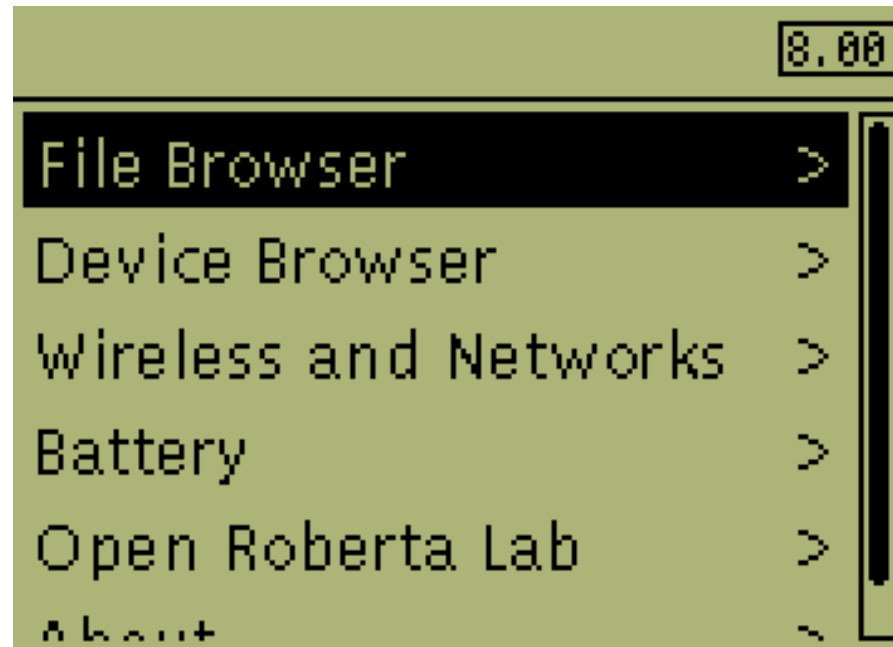
- ④ '경로를 설정하고 Flash!를 클릭한다.
- ⑤ 설치가 완료되면 Flash Complete!라는 화면으로 바뀐다.



3. EV3Dev 부팅

EV3 컨트롤러의 SD카드 슬롯에 만들어진 SD카드를 삽입하고, EV3 전원을 켜다.

부팅이 완료되면 아래와 같은 EV3 화면을 확인 할 수 있다.

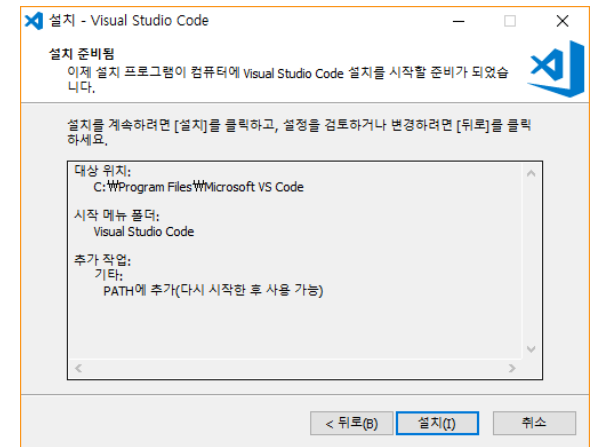
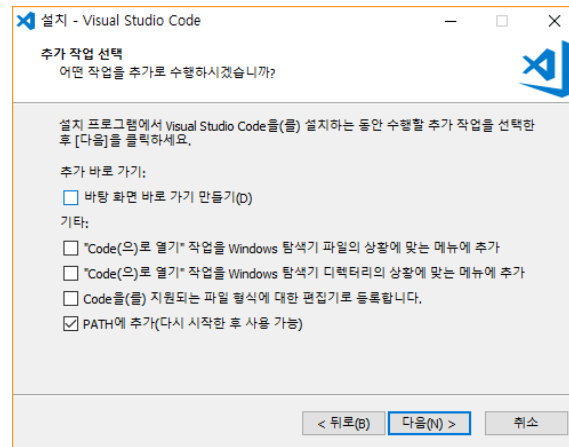
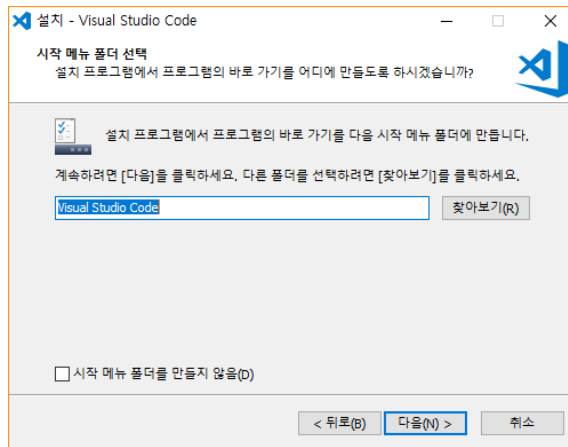
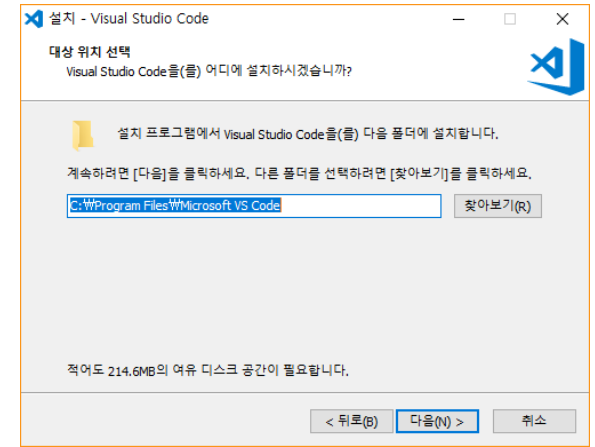
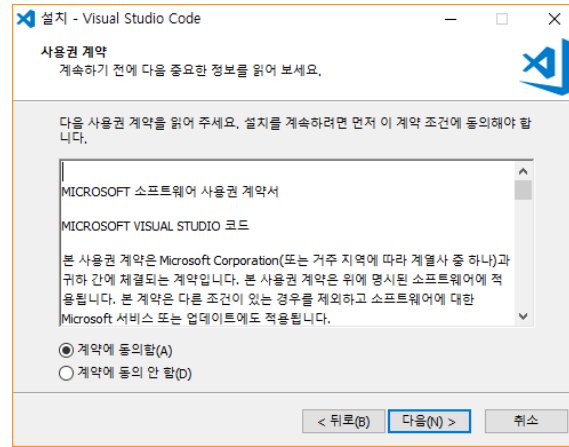
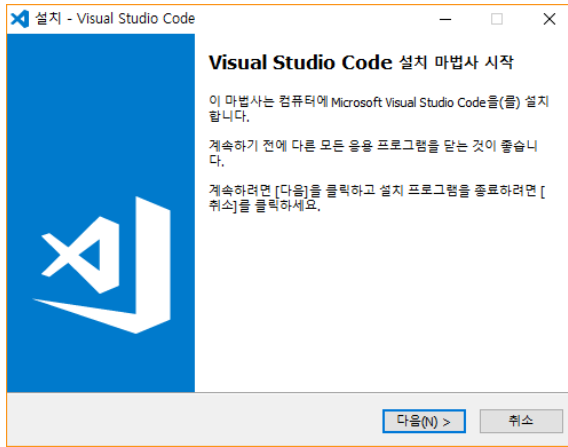


3. Visual Studio Code

<https://code.visualstudio.com/>으로 접속하여 설치파일을 다운로드

The image shows a composite of two parts. On the left is the Visual Studio Code website landing page, which features the text "Code editing. Redefined." and "Free. Open source. Runs everywhere." A green button labeled "Download for Windows" with "Stable Build" underneath is highlighted with a red rectangle. Below this button is a link for "Other platforms and Insiders Edition" and a note: "By using VS Code, you agree to its license and privacy statement." On the right is a screenshot of the Visual Studio Code IDE interface. The "EXTENSIONS" sidebar is open, showing a list of installed and available extensions such as C#, Python, Debugger for Chrome, C/C++, Go, and ESLint. The main editor area displays a TypeScript file named "app.ts" with code for an Express.js server. The code includes imports for 'app', 'debugModule', and 'http', and defines a 'normalizePort' function. The status bar at the bottom indicates the current file is "app.ts" and the editor is in "TypeScript" mode.

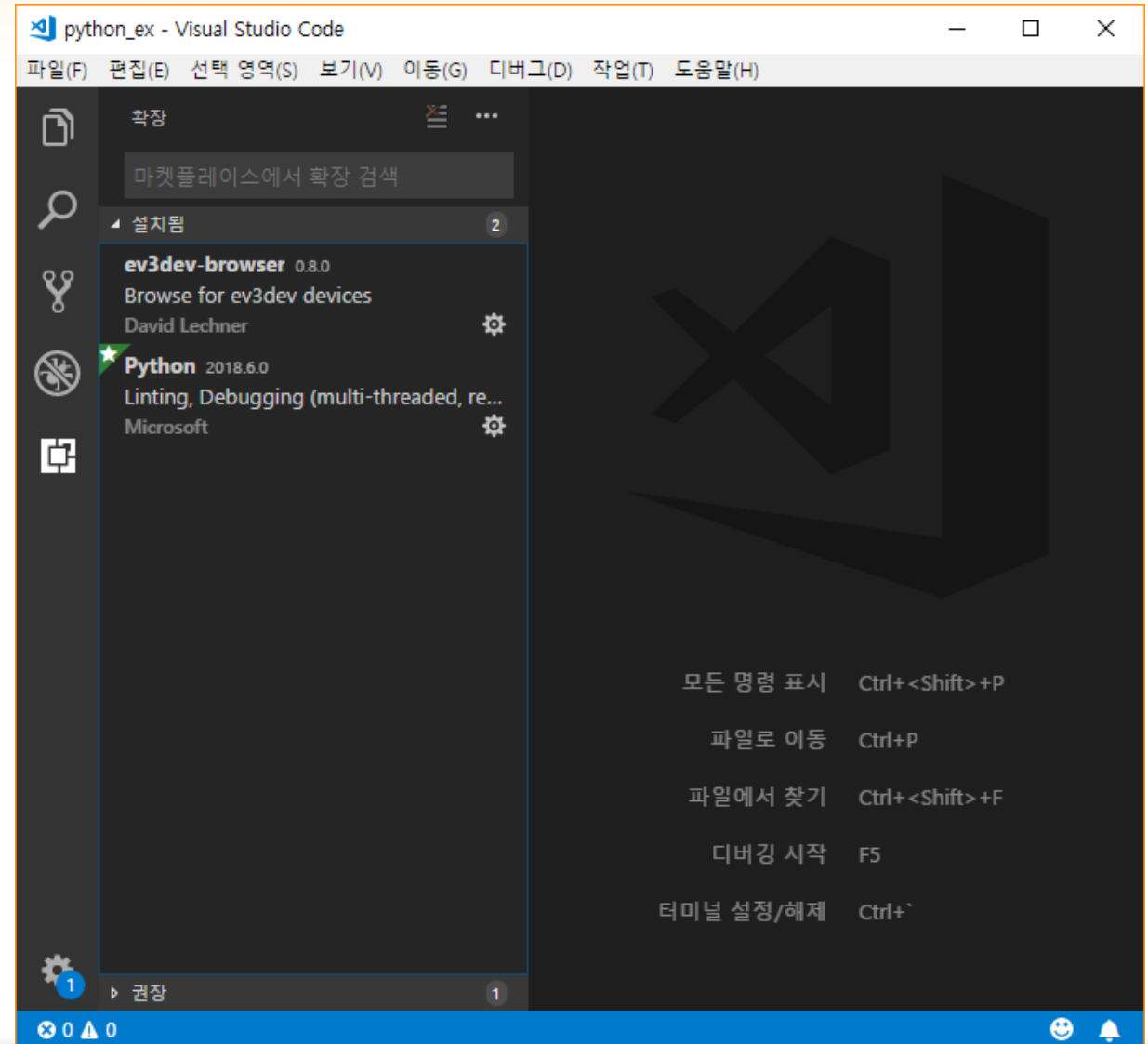
I. Visual Studio Code 설치



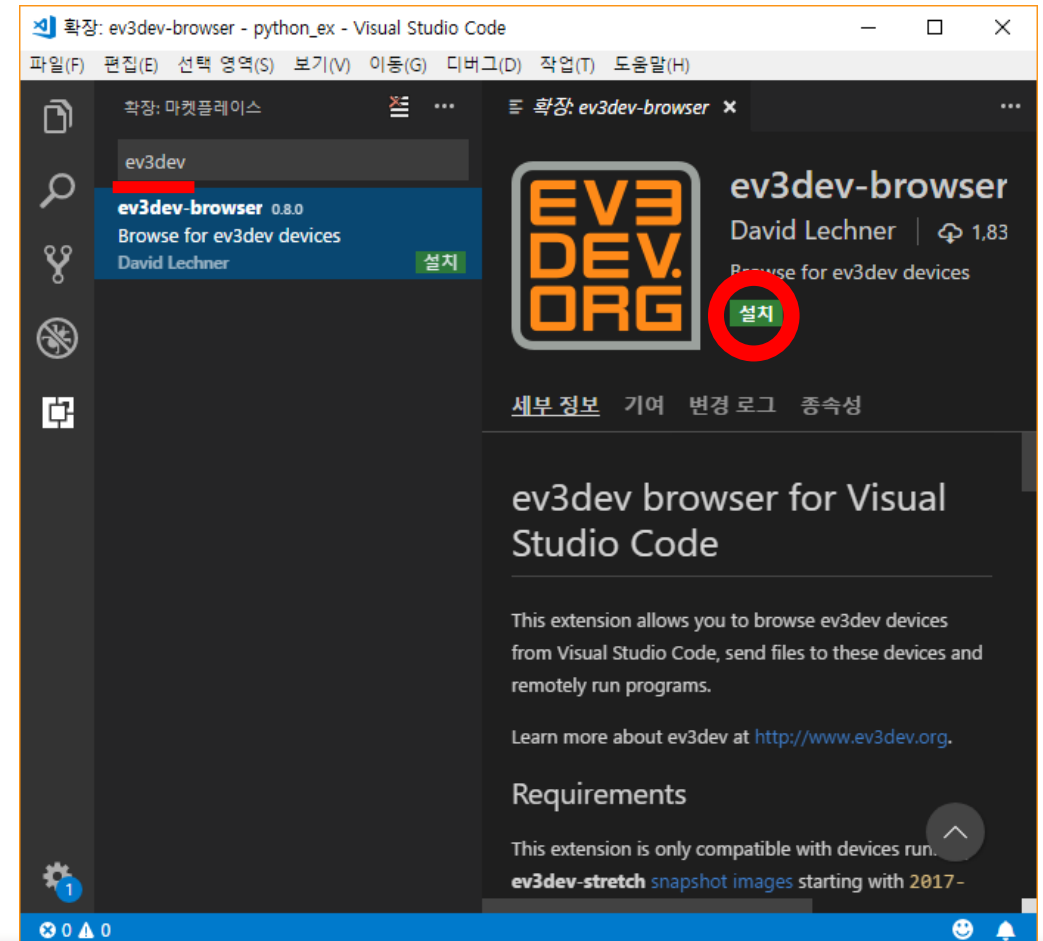
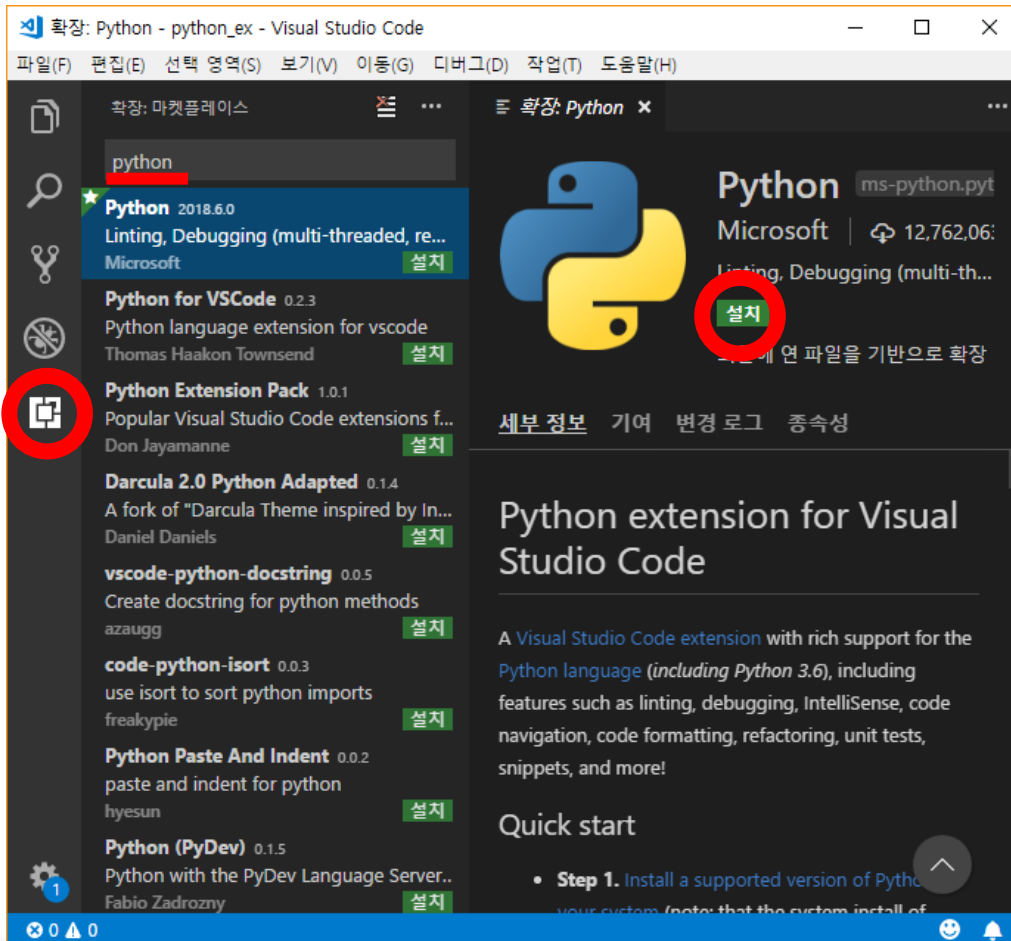
II. Visual Studio Code 실행

- 실행 시 주의 할 점

실행 아이콘을 오른쪽 클릭을 하여
"관리자 권한으로 실행"
을 선택한다.



III. 확장 에드온 설치



IV. vscode-hello-python-master 프로젝트 다운로드

<https://github.com/ev3dev/vscode-hello-python> 으로 접속하여 압축파일 다운로드

ev3dev / vscode-hello-python

Watch 3 Star 2 Fork 0

Code Pull requests 0 Insights

Join GitHub today

GitHub is home to over 28 million developers working together to host and review code, manage projects, and build software together.

Sign up

Hello World for ev3dev + Visual Studio Code + Python

18 commits 1 branch 0 releases 1 contributor MIT

Branch: master New pull request

Find file Clone or download

Clone with HTTPS

Use Git or checkout with SVN using the web URL.

<https://github.com/ev3dev/vscode-hello-p>

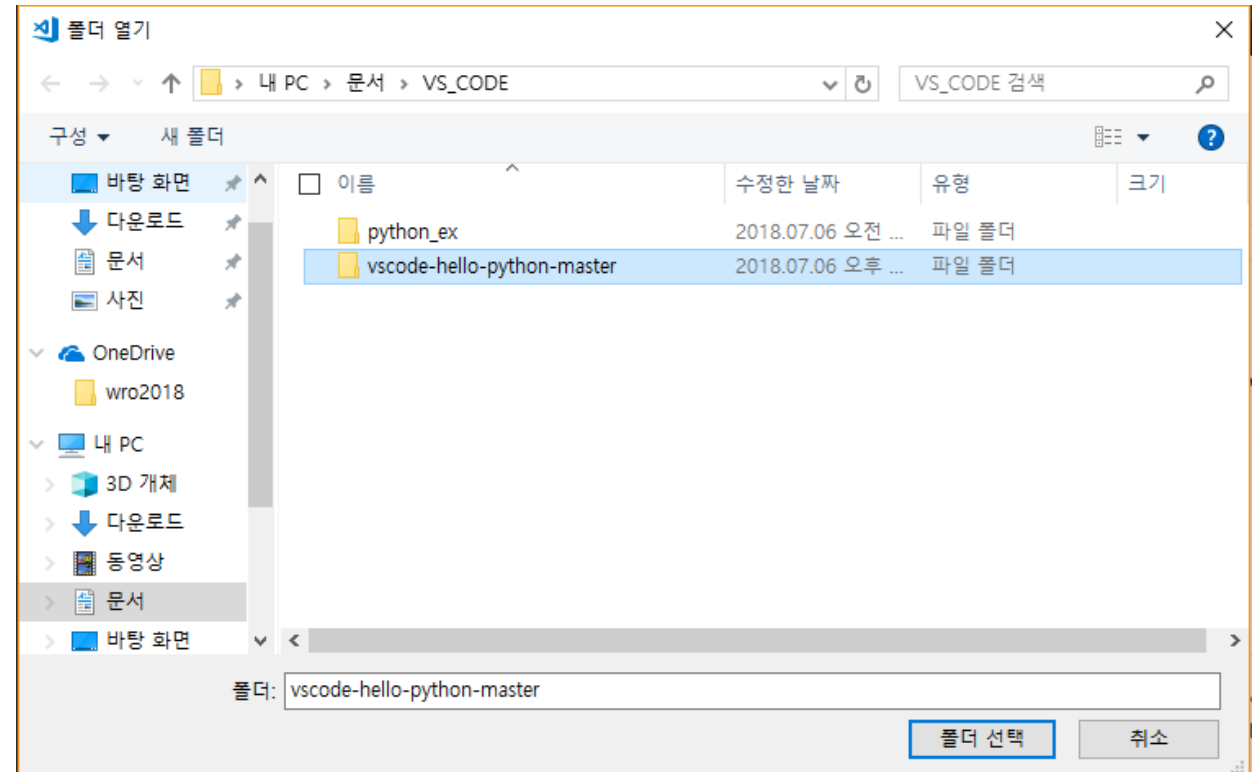
Open in Desktop Download ZIP

.README	Cut down steps by downloading zip instead of cloning git
.vscode	Remove unused setting
.gitattributes	add .gitattributes
.gitignore	Initial commit

V. vscode-hello-python-master 프로젝트 열기

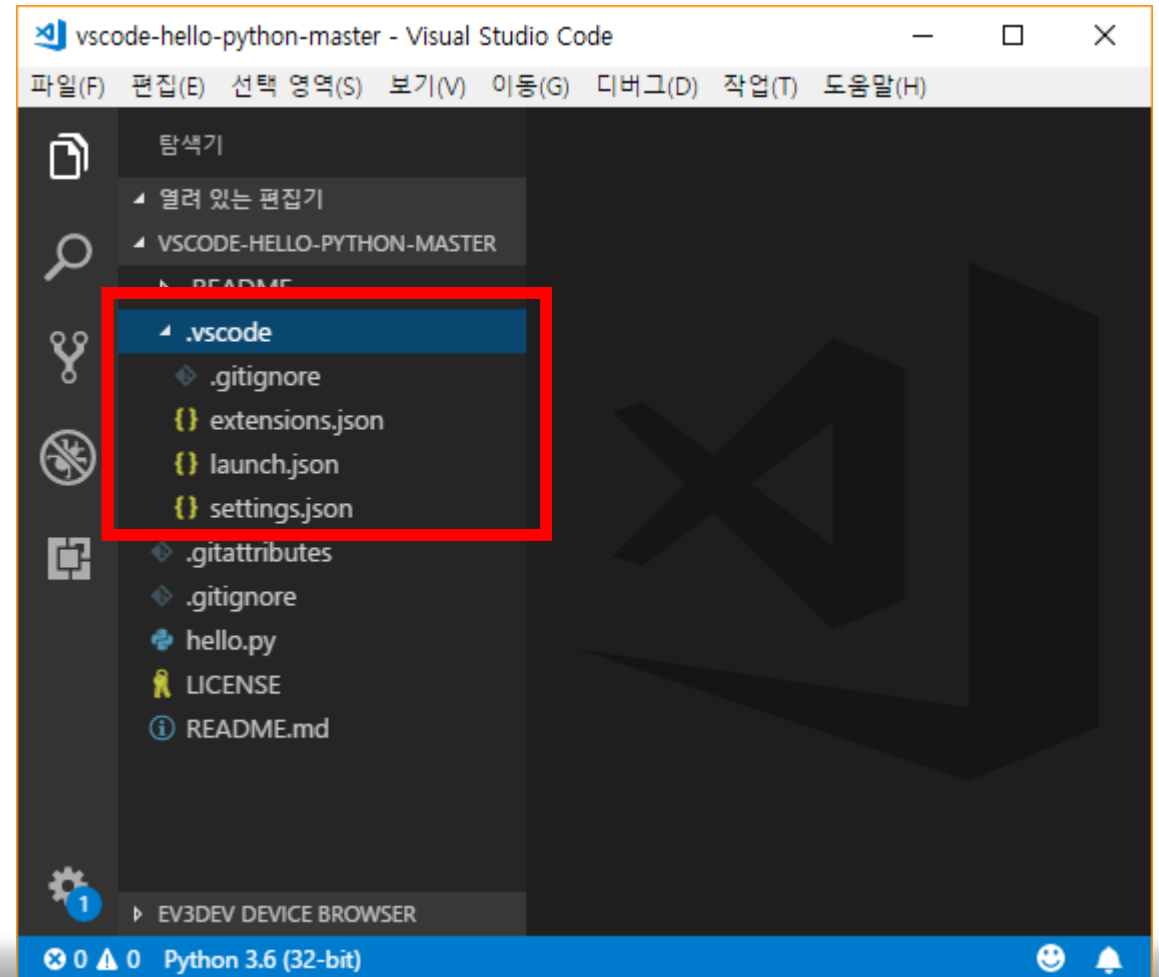
압축을 해제한 폴더를 VS Code에서 열기

1. 도구모음의 파일
2. 폴더 열기
3. Vscod-hello-python-master 선택
4. 폴더 선택 버튼



VI. vscode-hello-python 프로젝트 내용

- 빨간색 사각형 안의 .vscode 폴더와 launch.json, settings.json 파일은 ev3 프로그래밍을 위해 반드시 필요한 구성 요소
- 후에 새로운 프로젝트 폴더를 만들 경우 복사하여 넣어주면 된다



VI. launch.json 파일 내용 수정

- 현재 화면에 표시되고 있는 프로그램을 실행 시킬 수 있도록 도와주는 설정

```
launch.json ×
1 {
2   "version": "0.2.0",
3   "configurations": [
4     {
5       "name": "Download and Run",
6       "type": "ev3devBrowser",
7       "request": "launch",
8       "program": "/home/robot/${workspaceRootFolderName}/hello.py"
9     }
10  ]
11 }
```

Hello.py -> \${relativeFile}

```
launch.json ×
1 {
2   "version": "0.2.0",
3   "configurations": [
4     {
5       "name": "Download and Run",
6       "type": "ev3devBrowser",
7       "request": "launch",
8       "program": "/home/robot/${workspaceRootFolderName}/${relativeFile}"
9     }
10  ]
11 }
```

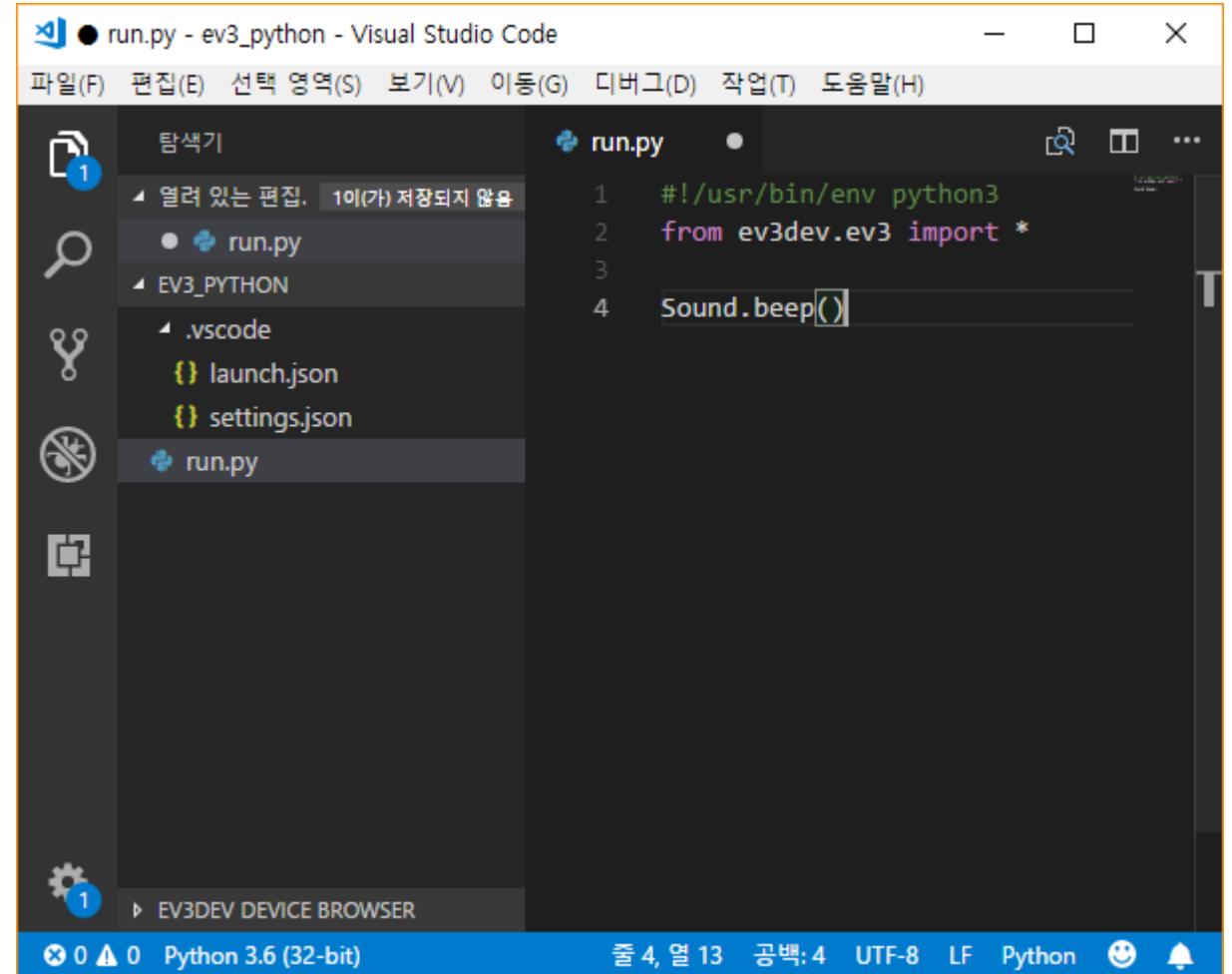
VII. 간단한 코드 작성

새로운 프로젝트 폴더를 생성

- 도구모음의 파일 -> 새 파일 선택
- 파일의 이름을 run.py로 저장

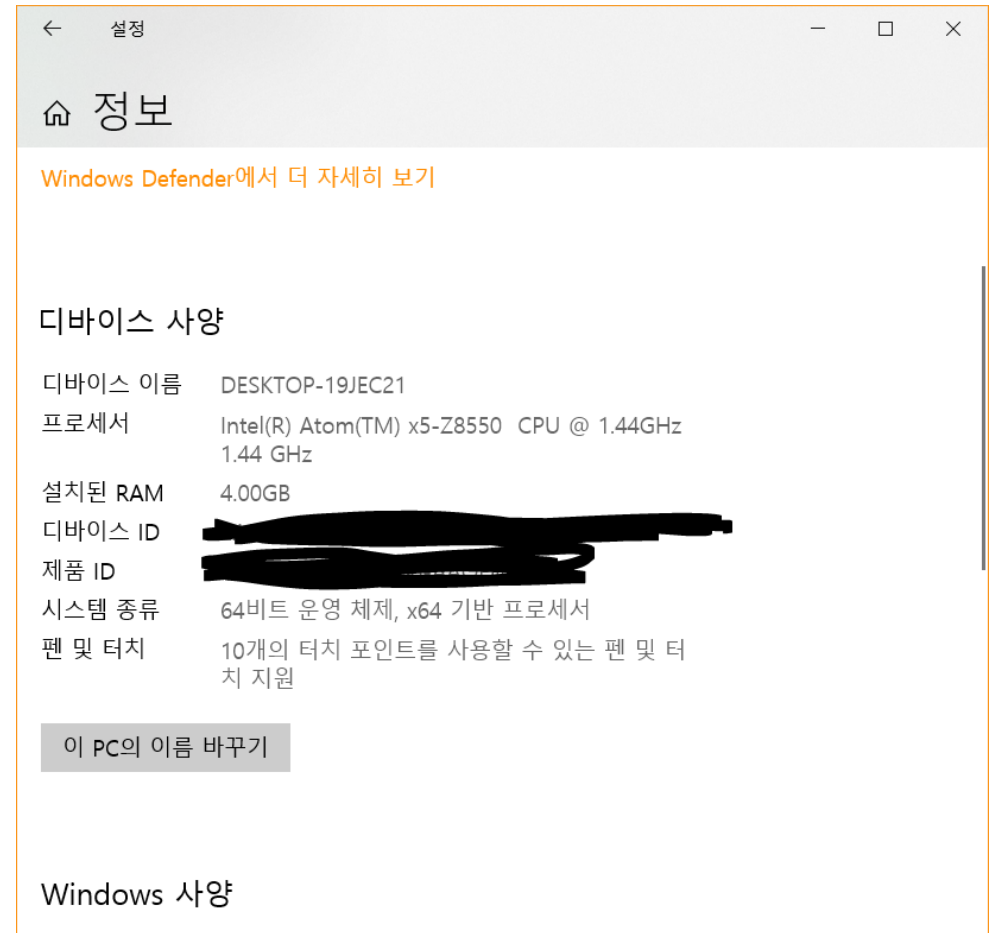
```
#!/usr/bin/env python3  
from ev3dev.ev3 import *
```

```
Sound.beep()
```



5. EV3와 PC 연결하기 - 블루투스 연결

1. 사용하는 노트북의 이름 찾기
2. https://youtu.be/niziYwWA_7U
3. https://youtu.be/mdwFQ2sr_mU
4. https://youtu.be/d5mdD_TOerl



6. EV3와 PC 연결하기 - 케이블 연결

1. <https://youtu.be/OZHH5t4BjNI>

프로그래밍 기초

• 변수 – 컨테이너, 가방, 그릇, 풍선

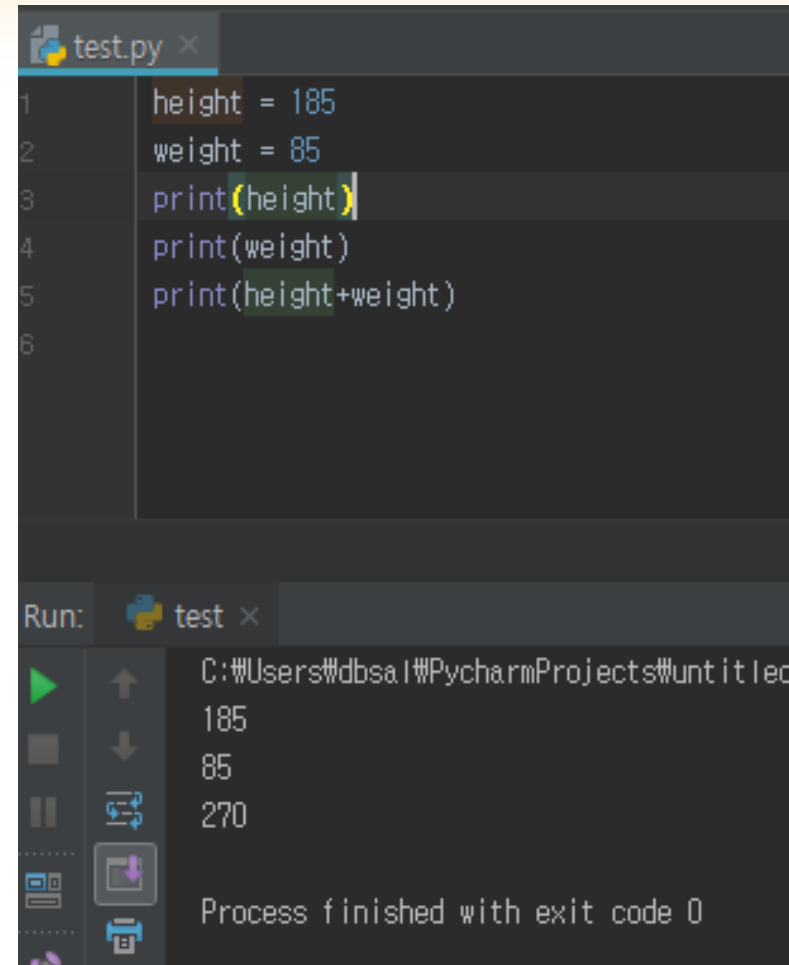
- 다양한 데이터를 보관할 수 있는 공간
- 변수 안에 넣어 둔 데이터는 언제 어디서 열어 봐도 항상 동일 하다.
- 변수의 내용물은 언제든지 바꿀 수 있다.

• 변수 선언 시 유의 사항

1. 변수명은 영문자(A-Z, a-z), 밑줄문자(_), 숫자를 조합하여 구성한다.
2. 변수명의 첫글자는 반드시 영문자나 밑줄문자(_)로 시작해야 한다.
3. 대소문자를 구분해서 사용해야 한다.
4. 이미 다른 용도로 사용되고 있는 키워드 또는 예약어는 변수명으로 사용할 수 없다.

- 변수 선언하기

```
height = 185  
weight = 85  
print(height)  
print(weight)  
print(height+weight)
```



The screenshot shows a Python IDE window titled 'test.py' with the following code:

```
1 height = 185  
2 weight = 85  
3 print(height)  
4 print(weight)  
5 print(height+weight)  
6
```

Below the code editor is a 'Run' console window titled 'test' showing the output of the program:

```
C:\Users\dbsa1\PycharmProjects\untitled\test.py  
185  
85  
270  
Process finished with exit code 0
```

- 조건문 - 특정한 조건에 따라 실행 여부가 결정된다.
 - if 조건 : - 만약 ~이면
 - elif 조건 : - 아니면 만약 ~이면
 - else : - 위 조건들이 아닌 모든 경우
- 예시) 불량품이 발생 했을 때 상점의 대처
 - if 구매 후 3일 이전: 환불
 - elif 구매 후 3일 이후: 교환
 - elif 구매 후 7일 이후: 환불/교환 불가
 - else : 경찰 부름

• if – elif – else 사용하기

```
number = 12

if number < 5:
    print("5보다 작아요")
elif number == 5:
    print("5입니다")
elif number > 5:
    print("5보다 커요")
else:
    print("알 수 없어요")
```

```
run.py x
1 number = 12
2
3 if number < 5:
4     print("5보다 작아요")
5 elif number == 5:
6     print("5입니다")
7 elif number > 5:
8     print("5보다 커요")
9 else:
10    print("알 수 없어요")
11
12
13
run run
"C:\Program Files (x86)\Python36-32
5보다 커요
Process finished with exit code 0
```

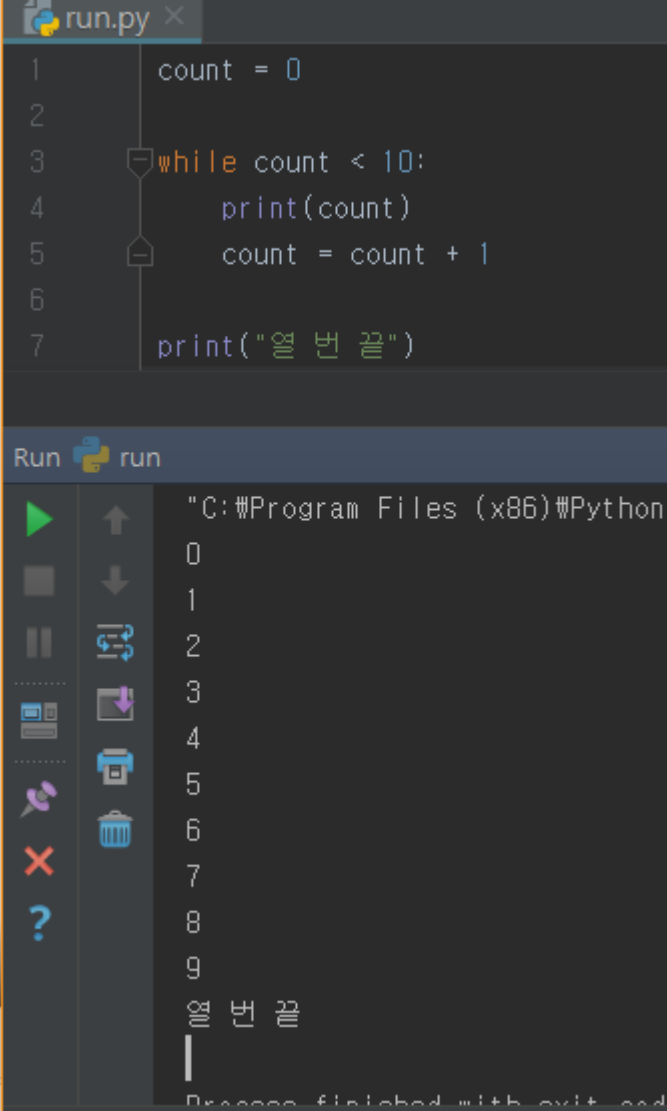
- 반복문 - 특정한 조건에 따라 반복 여부가 결정된다.
 - while 조건: 조건을 만족 하는 동안 반복
 - for 변수 in 리스트: 리스트 안의 모든 경우 만큼 반복
- 예시) 불량품이 발생 했을 때 상점의 대처
 - while 물건이 불량품인 경우: 교환

• while 사용하기

```
count = 0
```

```
while count < 10:  
    print(count)  
    count = count + 1
```

```
print("열 번 끝")
```



The screenshot shows a Python IDE window titled 'run.py'. The code in the editor is as follows:

```
1 count = 0  
2  
3 while count < 10:  
4     print(count)  
5     count = count + 1  
6  
7 print("열 번 끝")
```

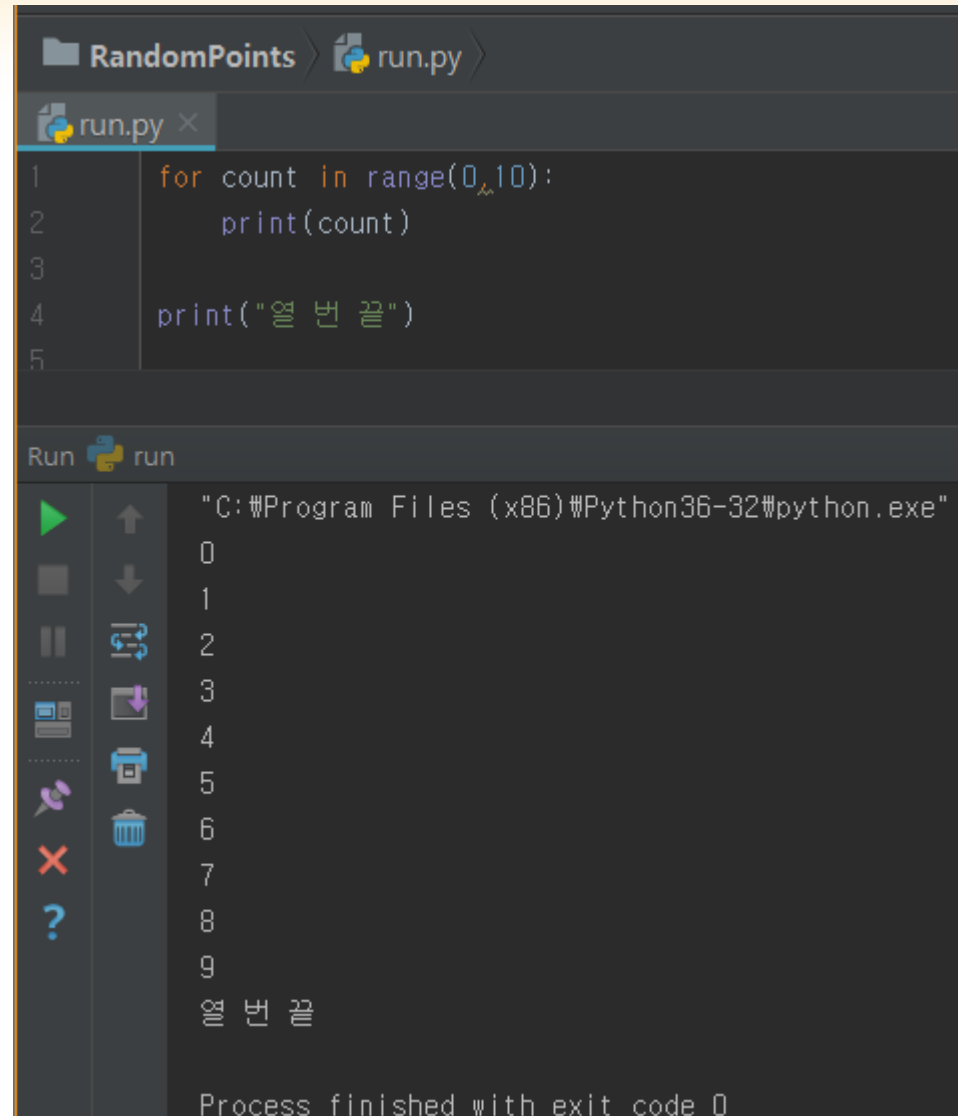
Below the editor is a 'Run' panel with a 'run' button. The output console shows the execution results:

```
"C:\#Program Files (x86)\#Python  
0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
열 번 끝  
Process finished with exit code
```

- for 사용하기

```
for count in range(0,10):  
    print(count)
```

```
print("열 번 끝")
```



The screenshot shows a Python IDE window titled 'RandomPoints' with a file named 'run.py'. The code in the editor is:

```
1 for count in range(0,10):  
2     print(count)  
3  
4     print("열 번 끝")  
5
```

Below the editor is a 'Run' button and a terminal window. The terminal shows the output of the program:

```
Run run  
"C:\Program Files (x86)\Python36-32\python.exe"  
0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
열 번 끝  
Process finished with exit code 0
```

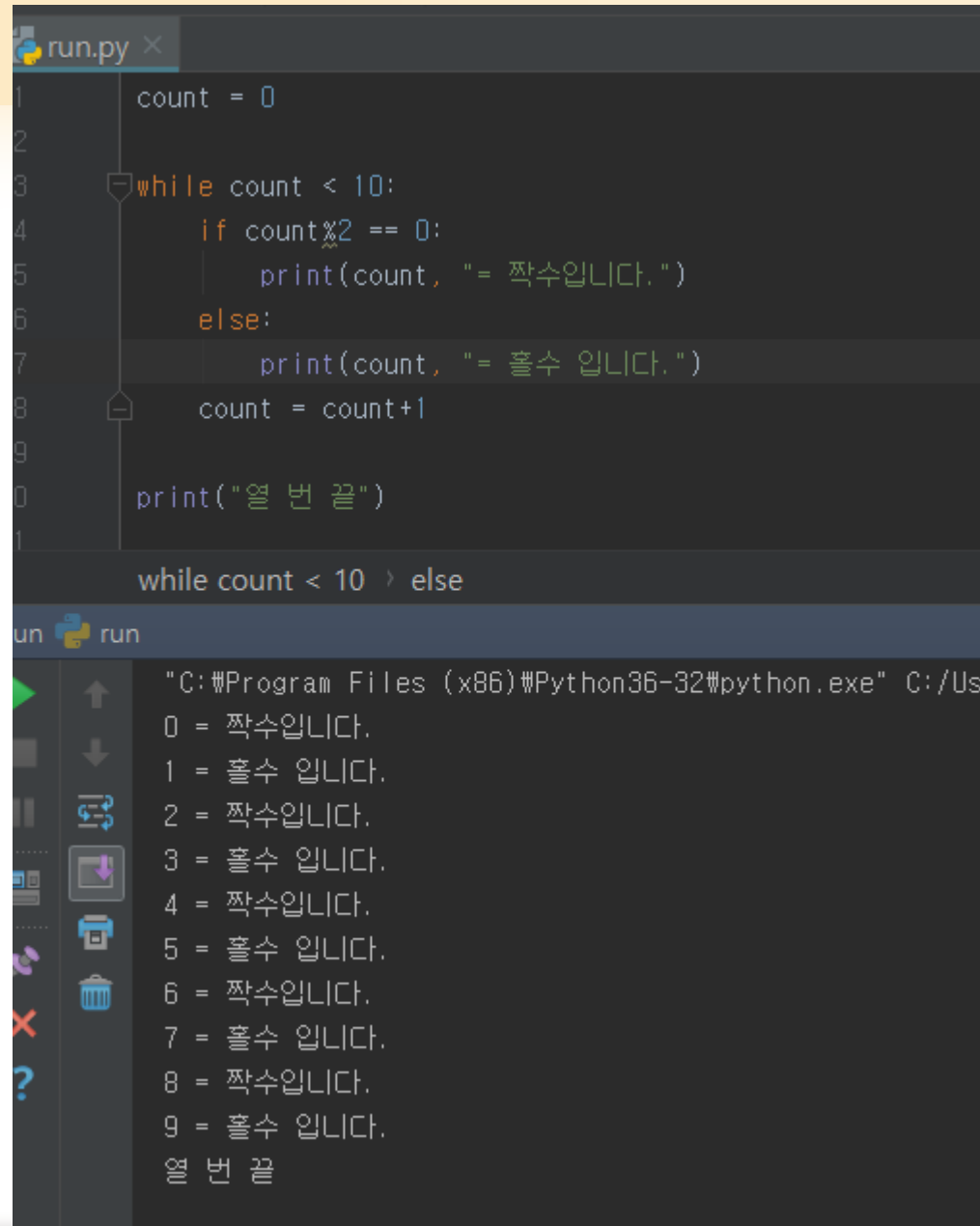
- 혼합문 - 조건문과 반복문이 동시에 쓰이는 경우
- 예시) 불량품이 발생 했을 때 상점의 대처
 - while 물건이 불량품인 동안:
 - if 교환 횟수가 5번 이하: 교환
 - else : 환불

• 혼합문 사용하기

```
count = 0
```

```
while count < 10:  
    if count%2 == 0:  
        print(count, "= 짝수입니다.")  
    else:  
        print(count, "= 홀수 입니다.")  
    count = count+1
```

```
print("열 번 끝")
```



The screenshot shows a Python IDE window titled 'run.py' with the following code:

```
1 count = 0  
2  
3 while count < 10:  
4     if count%2 == 0:  
5         print(count, "= 짝수입니다.")  
6     else:  
7         print(count, "= 홀수 입니다.")  
8     count = count+1  
9  
10 print("열 번 끝")
```

Below the code editor, the execution output is displayed:

```
run run  
"C:\Program Files (x86)\Python36-32\python.exe" C:/Us  
0 = 짝수입니다.  
1 = 홀수 입니다.  
2 = 짝수입니다.  
3 = 홀수 입니다.  
4 = 짝수입니다.  
5 = 홀수 입니다.  
6 = 짝수입니다.  
7 = 홀수 입니다.  
8 = 짝수입니다.  
9 = 홀수 입니다.  
열 번 끝
```


• 연산자

- 연산자를 활용하면 변수에 저장된 값을 다양하게 조작하여 사용할 수 있다.
- 프로그램 상에 괄호 없이 두 개 이상의 연산자를 한 문장에서 혼합하여 사용하게 되면, 연산자의 우선 순위에 따라 실행된다.

<산술 연산자>

연산자	의미	예시	연산 결과
+	덧셈	4 + 2	6
-	뺄셈	4 - 2	2
/	나눗셈	4 / 2	2
*	곱셈	4 * 2	8
%	나머지	4 % 2	0

<관계 연산자>

연산자	의미	예시	연산 결과
<	a가 b보다 작다	4 < 2	false
>	a가 B보다 크다	4 > 2	true
==	a와 b가 같다	4 == 2	false
!=	a와 b가 같지 않다	4 != 2	true
<=	a가 b보다 작거나 같다	4 <= 2	False
>=	a가 b보다 크거나 같다	4 >= 2	true

<대입 연산자>

연산자	의미	예시	연산 결과
+=	덧셈 누적	a+=1	a=a+1
-=	뺄셈 누적	a-=1	a=a-1
/=	나눗셈 누적	a/=1	a=a/1
=	곱셈 누적	a=1	a=a*1

<논리 연산자>

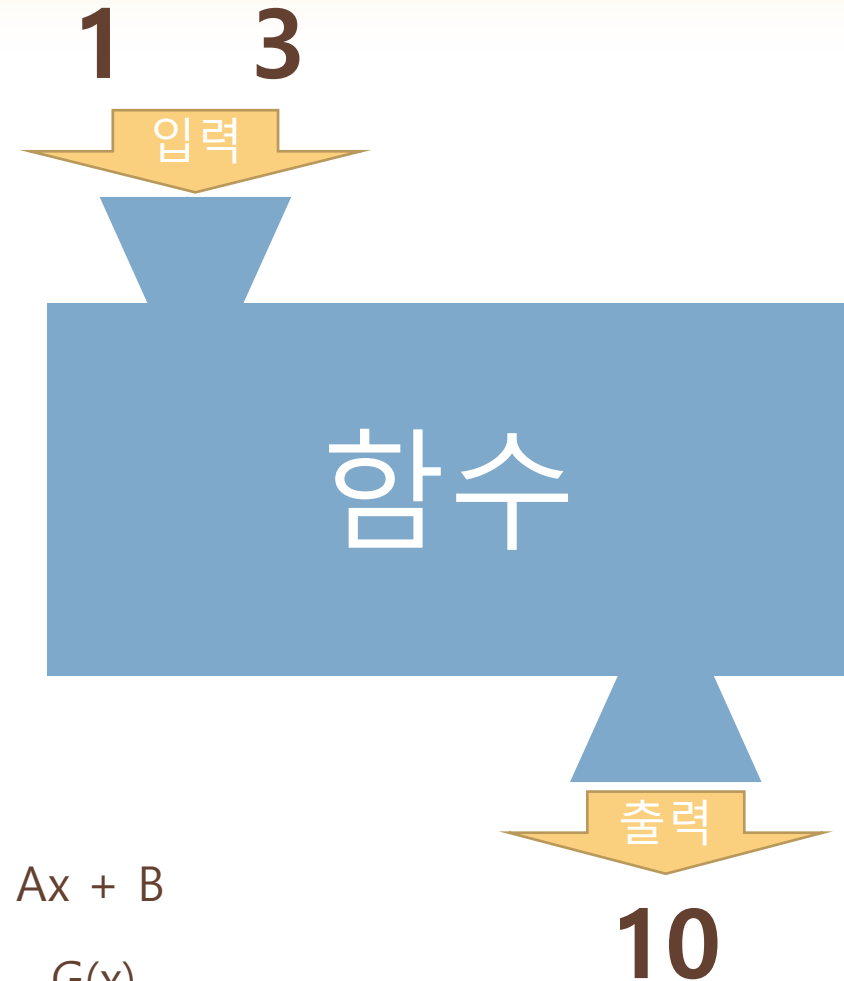
연산자	의미	예시	연산 결과
and	양쪽의 값이 모두 true인 경우 true	(2<4) and (5>8)	false
or	어느 한 쪽만 true인 경우 true	(2<4) or (5<8)	true
not	True면 false, false면 true	not(4<20)	false

• 함수

- 입력을 받아 출력을 내보내는 것
- 불필요한 코드의 반복을 삭제해 주는 편리한 도구
- `def 함수이름(입력1, 입력2, ...):`

• 함수의 3요소

- 함수 이름
- 입력
- 출력 -> 없을 수도 있다



$$y = Ax + B$$

$$F(x) \quad G(x)$$

• adder() 함수 만들기

```
def adder(a, b):  
    return a+b  
##여기까지 함수 정의
```

```
print(adder(1, 1)) ##호출  
print(adder(2, 3)) ##호출  
print(adder(adder(1, 1), adder(2, 3))) ##호출
```

def = 함수 정의 시작
adder = 함수 이름
a = 입력1
b = 입력2
return = 출력

```
test.py x  
1 def adder(a, b):  
2     return a+b  
3  
4  
5 print(adder(1, 1))  
6 print(adder(2, 3))  
7 print(adder(adder(1, 1), adder(2, 3)))  
8  
Run: test x  
C:\Users\wbsal\PycharmProjects\untitled\ver  
2  
5  
7  
Process finished with exit code 0
```

• wait() 함수 만들기 1

```
from time import time
```

```
current_time = time()  
print(current_time)
```

```
while True:  
    if time() - current_time > 5:  
        break
```

```
print(time())
```

```
test.py x  
1 import time  
2  
3  
4 t = time  
5 current_time = t.time()  
6 print(current_time)  
7  
8 while True:  
9     if t.time() - current_time > 5:  
10        break  
11  
12 print(t.time())  
13
```

Run: test x

```
C:\Users\#dbsa1\#PycharmProjects\#untitled\#venv\#Scr  
1530714234.59792  
1530714239.598307  
Process finished with exit code 0
```

• wait() 함수 만들기 2

```
from time import time
```

```
def wait(second: int):
    current_time = time()
    print(current_time)
```

```
while True:
    if time() - current_time > second:
        break
```

```
print(time())
##여기까지 함수 정의
```

```
wait(5) ##함수 호출
```

def = 함수 정의 시작
wait = 함수 이름
second = 입력

```
test.py x
1 import time
2
3
4 def wait(second):
5     t = time
6     current_time = t.time()
7     print(current_time)
8
9     while True:
10        if t.time() - current_time > second:
11            break
12
13        print(t.time())
14
15
16 wait(5)
```

Run: test x

```
C:\Users\#dbsa.l#\PycharmProjects#\untitled#\env#\Scripts
1530714415.4109905
1530714420.4115455
```

- 프로그래밍이란?

- 한 편의 연극을 만드는 것.
- 메모리 = 무대
- 코드 = 대본

- 배우가 소품들을 가지고 연기를 한다.
- 객체가 변수들을 가지고 함수로 동작한다.

• 대본을 잘 쓰려면?

- 시간의 흐름에 따라 논리적으로
- 언제? 어디서? 누가? 무엇을?
어떻게? 왜?
- 장면 하나하나를 구체적으로 상상
- 배우의 특징을 고려

• 프로그래밍을 잘 작성하려면?

- 순차 진행에 따라 논리적으로
- 언어 문법에 알맞게
- 장면 하나하나를 구체적으로 상상
- 적절한 변수, 함수, 객체의 사용

로봇 제어 기초

필수

• 로봇 프로그래밍

- 프로그램 작성 방법에서 실행해 보았던 코드에 사용된 함수에 대해 알아본다.
- 기본 설정
 - `#!/usr/bin/env python3` : EV3에서 프로그램을 실행 시키기 위해서는 반드시 제일 첫 행에 입력해야한다.
 - `from ev3dev.ev3 import *` : EV3를 사용하기 위한 모든 public 속성/함수를 사용하기 위해 반드시 필요하다. 모든 프로그램의 첫 행에 입력한다.

```
#!/usr/bin/env python3  
from ev3dev.ev3 import *
```

Sound.beep()

로봇 제어 기초

모터 제어

• 모터 제어

- 모터 객체를 생성하고 객체 안의 함수를 이용하여 제어한다.

• 모터 제어 함수

- LargeMotor(출력 포트) - 모터 객체
- run_forever(speed = 0~1000) - 모터를 특정 속도로 회전 시키는 함수
- stop(stop_action = 'hold') - 모터를 정지 시키는 함수. hold, coast, break가 있다.

- 모터 하나를 5초 동안 회전 시키기

```
#!/usr/bin/env python3
from ev3dev.ev3 import *
from time import time

motorB = LargeMotor('outB')
motorB.run_forever(speed_sp=300)
current_time = time()

while True:
    if time() - current_time > 5:
        break

motorB.stop(stop_action="hold")
```

- 전진 5초 하기

```
#!/usr/bin/env python3  
from ev3dev.ev3 import *  
from time import time
```

삽입

```
motorB = LargeMotor('outB')  
motorC = LargeMotor('outC')  
motorB.run_forever(speed_sp=300)  
motorC.run_forever(speed_sp=300)
```

```
wait(second=5)
```

```
motorB.stop(stop_action="hold")  
motorC.stop(stop_action="hold")
```

```
def wait(second: int):  
    current_time = time()  
    while True:  
        if time() - current_time > second:  
            break
```

- 모터의 회전 각도를 이용하기
 - 모터에 내장된 센서의 값을 이용
 - 우선 누적되어 있는 회전 각도를 0으로 초기화
 - +방향 회전과 -방향 회전을 구분

```
motorB = LargeMotor('outB')
motorB.position = 0
motorB.run_forever(speed_sp = 300)
```

```
while True:
    if motorB.position >= 360:
        break
```

```
motorB.stop(stop_action = "hold")
```

```
motorB = LargeMotor('outB')
motorB.position = 0
motorB.run_forever(speed_sp = -300)
```

```
while True:
    if motorB.position <= -360:
        break
```

```
motorB.stop(stop_action = "hold")
```

• 전진 360도

```
motorB = LargeMotor('outB')
motorC = LargeMotor('outC')
motorB.position = 0
motorB.run_forever(speed_sp = 300)
motorC.run_forever(speed_sp = 300)
```

```
while True:
    if motorB.position >= 360:
        break
```

```
motorB.stop(stop_action = "hold")
motorC.stop(stop_action = "hold")
```

• 후진 360도

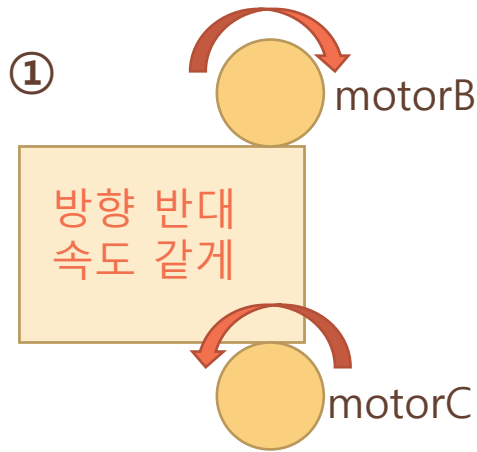
```
motorB = LargeMotor('outB')
motorC = LargeMotor('outC')
motorB.position = 0
motorB.run_forever(speed_sp = -300)
motorC.run_forever(speed_sp = -300)
```

```
while True:
    if motorB.position <= -360:
        break
```

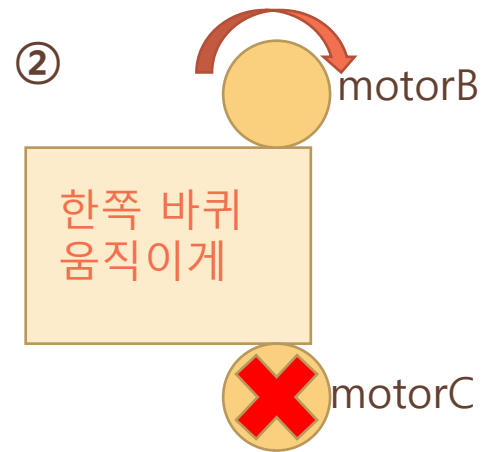
```
motorB.stop(stop_action = "hold")
motorC.stop(stop_action = "hold")
```


• 회전하기

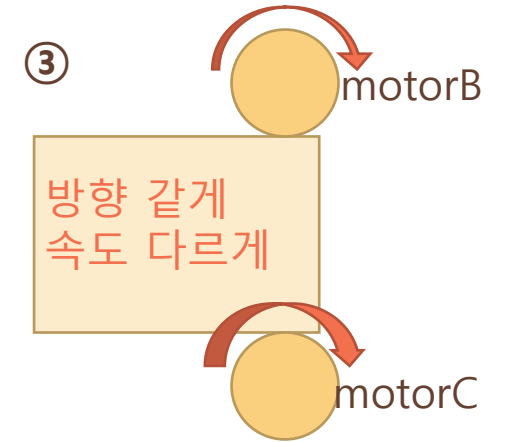
- 로봇을 회전시키는 방법



<point turn>



<swing turn>



<curve turn>

• point turn

```
motorB = LargeMotor('outB')
motorC = LargeMotor('outC')
motorB.position = 0
motorB.run_forever(speed_sp = 300)
motorC.run_forever(speed_sp = -300)
```

```
while True:
    if motorB.position >= 360:
        break
```

```
motorB.stop(stop_action = "hold")
motorC.stop(stop_action = "hold")
```

• swing turn

```
motorB = LargeMotor('outB')
motorC = LargeMotor('outC')
motorB.position = 0
motorB.run_forever(speed_sp = 300)
motorC.run_forever(speed_sp = 0)
```

```
while True:
    if motorB.position >= 360:
        break
```

```
motorB.stop(stop_action = "hold")
motorC.stop(stop_action = "hold")
```

• curve turn

```
motorB = LargeMotor('outB')
motorC = LargeMotor('outC')
motorB.position = 0
motorB.run_forever(speed_sp = 300)
motorC.run_forever(speed_sp = 100)
```

```
while True:
    if motorB.position >= 360:
        break
```

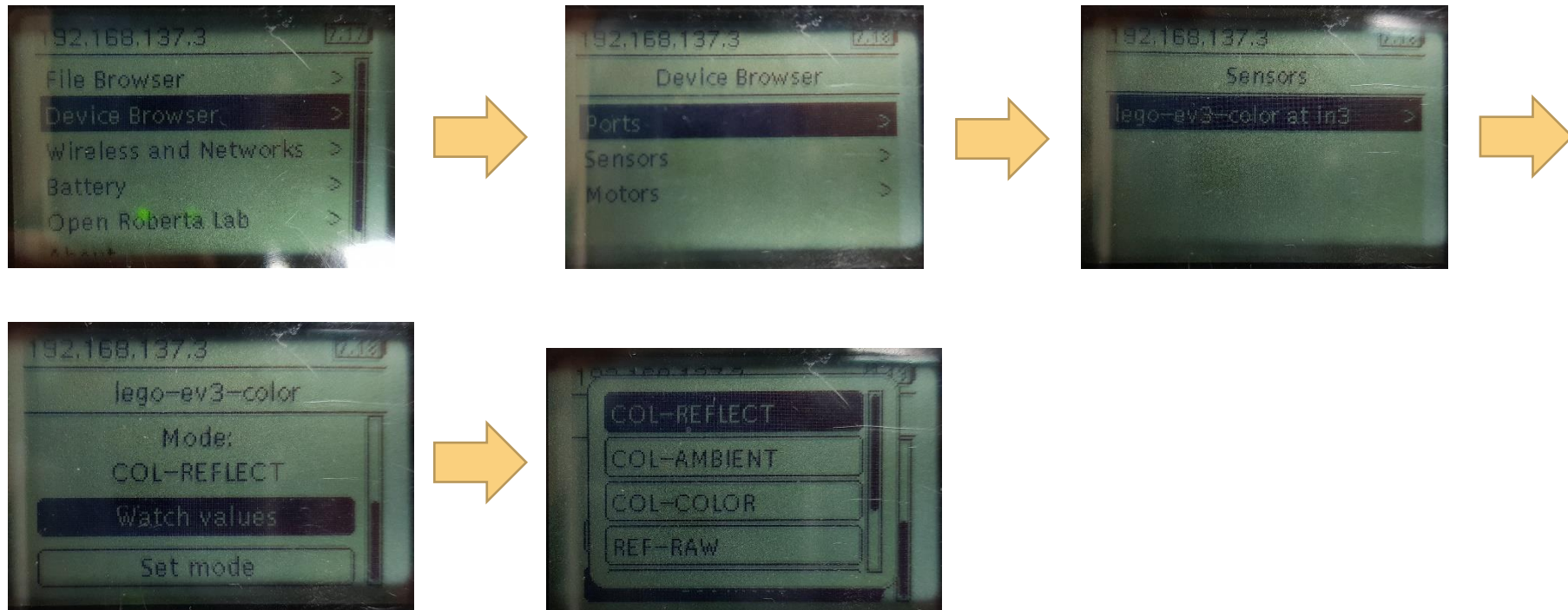
```
motorB.stop(stop_action = "hold")
motorC.stop(stop_action = "hold")
```

로봇 제어 기초

컬러 센서

• 컬러 센서 활용

- 컬러 센서로 색상의 값을 측정하는 방법.



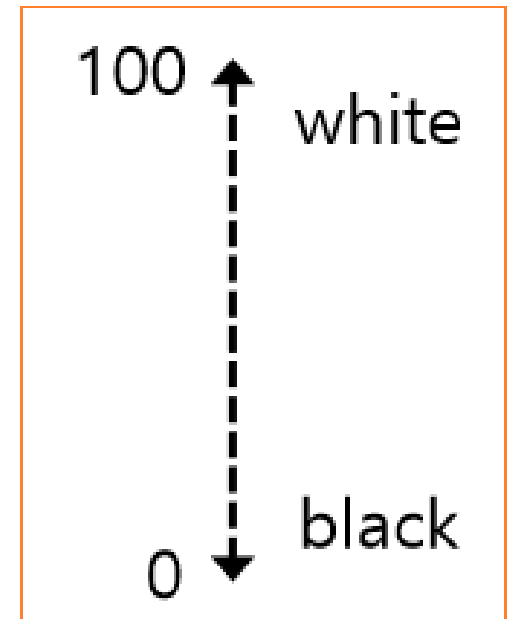
• 컬러 모드(COL-COLOR)

- 컬러 센서가 감지하는 표면의 색상에 따라 다른 숫자 값이 반환된다.

숫자	색상	숫자	색상	숫자	색상	숫자	색상
0	None	1	Black	2	Blue	3	Green
4	Yellow	5	Red	6	White	7	Brown

• 반사광 모드(COL-REFLECT)

- 컬러 센서를 반사광 모드로 작동시키면, 센서는 현재 측정하고 있는 표면이 빛에 반사되는 정도를 감지하여 측정값을 0~100사이의 값으로 반환한다.
- 어두운 영역과 밝은 영역을 구분할 때는 두 영역의 경계로 판단할 수 있는 **문턱값(threshold)**을 정해야 한다.
 - 가장 일반적으로 문턱값을 정할 때는 두 표면의 빛 값을 측정한 후 평균값을 계산하는 방법을 활용한다.



• 컬러 센서

- 컬러 센서 객체를 생성하고 객체 안의 함수를 이용하여 제어한다.

• 센서 값 제어

- ColorSensor(입력 포트) – 컬러센서 객체를 생성
- mode = 센서 모드 – 센서 모드를 지정,
 - COL-REFLECT, COL-COLOR, COL-AMBIENT
- value() – 센서가 감지하고 있는 값

- 컬러센서가 빨간색을 인식 할 때 까지 전진 후 정지

```
motorB = LargeMotor('outB')
motorC = LargeMotor('outC')
color3 = ColorSensor('in3')
color3.mode = 'COL-COLOR'
motorB.run_forever(speed_sp = 300)
motorC.run_forever(speed_sp = 300)
```

```
while True:
    if color3.value() == 5:
        break
```

```
motorB.stop(stop_action = "hold")
motorC.stop(stop_action = "hold")
```


- 밝으면 빠르게 어두우면 느리게

```
motorB = LargeMotor('outB')  
motorC = LargeMotor('outC')
```

```
color3 = ColorSensor('in3')  
color3.mode = 'COL-REFLECT'
```

```
current_time = time()
```

```
while time() - current_time < 5:  
    motorB.run_forever(speed_sp = color3.value()*100)  
    motorC.run_forever(speed_sp = color3.value()*100)
```

```
motorB.stop(stop_action = "hold")  
motorC.stop(stop_action = "hold")
```

로봇 제어 응용

•정확한 회전 연습

- 모터의 회전 각을 이용
- 포인트 턴과 스윙 턴으로 90도 180도 360도 정확하게 회전하기
- 포인트 턴과 스윙 턴 각각의 특징과 차이점을 확인
- 포인트 턴의 회전 각도 * 2 = 스윙 턴의 회전 각도

• 사각형을 그리며 움직이고 제자리로 돌아오기

1. 직진 - 회전 - 직진 - 회전 - 직진 - 회전 - 직진 - 회전
2. 직진과 회전 사이의 '대기시간'을 넣어 관성 제거하기
3. 반복문을 사용하여 코드의 반복 줄이기

- 사각형 돌기

```
for count in range(0,4):
    motorB.run_forever(speed_sp=300)
    motorC.run_forever(speed_sp=300)
    wait(2)
    motorB.stop(stop_action="hold")
    motorC.stop(stop_action="hold")

    wait(0.5)

    turn_right(360)

    wait(0.5)
```

```
def wait(second: int):
    current_time = time()
    while True:
        if time() - current_time > second:
            break
```

```
def turn_right(angle: int):
    motorB.run_forever(speed_sp=300)
    motorC.run_forever(speed_sp=-300)
    motorB.position = 0
    while True:
        if motorB.position >= angle:
            break
```

```
motorB.stop(stop_action="hold")
motorC.stop(stop_action="hold")
```

• N번 째 검은 선에서 정지하기

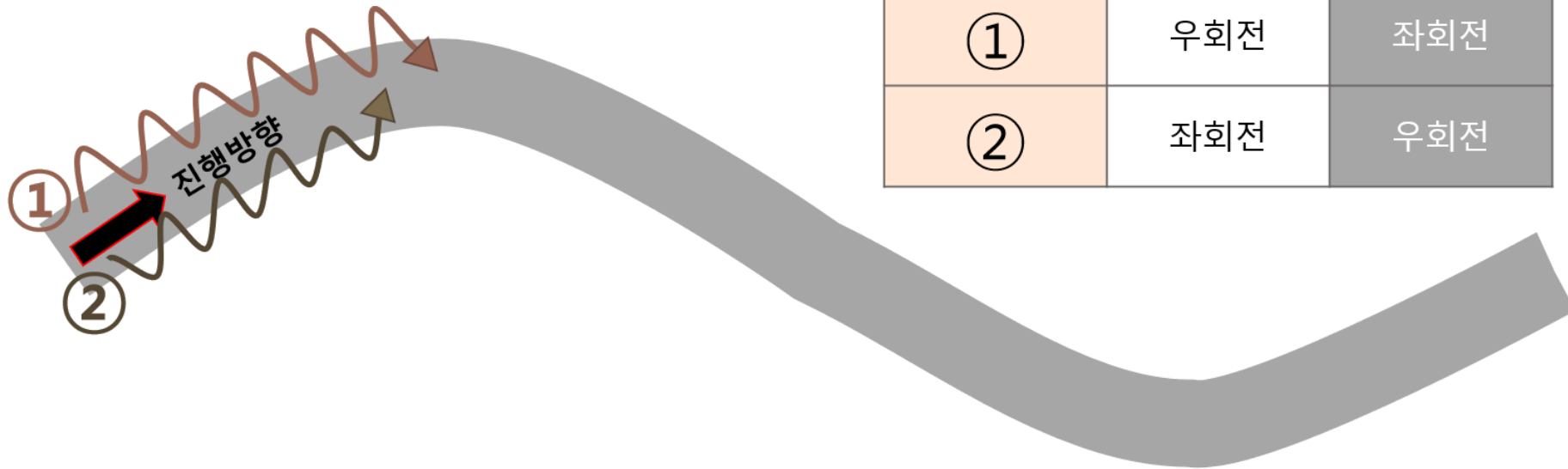
- 컬러 센서를 이용하여 검은 선의 개수를 세며 전진하기
- 센서가 검은 선을 통과하는 시간(또는 모터 회전 각도)가 필요
- 반복문을 사용하여 코드의 반복을 제거

- 4번 째 검은선에서 멈추기

```
for count in range(0,4):  
    motorB.run_forever(speed_sp=300)  
    motorC.run_forever(speed_sp=300)  
  
    while True:  
        if color3.value() < 15:  
            break  
  
        Sound.beep()  
        wait(0.5)  
  
motorB.stop(stop_action="hold")  
motorC.stop(stop_action="hold")
```

• 라인트레이싱

- 컬러 센서를 활용한 라인트레이싱



- 10 초 동안 라인트레이싱

- 30 = 문턱 값
- (흰색의 value + 검은색의 value) / 2

```
motorB = LargeMotor('outB')  
motorC = LargeMotor('outC')
```

```
color3 = ColorSensor('in3')  
color3.mode = 'COL-REFLECT'
```

```
current_time = time()
```

```
while time() - current_time < 10:  
    if color3.value() < 30:  
        motorB.run_forever(speed_sp = 0)  
        motorC.run_forever(speed_sp = 300)  
    else :  
        motorB.run_forever(speed_sp = 300)  
        motorC.run_forever(speed_sp = 0)
```

```
motorB.stop(stop_action = "hold")  
motorC.stop(stop_action = "hold")
```

• 참고 사이트

- EV3 Python 학습 사이트 : www.ev3python.com
- API 참조 : <http://python-ev3dev.readthedocs.io/en/latest/spec.html>
- Visual studio code 설치 관련: <https://youtu.be/cqtRqsl6xMc>
- EV3와 PC의 연결 : <https://youtu.be/TNXqizQTZhs>

• 교육, 기술 지원 및 경기장 구매 문의

퓨너스 (www.funers.com) T.070-8670-8911