

Python 함수와 클래스, 모듈화 이해하기

변영철 교수
(ycb@jejunu.ac.kr)

“

오, 너 코딩 **잘한다!**

프로그램을 잘 짠다는 것은?

클래스를 이용한, 모듈화,
모듈을 플러그 인할 수 있는 응용 프레임워크

1절. 환경 설정

Python 설치하기

구글에서 'Python 다운로드'
검색하여 설치

1절. 환경 설정

PyCharm 설치하기

구글에서 ‘PyCharm 다운로드’
검색하여 설치



JetBrains

<https://www.jetbrains.com/ko-kr/pycharm/download> ⋮

PyCharm 다운로드: JetBrains가 만든 전문 개발자용 Python ...

Windows, macOS 또는 Linux용 최신 버전의 **PyCharm**을 다운로드하세요. ... **PyCharm** Professional. 전문 개발자용 Python IDE. 다운로드 .exe. 30일 무료 평가판.

US\$8.90 ~ US\$19.90

[기타 버전](#) · [PyCharm 알아보기](#) · [가격 책정](#) · [얼리 액세스 프로그램\(EAP\)](#)



버전: 2021.2.2
빌드: 212.5284.44
2021년 9월 15일

[시스템 요구 사항](#)

[설치 안내](#)

[기타 버전](#)

다운로드 PyCharm

Windows

macOS

Linux

Professional

과학 및 웹 Python 개발용. HTML, JS, SQL 지원.

다운로드

무료 평가판

Community

순수 Python 개발용

다운로드

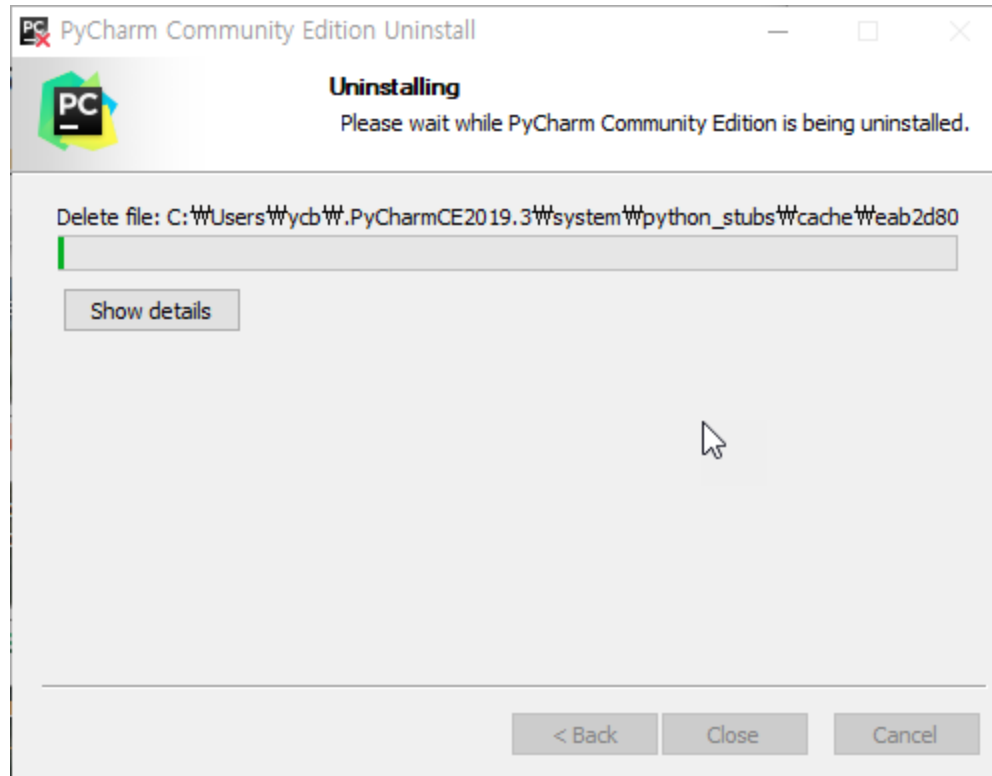
무료, 오픈 소스로 빌드됨

Feedback

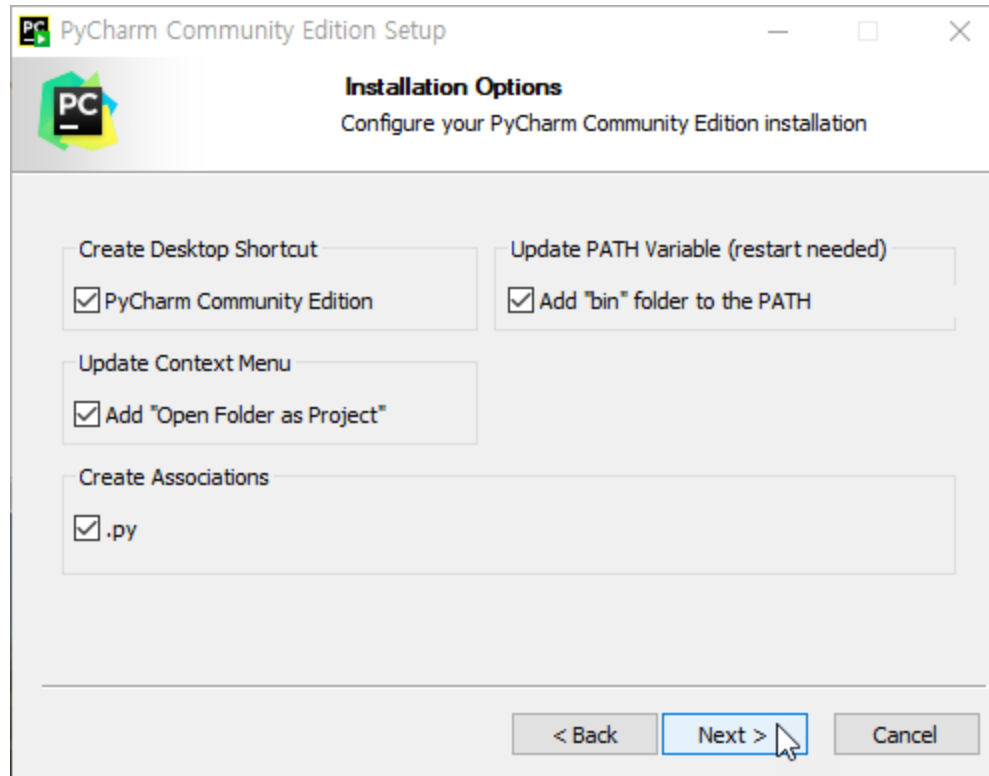


Toolbox App을 설치하여 PyCharm 및 향후

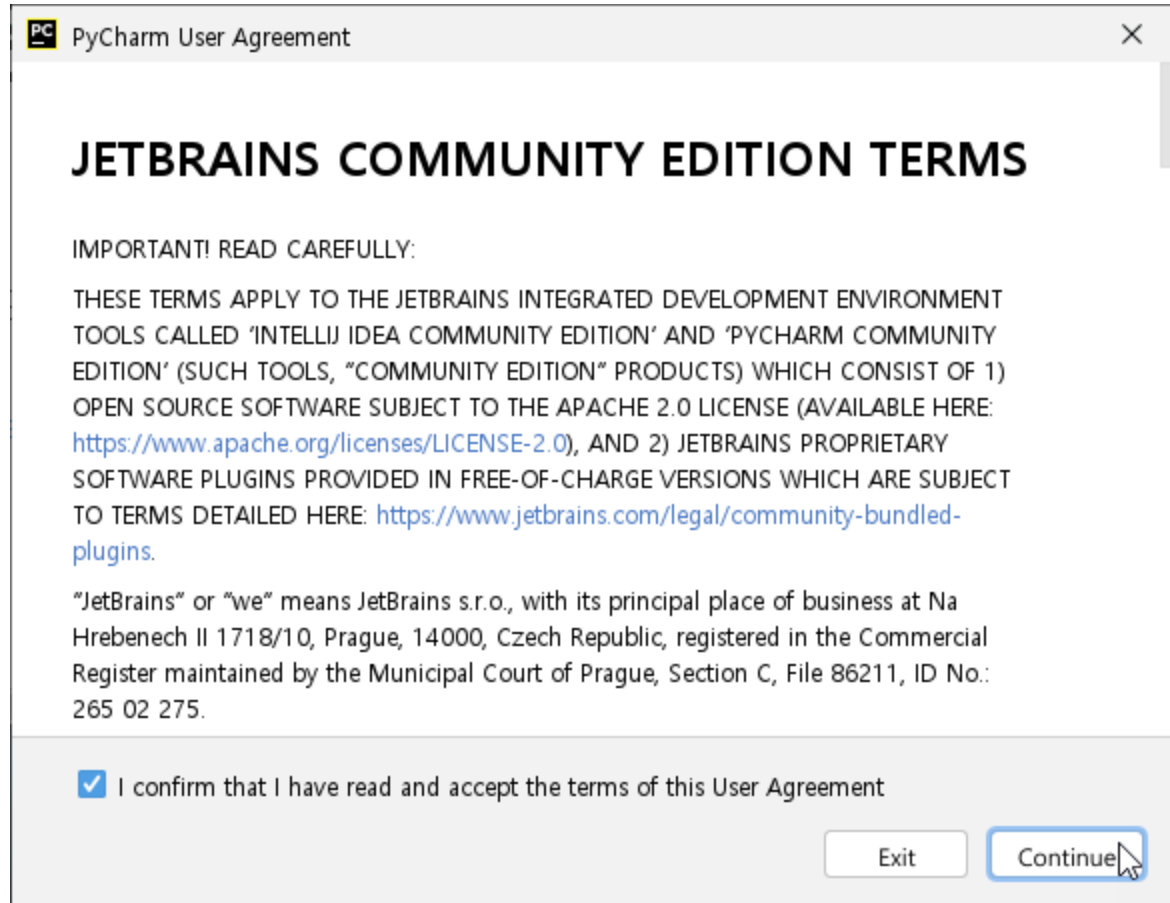
1절. 환경 설정

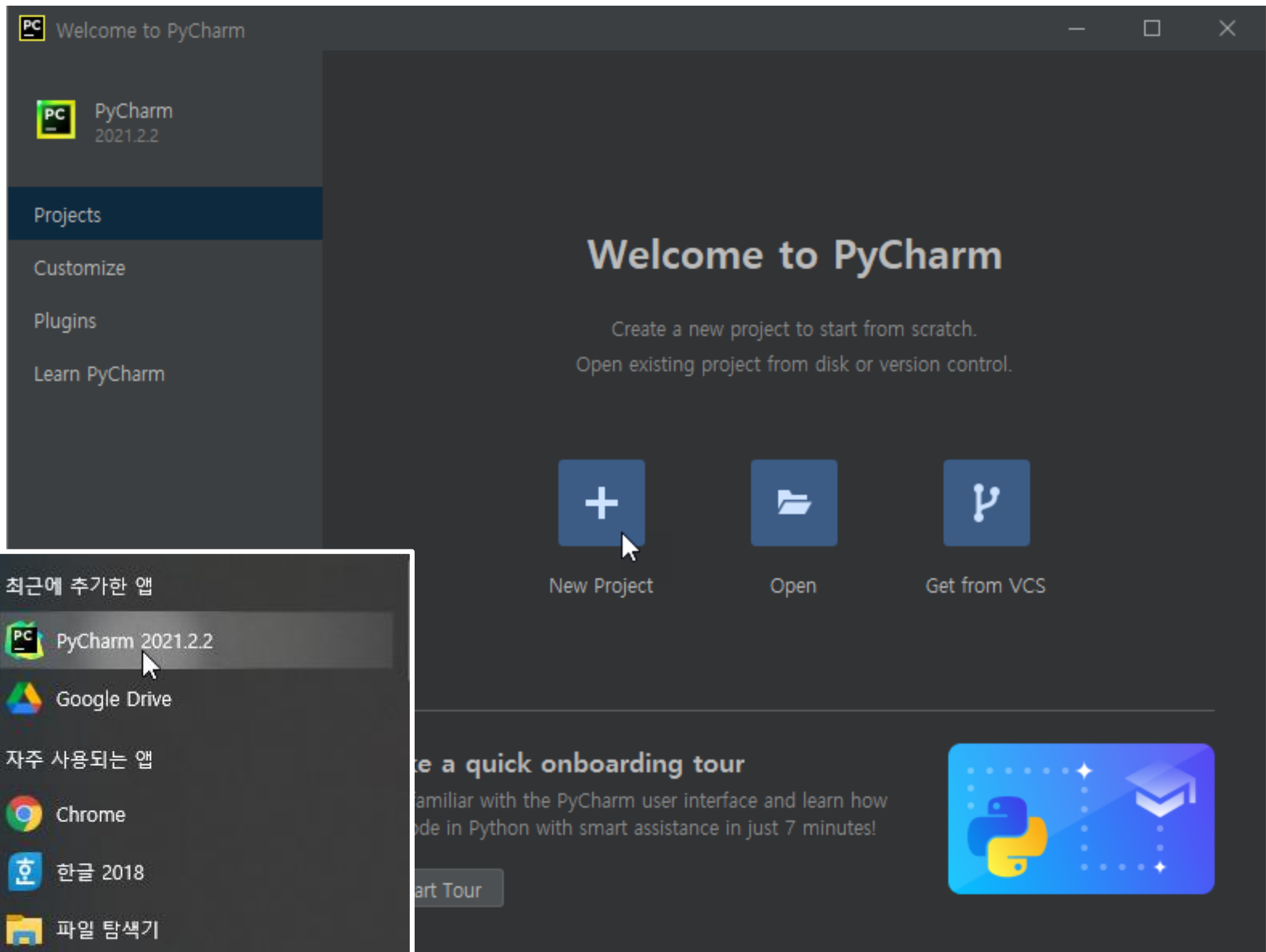


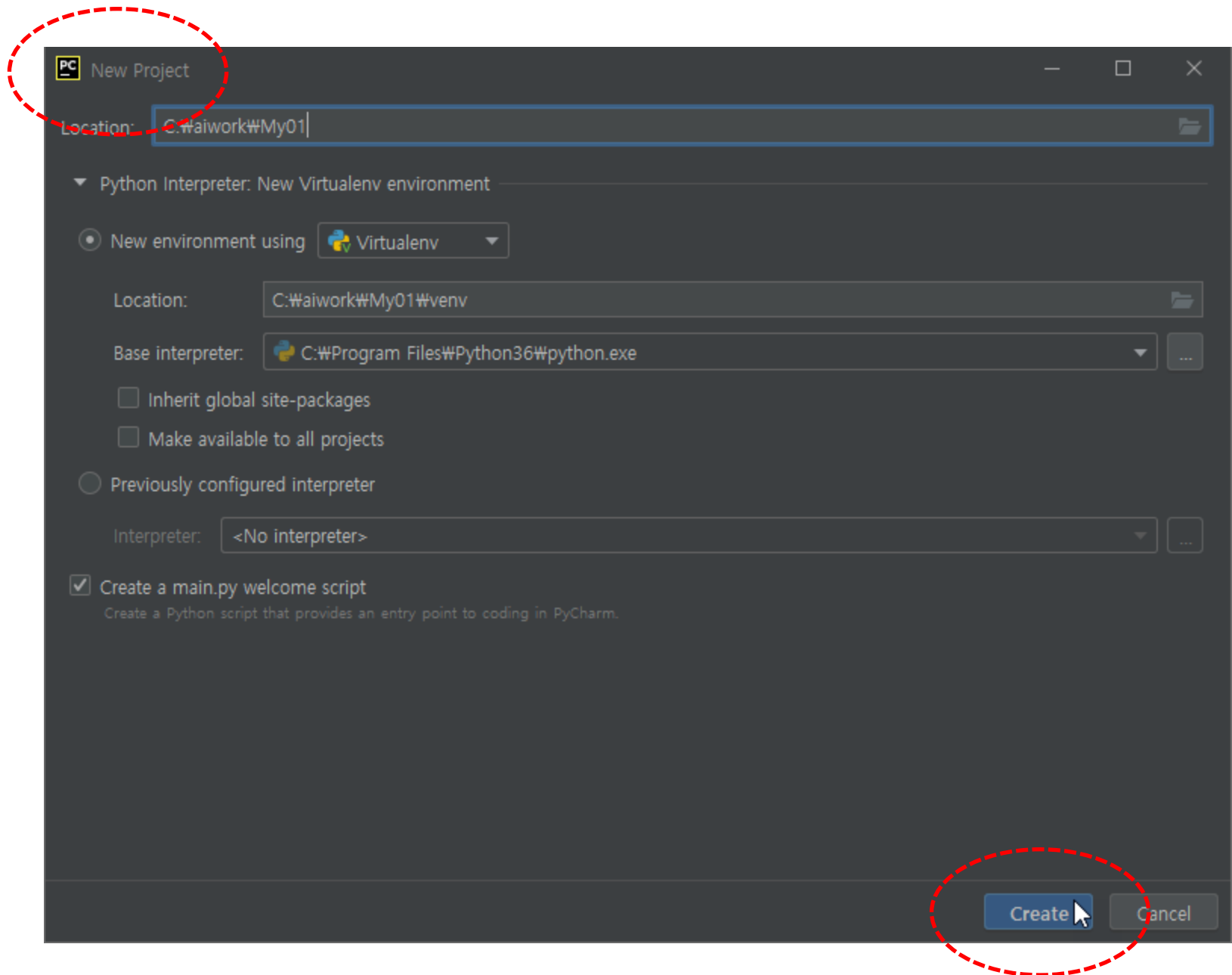
1절. 환경 설정

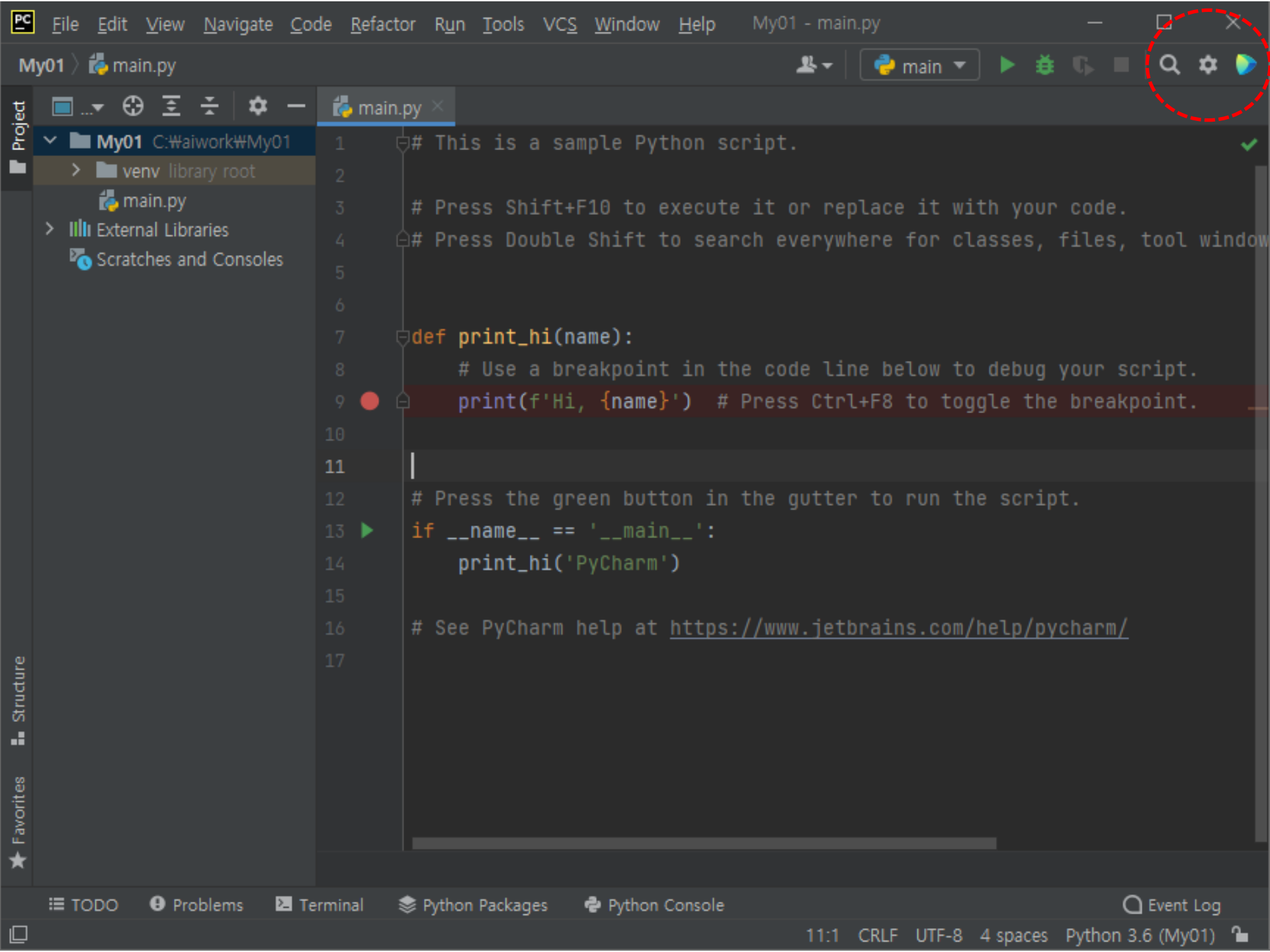


1절. 환경 설정









My01 > main.py

Project

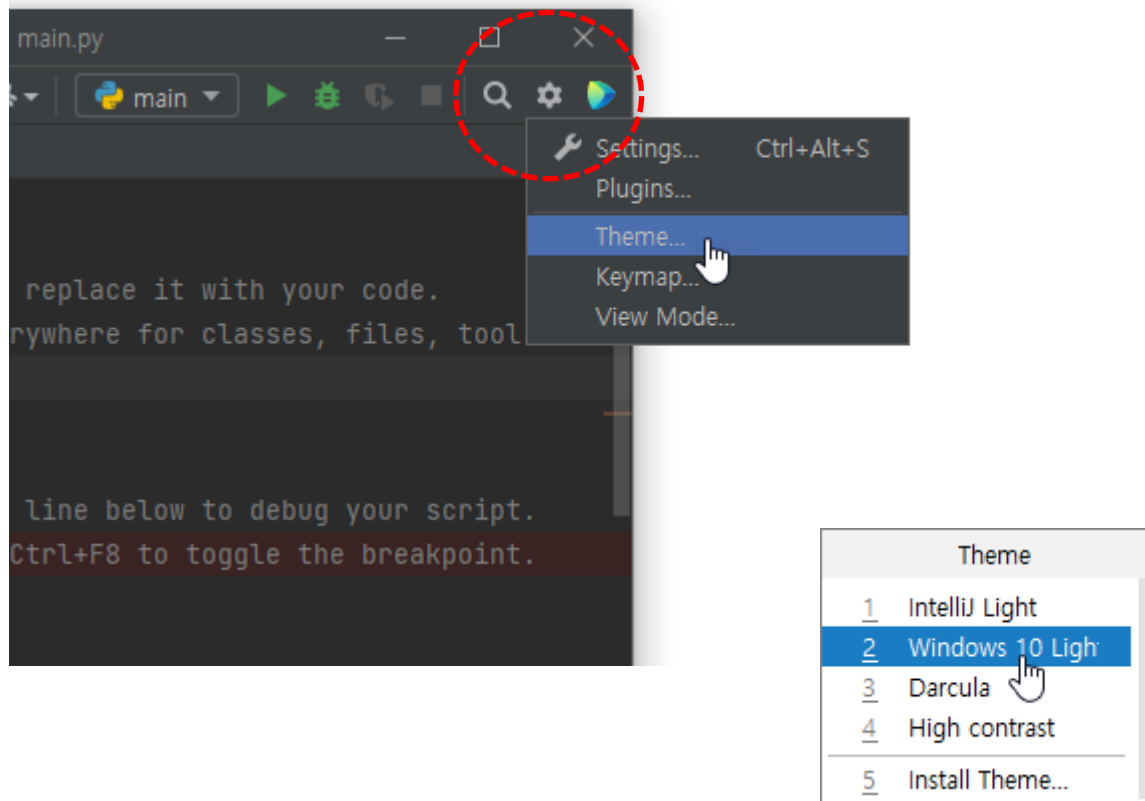
- My01 C:\waiwork\My01
 - venv library root
 - main.py
 - External Libraries
 - Scratches and Consoles

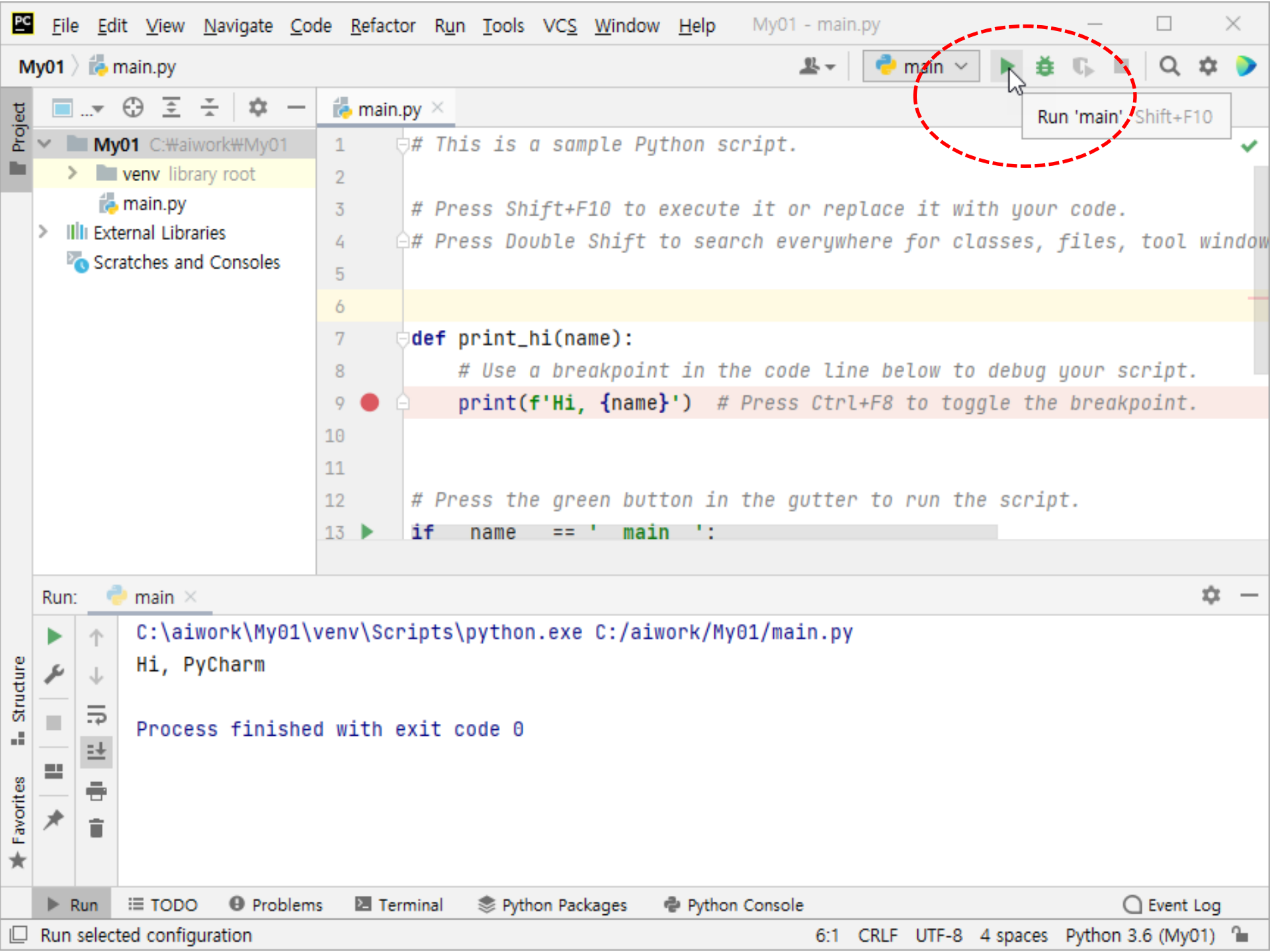
Structure

Favorites

```
1 # This is a sample Python script.
2
3 # Press Shift+F10 to execute it or replace it with your code.
4 # Press Double Shift to search everywhere for classes, files, tool window
5
6
7 def print_hi(name):
8     # Use a breakpoint in the code line below to debug your script.
9     print(f'Hi, {name}') # Press Ctrl+F8 to toggle the breakpoint.
10
11
12 # Press the green button in the gutter to run the script.
13 if __name__ == '__main__':
14     print_hi('PyCharm')
15
16 # See PyCharm help at https://www.jetbrains.com/help/pycharm/
17
```

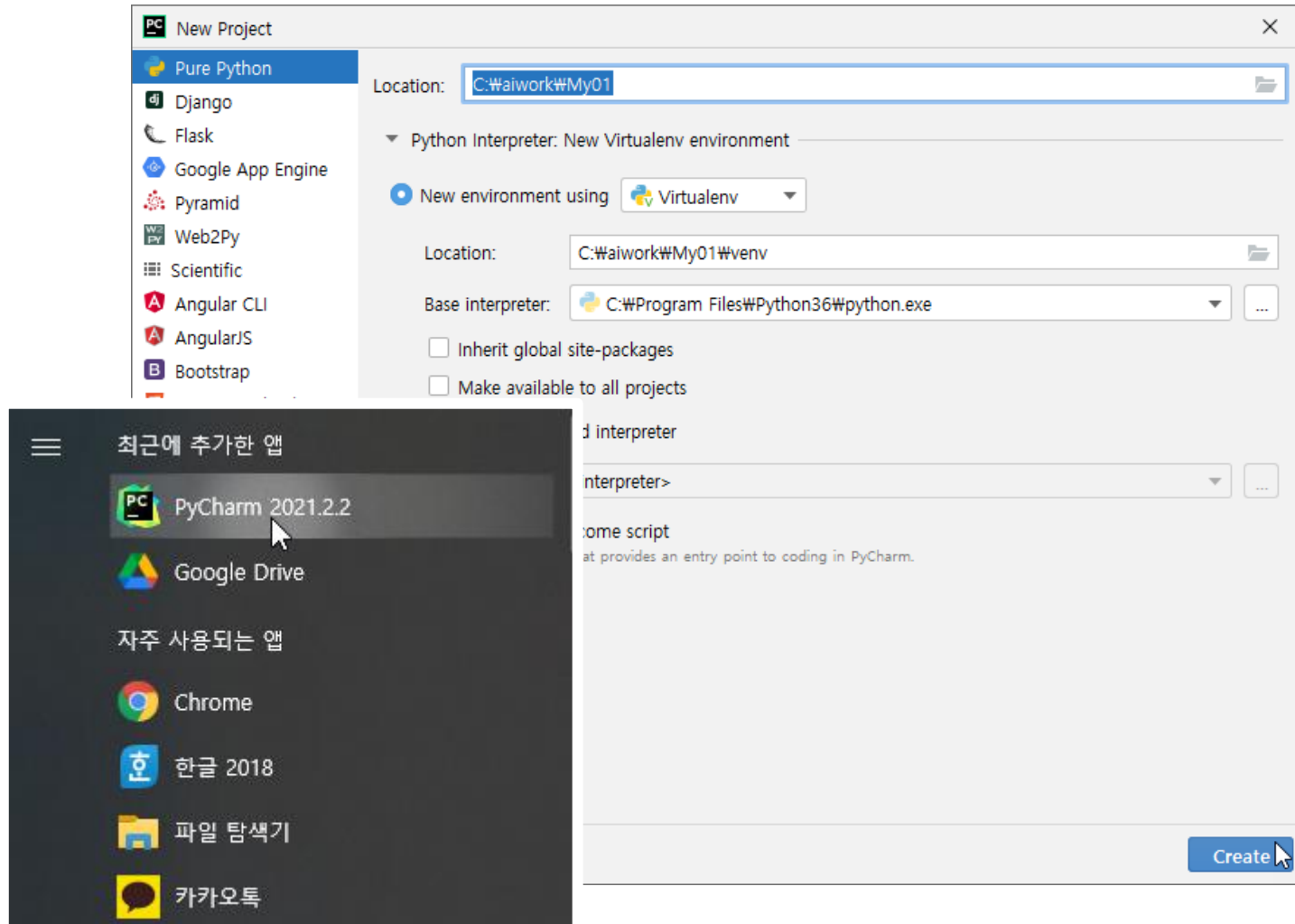
2절. 개발환경 익숙해지기





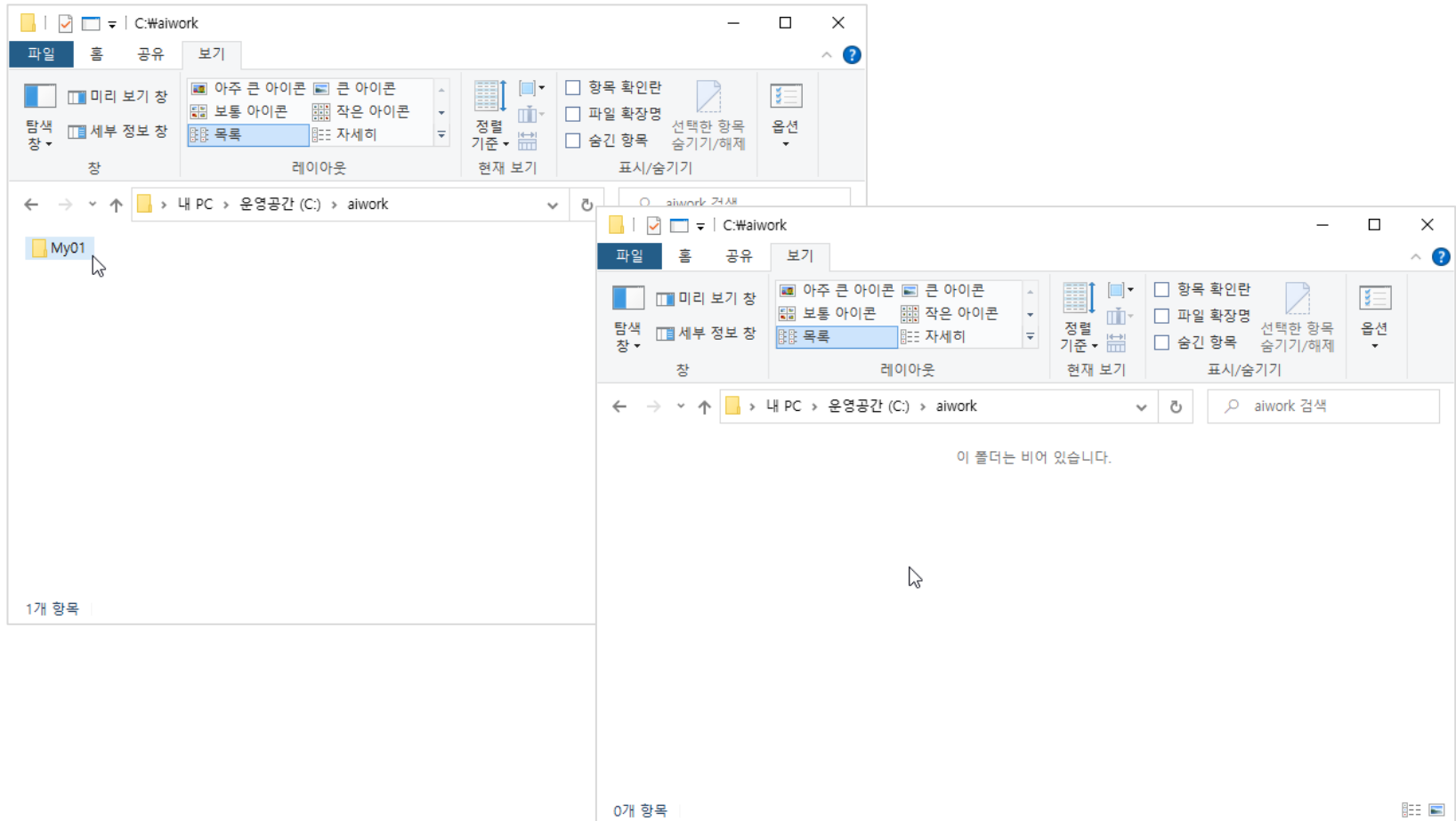
3절. 아주 간단한 Python 프로그램

- File | New Project



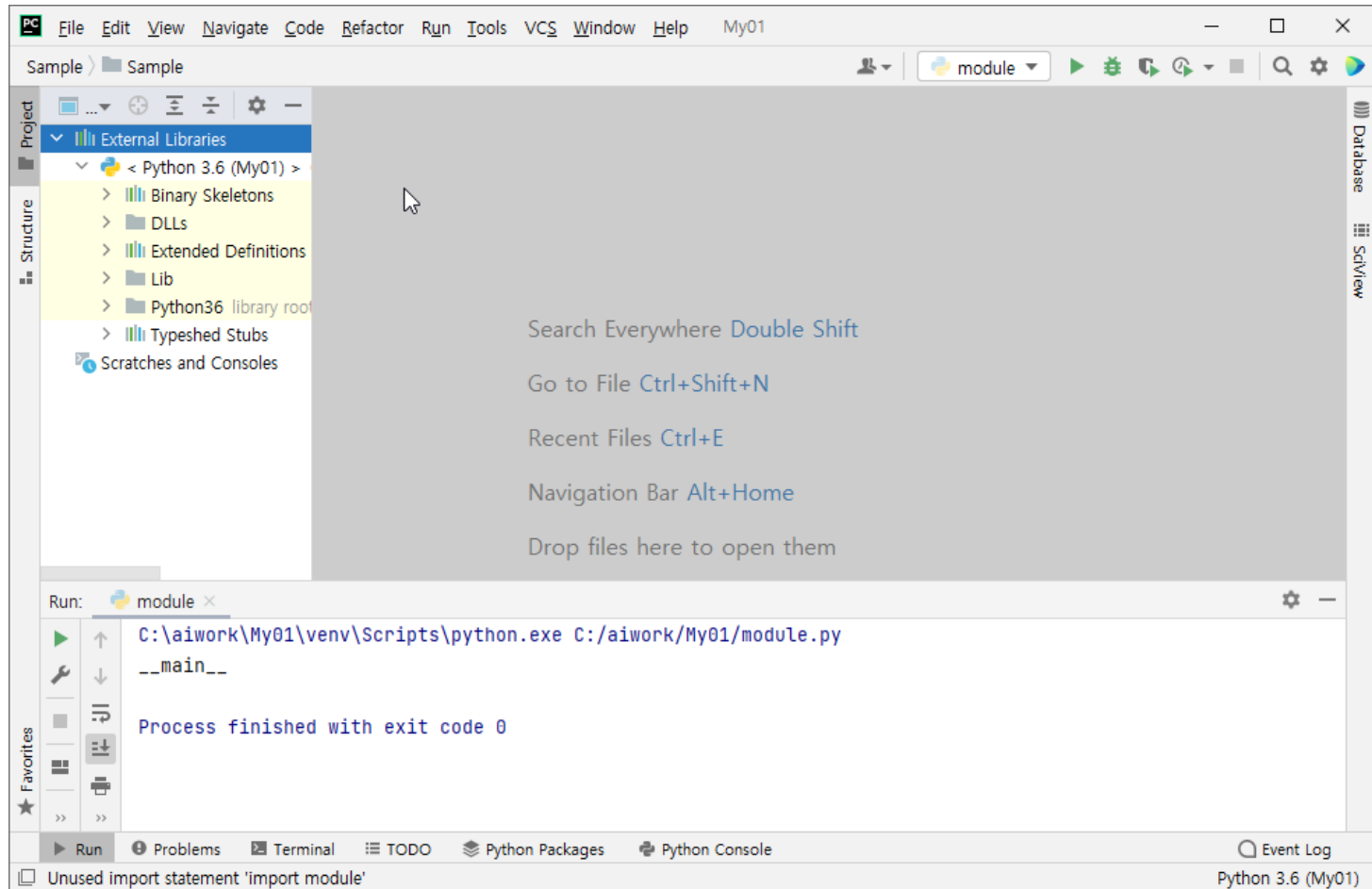
3절. 아주 간단한 Python 프로그램

- 생성한 My01 프로젝트 삭제하기



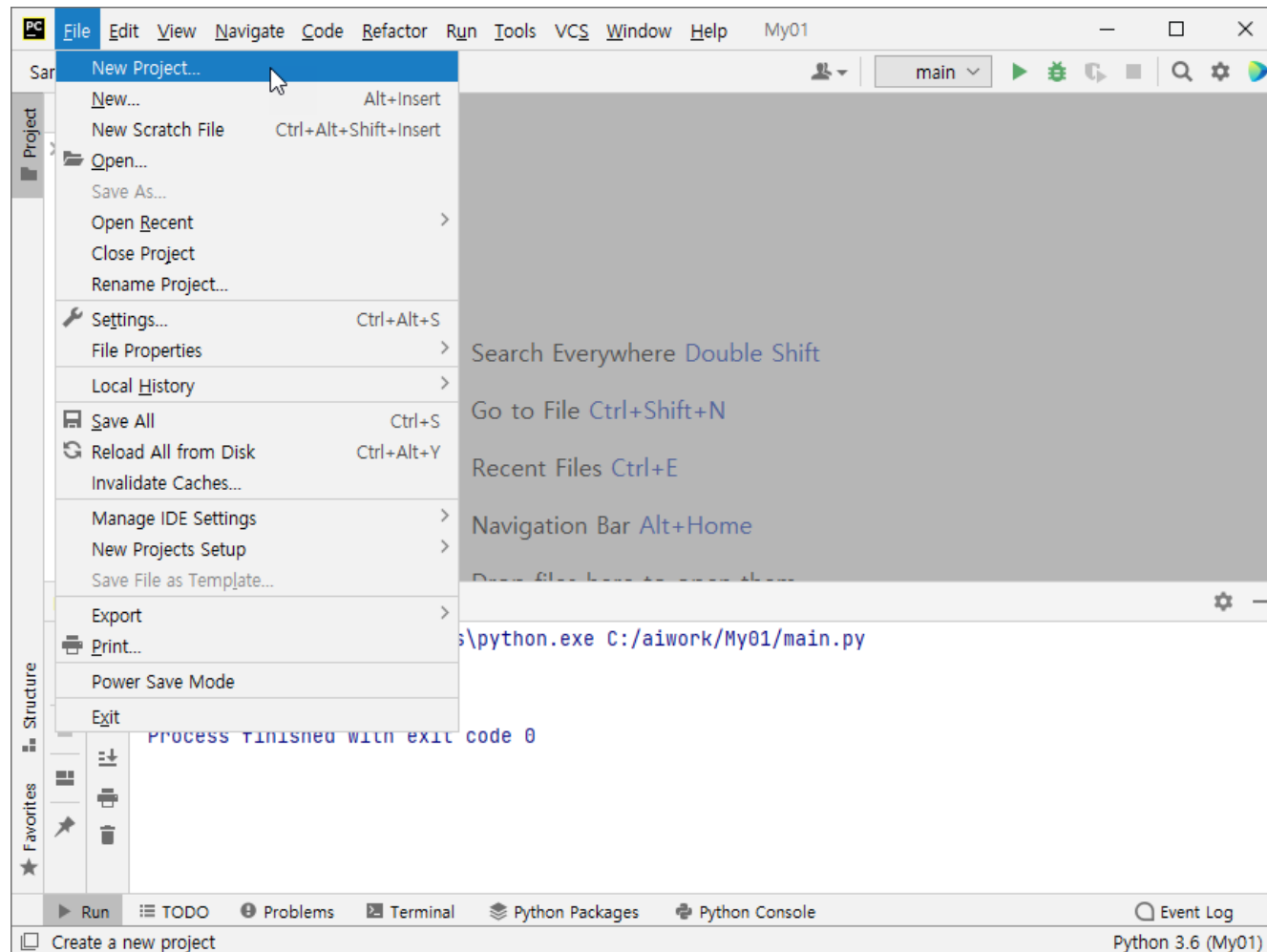
3절. 아주 간단한 Python 프로그램

- 탐색기에서 삭제한 후 PyCharm 모습 (삭제된 모습)



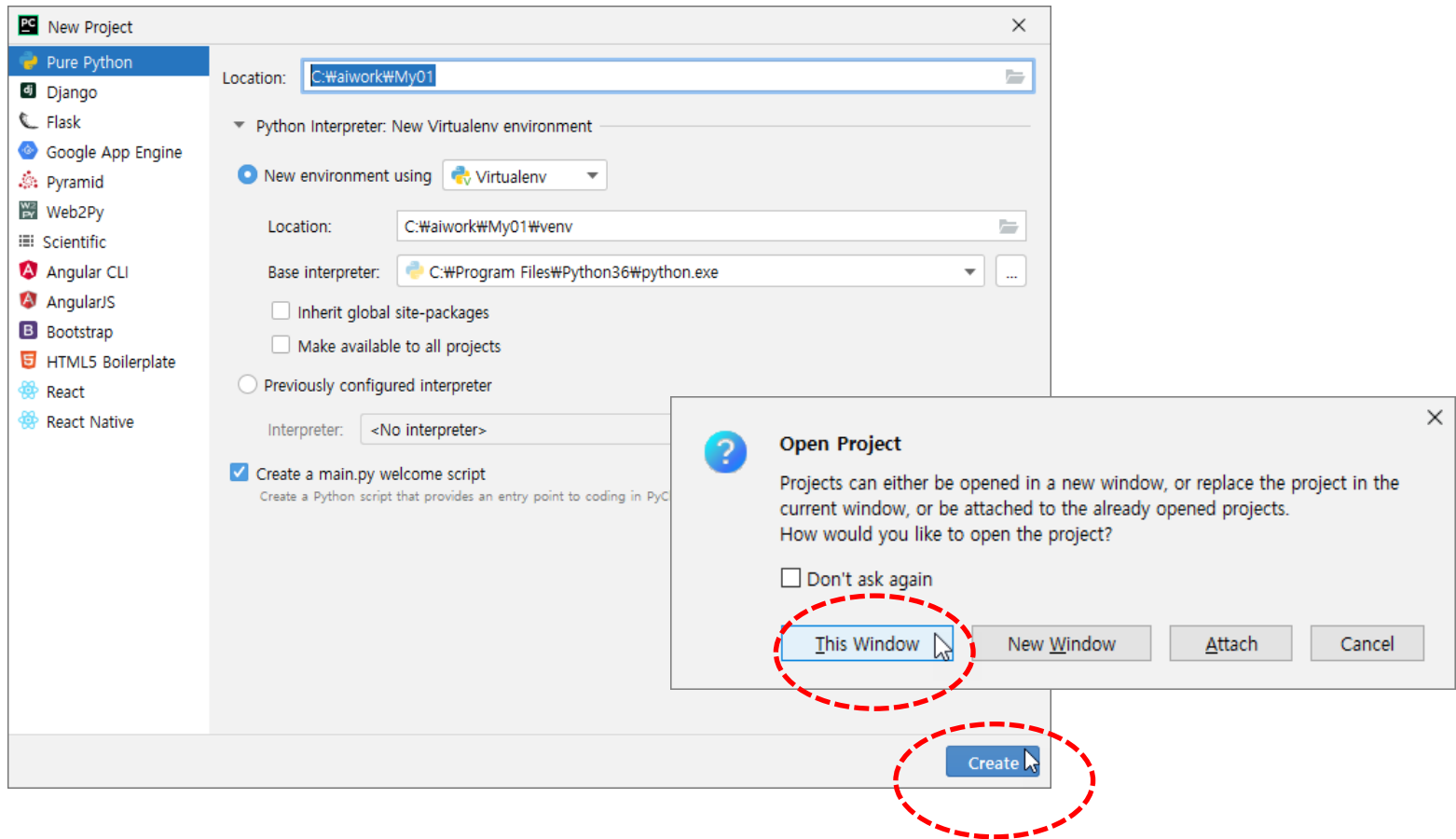
3절. 아주 간단한 Python 프로그램

- 다시 My01 프로젝트 생성하기, File | New Project



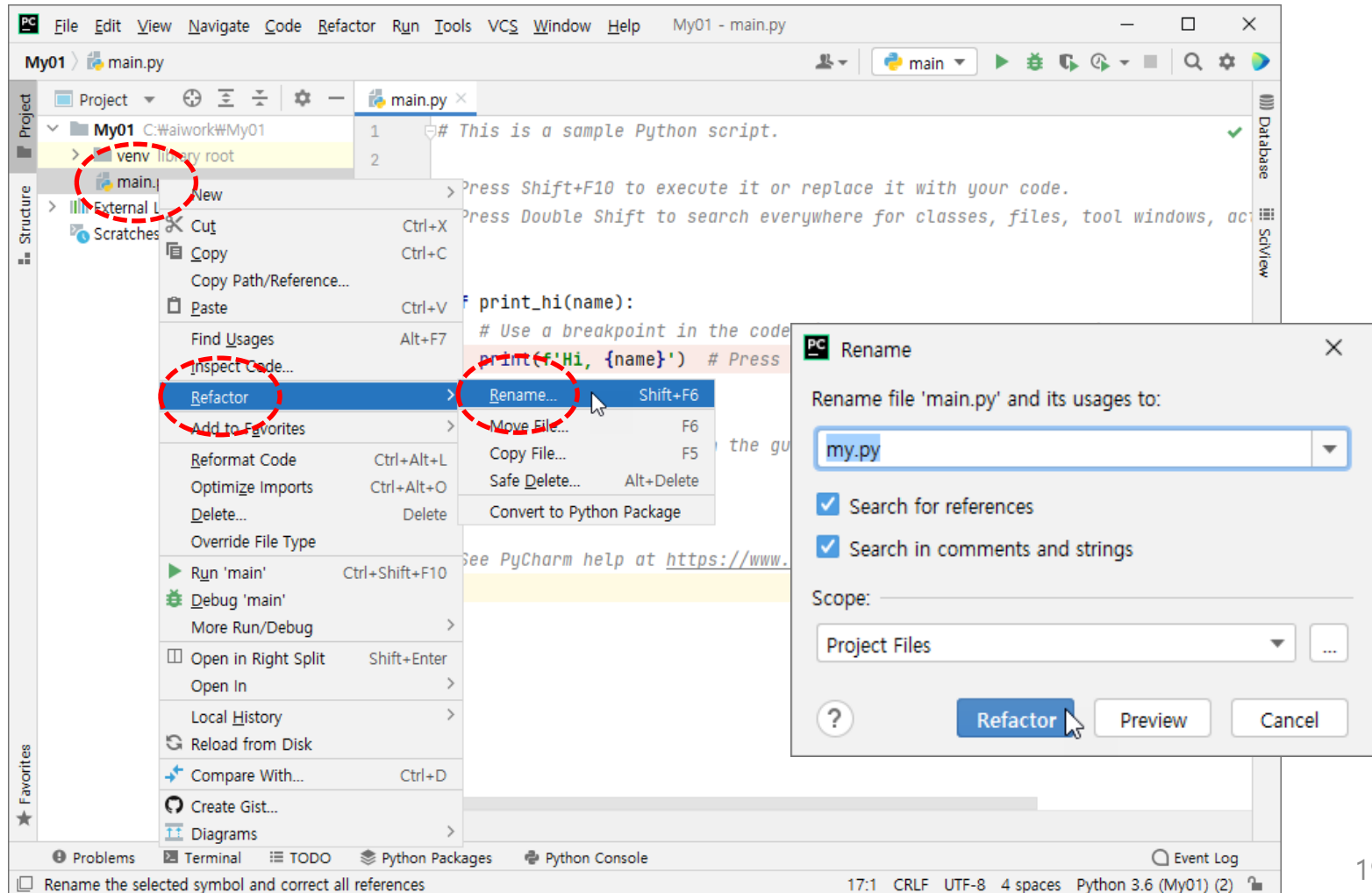
3절. 아주 간단한 Python 프로그램

- 다시 My01 프로젝트 생성하기



3절. 아주 간단한 Python 프로그램

- 파일 이름 바꾸기 (main.py → my.py)



3절. 아주 간단한 Python 프로그램

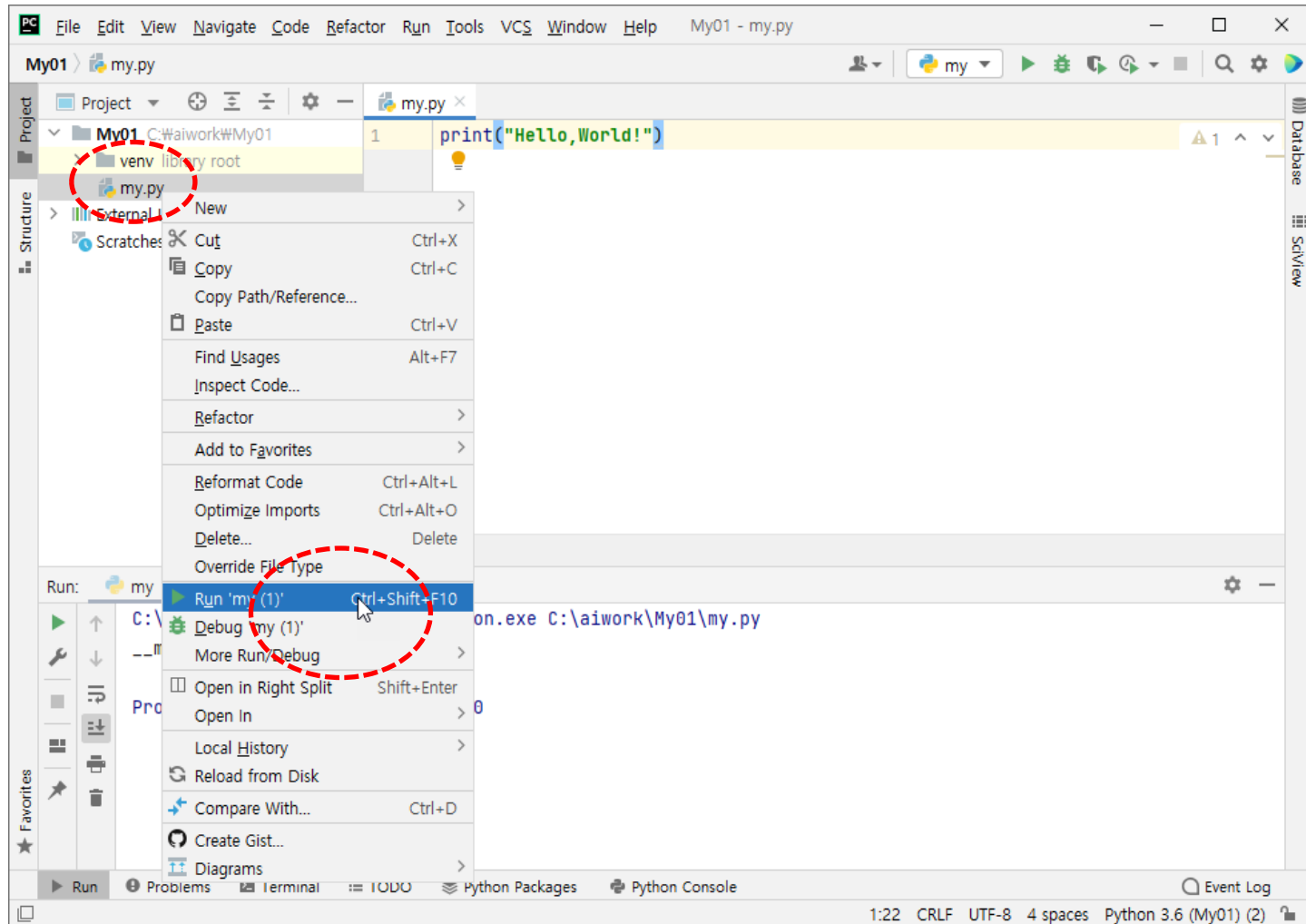
- 코드 입력

```
print("Hello,World!")
```

- 파이썬은 메인 함수가 없음.
- 들여쓰기 하지 않은 모든 코드(level 0 코드)가 실행됨.

3절. 아주 간단한 Python 프로그램

- 파일 이름 위에서 오른쪽 버튼 클릭 | 실행



3절. 아주 간단한 Python 프로그램

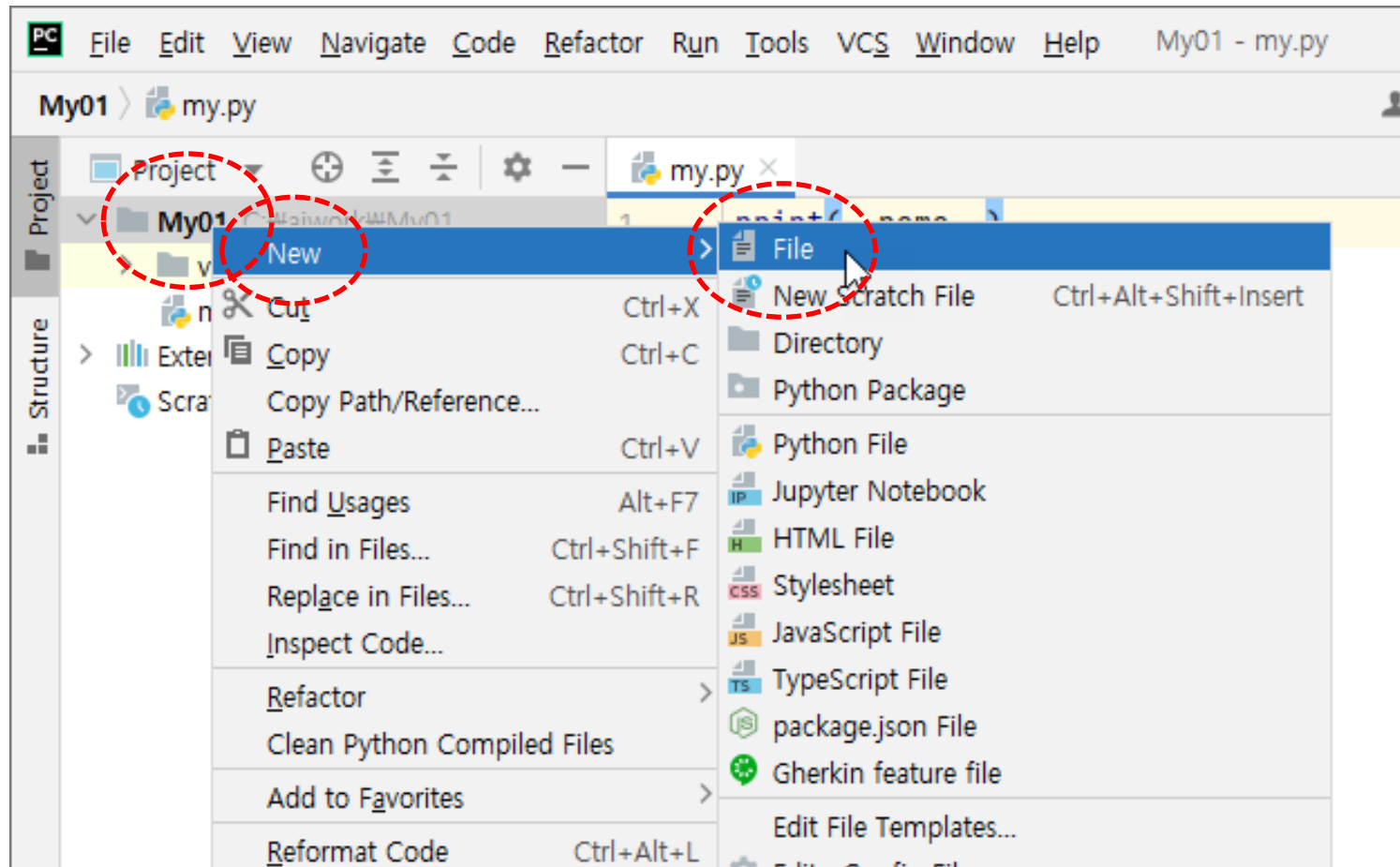
- 모듈 이름(__name__) 출력해보기

```
print(__name__)
```

파일 이름 위에서 오른쪽 버튼 클릭 | 실행

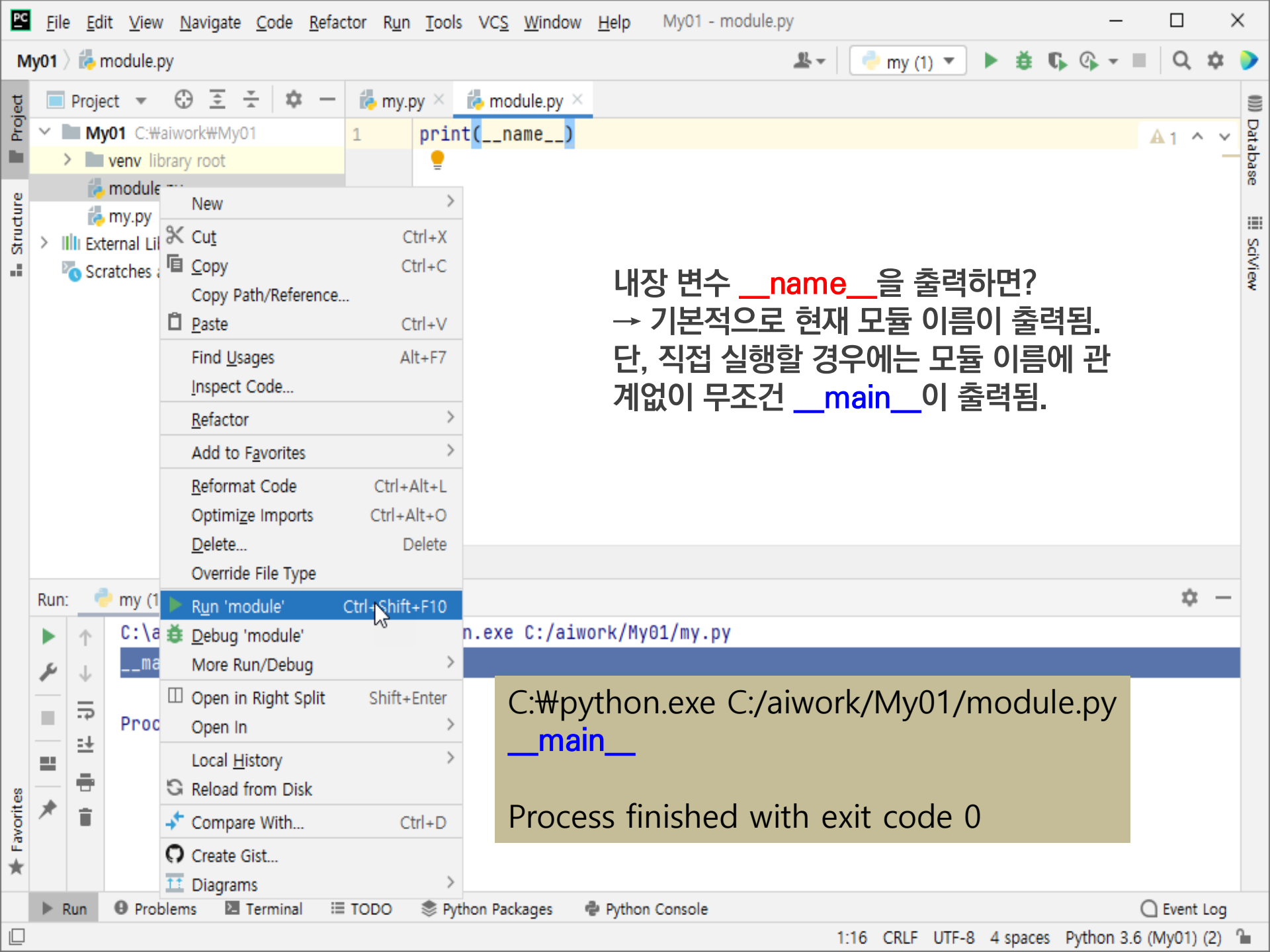
3절. 아주 간단한 Python 프로그램

- 새로운 파일 작성하기



New File

module.py



내장 변수 `__name__`을 출력하면?
→ 기본적으로 현재 모듈 이름이 출력됨.
단, 직접 실행할 경우에는 모듈 이름에 관계없이 무조건 `__main__`이 출력됨.

```
C:\python.exe C:/aiwork/My01/module.py  
__main__
```

```
Process finished with exit code 0
```

3절. 아주 간단한 Python 프로그램

- `my.py` 코드를 아래와 같이 수정 후 실행

```
import module
```

my.py

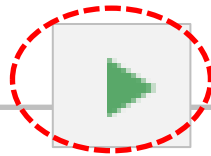
```
import module
```

module.py

```
print(__name__)
```

※ my.py 파일명 위에서 오른쪽 버튼을
클릭한 후 실행

실행결과



```
C:\wpython.exe C:/aiwork/My01/my.py  
module
```

```
Process finished with exit code 0
```

3절. 아주 간단한 Python 프로그램

- 파이썬 파일이 **직접 실행되면**, `__name__` 변수에는 `'__main__'` 값이 저장됨.
- 다른 곳에서 **import되어 간접 실행되면**, `__name__` 변수에는 파일명(가령 `'module'`)이 저장됨.



my.py

```
import module
```

module.py

```
if __name__ == "__main__":  
    print("직접 실행!")  
else:  
    print('다른 곳에서 실행!')
```

1. my.py에서 임포트하여 **간접적으로** 실행하면?
2. 하지만, module.py를 **직접** 실행하면?

“

결국, 직접 실행할 때에는
특정 코드가 실행되도록 하고,
**다른 곳에서 import하여 실행할 때는
실행되지 않도록** 할 수 있다. (일반적)

3절. 아주 간단한 Python 프로그램

```
print("Hello,World!")
```

4절. 조금 복잡한(?) Python 프로그램

```
iX = 2
```

```
iY = 3
```

```
iResult = iX + iY
```

```
print("Sum = ", iResult)
```

“

변수는 자동으로 만들어진다.
우리가 만들지 않는다.

변수란 무엇?



값을 담는 그릇

변수는 언제 자동으로 만들어질까?

값을 할당할 때



파이썬에서는 변수가 언제 만들어질까?

iX = 2 ←

iY = 3 ←

iResult = iX + iY ←

print("Sum = ", iResult)

파이썬에서는 변수가 언제 만들어질까?

iX = 0 ←

iY = 0 ←

값을 '처음으로' 할당 할 때

iX = 2

iY = 3

iResult = iX + iY ←

print("Sum = ", iResult)

어제 **뭐**했어요?

▶ 간단히 대답하면?

이 부분 **뭐**하는 코드?

▶ 간단히 대답하면?

```
iX = 2
```

```
iY = 3
```

```
iResult = iX + iY
```

```
print("Sum = ", iResult)
```

이 부분 **뭐**하는 코드?

▶ 간단히 대답하면?

```
iX = 2
```

```
iY = 3
```

```
iResult = iX + iY
```

```
print("Sum = ", iResult)
```

“

값을 할당(assign)하는
코드입니다.

5절. 추상화와 함수

```
def assign():
```

```
    iX = 2
```

```
    iY = 3
```

```
assign()
```

```
iResult = iX + iY
```

```
print("Sum = ", iResult)
```

5절. 추상화와 함수

- 추상화(abstraction)

- 복잡한 내용을 **묶어서 간단히** 표현하는 것

- 예) 어제 뭐했니? [대답]

- 코드 추상화

- 복잡한 **코드**를 묶어서 간단히 표현하는 것

- 예) 이 부분 뭐하는 코드? [assign, add]

무슨 문제?

```
def assign():
```

```
    iX = 2
```

```
    iY = 3
```

```
assign()
```

```
iResult = iX + iY
```

```
print("Sum = ", iResult)
```




```
iX = 0
```

```
iY = 0
```

“변수가 만들어지는 곳은?”

```
def assign():
```

```
    iX = 2
```

```
    iY = 3
```

전역(**global**) 변수
vs. 지역(local) 변수

```
assign()
```

```
iResult = iX + iY
```

```
print("Sum = ", iResult)
```

iX = 0

iY = 0

“전역변수를 이용하려면?”

```
def assign():  
    global iX, iY  
    iX = 2  
    iY = 3
```

```
assign()  
iResult = iX + iY
```

```
print("Sum = ", iResult)
```

따라서,
global 의미는?

“

지역변수 만들지 말고
전역변수 이용하라.

네 방 안에 화장실 만들지 말고
밖에 있는 공용 화장실 이용해라.

```
iX = 0
```

```
iY = 0
```

```
def assign():  
    global iX, iY  
    iX = 2  
    iY = 3
```

```
def add():  
    return iX + iY
```

```
assign()  
iResult = add()
```

```
print("Sum = ", iResult)
```



iX = 0



iY = 0



```
def assign():  
    global iX, iY  
    iX = 2  
    iY = 3
```



```
def add():  
    return iX + iY
```

```
assign()  
iResult = add()  
  
print("Sum = ", iResult)
```



어떻게 해야 할까?



6절. 두번째 추상화

문자 시작

```
iX = 0
```

```
iY = 0
```

```
def assign():
```

```
    global iX, iY
```

```
    iX = 2
```

```
    iY = 3
```

```
def add():
```

```
    return iX + iY
```

끝

```
assign()
```

```
iResult = add()
```

```
print("Sum = ", iResult)
```

묶자 **XXX** 시작

iX = 0

iY = 0

```
def assign():  
    global iX, iY  
    iX = 2  
    iY = 3
```

```
def add():  
    return iX + iY
```

끝

```
assign()
```

```
iResult = add()
```

```
print("Sum = ", iResult)
```

코드뿐만 아니라

변수(데이터)까지도

한나로 묶어서 간단히 표현(XXX)

→ 추상화

코드 변수(**데이터**) 추상화

변수(**데이터**) 추상화

데이터 추상화

묶어서 만든 **XXX**를 무엇이라고 할까?

묶자 **xxx** 시작

iX = 0

iY = 0

```
def assign():  
    global iX, iY  
    iX = 2  
    iY = 3
```

```
def add():  
    return iX + iY
```

끝

추상 자료형

자료형은 뭐하라고 있는 것?

추상 자료형

자료형 XXX는 뭐하라고 있는 것?

변수 만들라고 있는 것

`gildong = XXX()`

C++, C#, Java 스타일

`XXX gildong = new XXX()`



변수를 여러 개 만들어보자.

gildong = XXX()

youngja = XXX()

cheolsu = XXX()

gildong = 사람()
youngja = 사람()
cheolsu = 사람()

‘사람’이라는 부류(class)

‘XXX’라는 부류(class)

사람, XXX = 부류(class)

```
class XXX:  
    iX = 0  
    iY = 0  
  
    def assign():  
        global iX, iY  
        iX = 2  
        iY = 3  
  
    def add():  
        return iX + iY
```

```
gildong = XXX()  
  
gildong.assign()  
iResult = gildong.add()  
  
print("Sum = ", iResult)
```

```
class XXX:
    iX = 0
    iY = 0

    def assign(self):
        global iX, iY
        self.iX = 2
        self.iY = 3

    def add(self):
        return self.iX + self.iY
```

```
gildong = XXX()

gildong.assign()
iResult = gildong.add()

print("Sum = ", iResult)
```



```
class Point:
    iX = 0
    iY = 0

    def assign(self):
        self.iX = 2
        self.iY = 3

    def add(self):
        return self.iX + self.iY
```

```
gildong = Point()

gildong.assign()
iResult = gildong.add()

print("Sum = ", iResult)
```

7절. 다른 파일로 분리

#point.py

```
class Point:
    iX = 0
    iY = 0

    def assign(self):
        self.iX = 2
        self.iY = 3

    def add(self):
        return self.iX + self.iY
```

#my.py

```
import point

gildong = point.Point()

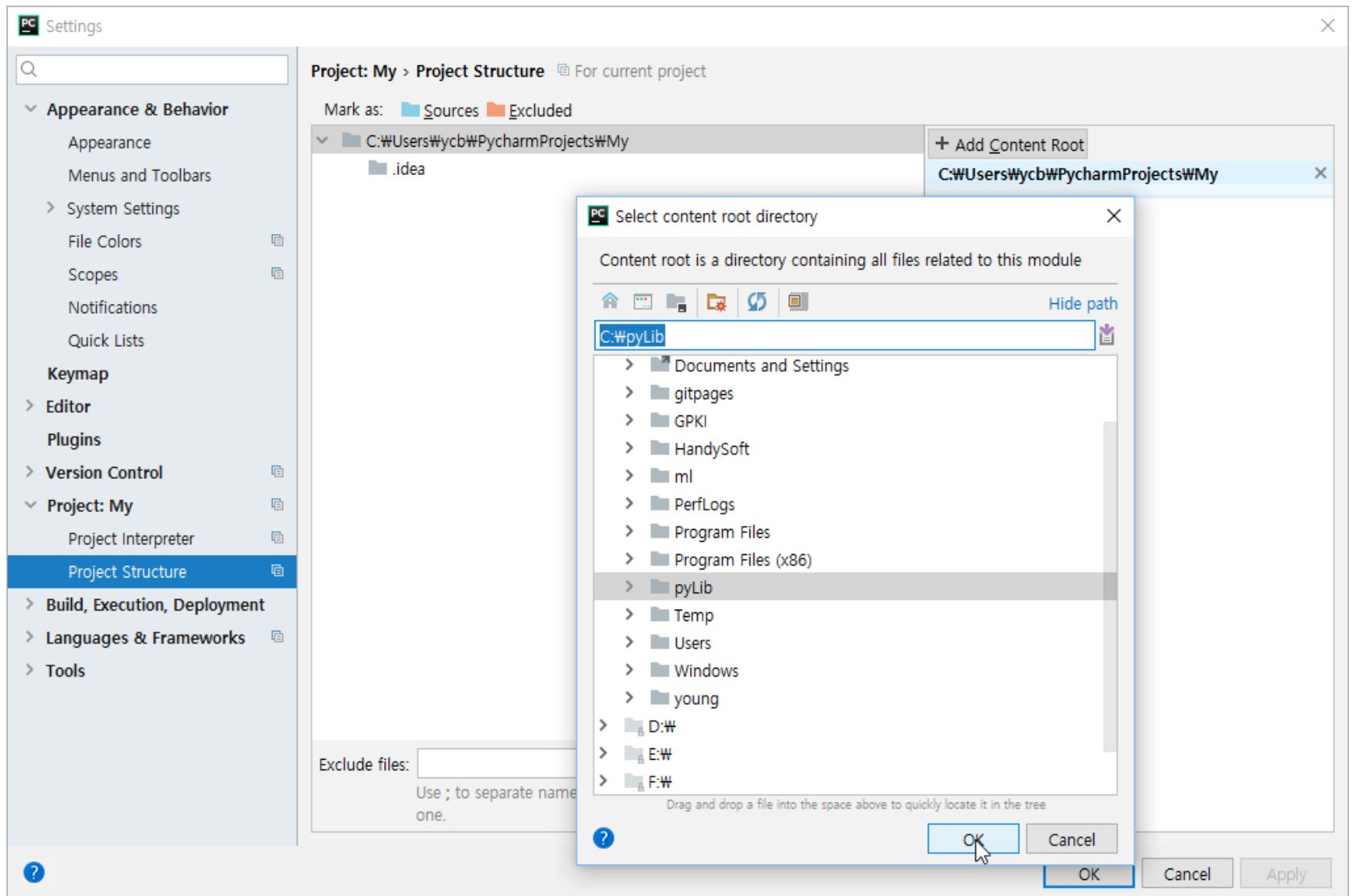
gildong.assign()
iResult = gildong.add()

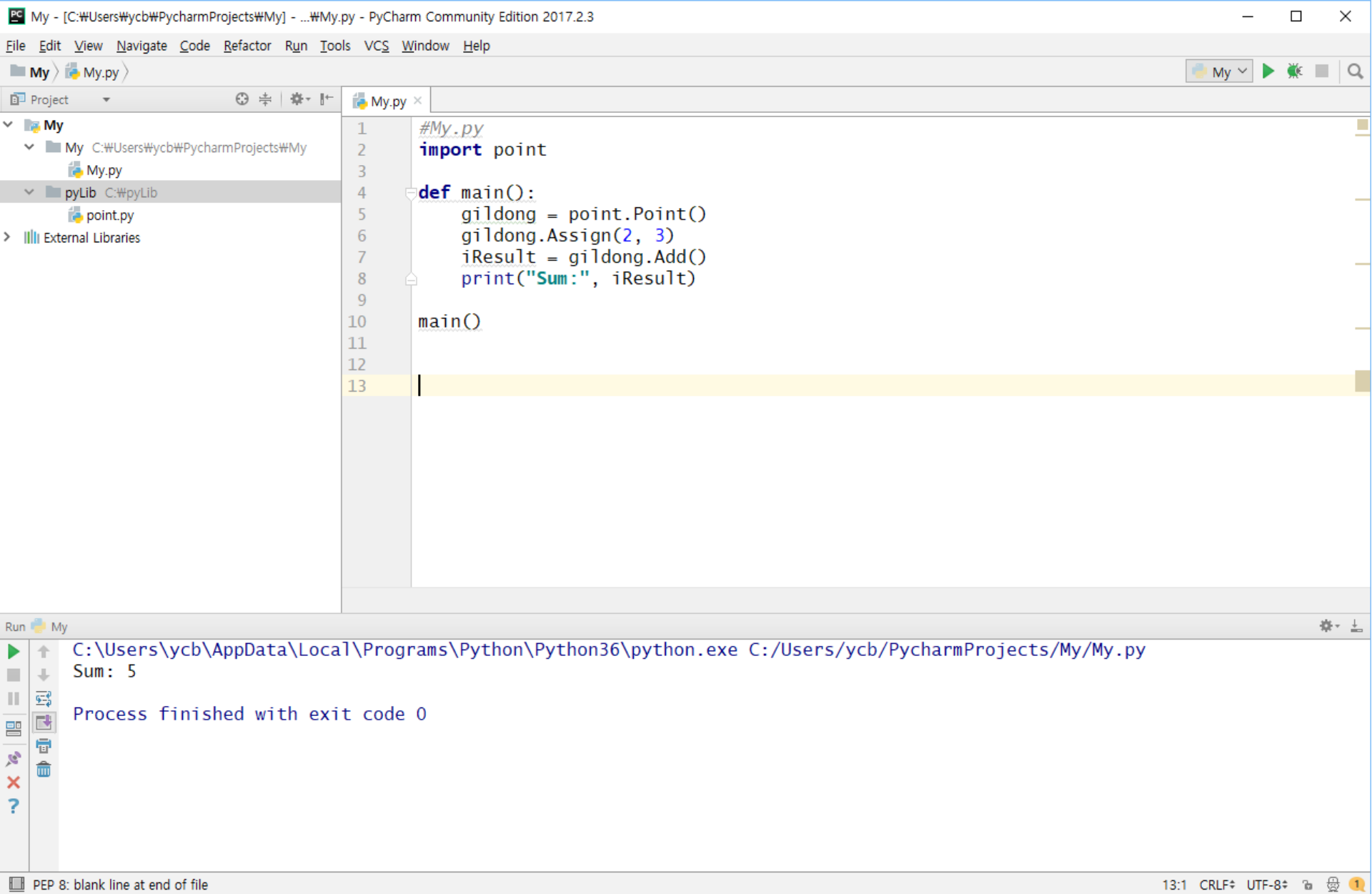
print("Sum = ", iResult)
```

8절. 나만의 라이브러리 폴더로 저장

- c:\pyLib 폴더 생성 후 point.py를 그 곳으로 이동

8절. 나만의 라이브러리 폴더 설정





9절. 모듈 사용 방법

```
import point
```

파일/모듈 명 클래스 명



```
gildong = point.Point()
```

```
gildong.assign()
```

```
iResult = gildong.add()
```

```
print("Sum:", iResult)
```

9절. 모듈 사용 방법

```
import point as p
```

```
gildong = p.Point()
```

```
gildong.assign()
```

```
iResult = gildong.add()
```

```
print("Sum:", iResult)
```

9절. 모듈 사용 방법

```
from point import Point
```

```
gildong = Point()
```

```
gildong.assign()
```

```
iResult = gildong.add()
```

```
print("Sum:", iResult)
```