

Python개요

안 화 수

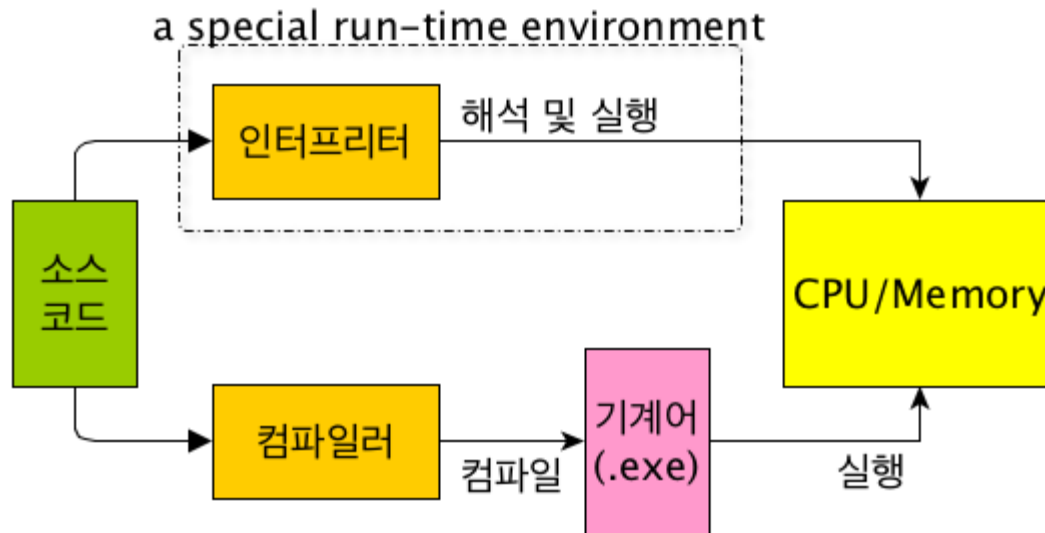
1. Python

❖ 파이썬

- ✓ 스크립트 언어의 한 종류로 1991년 프로그래머인 귀도 반 로섬(Guido van Rossum)이 발표한 고급 프로그래밍 언어로 현재 구글의 3대 개발 언어중에 하나 입니다.
- ✓ 파이썬이란 이름은 귀도가 좋아하는 코미디 프로그램인 "Monty Python's Flying Circus" 에서 따왔으며, 파이썬의 사전적인 의미를 나타내는 뱀을 아이콘으로 사용하고 있습니다.

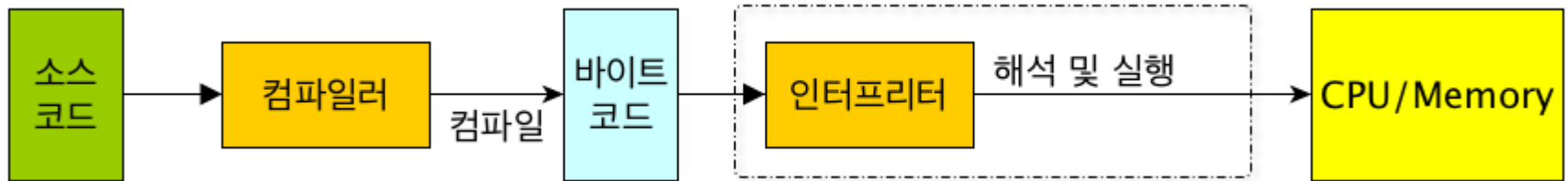
1. Python

- ❖ **컴파일(Compile) 방식**의 언어에 의해 작성되는 응용 프로그램은 컴파일러에 의해 기계어로 번역된 채로 실행되기 때문에, 수정이 빈번하게 발생할 경우에는 수정 후 다시 컴파일을 해야합니다.
 - ✓ 덩치가 큰 프로그램은 컴파일 시간이 오래 걸립니다.
 - ✓ 간단한 수정에도 오랜 기간의 컴파일 시간이 요구됩니다.



1. Python

- ❖ 반면에 수정이 빈번하게 발생하는 경우에는 소스 코드를 줄 단위로 읽어 바로 실행하는 **인터프리터 방식**이 상당히 유리합니다.
- ❖ 스크립트 소스코드를 컴파일 방식에 의해 중간 코드(Bytecode)로 우선 만들고, 이를 다시 인터프리터 방식으로 해석하여 수행



- ❖ 스크립트 언어의 장점
 - ✓ 개발 시간이 단축
 - ✓ 소스 코드 수정이 빠르고 간단
- ❖ 스크립트 언어의 단점
 - ✓ 중간 코드를 만드는 것은 간단하지만 그것을 실제로 실행시키는 것은 많은 작업이 필요
 - ✓ 실행 시간이 오래 걸림

1. Python

❖ 대표적인 스크립트 언어

- ✓ JavaScript
- ✓ ActionScript
- ✓ Perl
- ✓ PHP
- ✓ Python
- ✓ Scala
- ✓ Lua
- ✓ Ruby

1. Python

❖ 파이썬의 특징

✓ 들여쓰기 - 가독성

파이썬의 문법은 간결하고 가독성이 좋습니다.

특히 코드블럭을 들여쓰기(indentation)로 구분하여, 가독성이 높아지는 구조로 되어 있습니다.

✓ 풍부한 라이브러리

파이썬에서는 매우 광범위한 라이브러리가 기본적으로 포함되어 있으며, 확장성도 무궁무진 합니다.

✓ 접착성

기본적으로 제공되는 라이브러리 말고도 쉽게 라이브러리를 추가할 수 있는데, C로 구현되어 있는 모듈을 쉽게 만들어 붙일 수 있습니다.

특히 파이썬은 C보다는 느리므로 속도 문제가 생기는 부분은 C로 구현해 붙일때 유용합니다.

또한 파이썬에는 없고 C에는 이미 있는 기능을 붙여서 사용할 수 있습니다. 반대로 파이썬의 기능을 C에서 사용할수도 있습니다.

1. Python

❖ 파이썬의 특징

✓ 라이선스 - 무료

파이썬은 파이썬 소프트웨어 재단(Python Software Foundation)에서 관리하고 있으며, 라이선스는 거의 무료와 다름없는 Python Software Foundation License를 따르고 있습니다.

✓ 유니코드

파이썬에서는 문자열이 모두 유니코드로 처리 됩니다. 즉, 한글, 한자 등을 표현하기 위해 별도로 인코딩을 하지 않아도 된다.

✓ 동적 타이핑과 자동 메모리 관리 기능

파이썬은 런타임 시에 타입 체크를 하는 동적 타이핑과 자동으로 메모리를 관리해주는 기능이 있습니다.

1. Python

❖ 파이썬의 장점

- ✓ Guido가 생각했던 Python 문법적 특징은 들여쓰기를 철저하게 지키도록 언어로 설계
- ✓ 코드의 가독성이 좋음
- ✓ C 언어에서처럼 { } 등의 괄호를 넣지 않기 때문에 프로그램을 좀더 깔끔함
- ✓ 파이썬 코드는 재사용하기가 쉬움
- ✓ 코드의 분석이 쉽기 때문에 다른 사람이 작성한 코드를 받아서 작업하는 사람들이 훨씬 더 작업이 편리
- ✓ 생태계가 좋음

1. Python

❖ 파이썬의 구현

- ✓ C파이썬: C로 작성된 인터프리터를 사용하는 일반적인 파이썬으로 ipython이라고도 합니다.
- ✓ 스택리스 파이썬 : C 스택을 사용하지 않는 인터프리터
- ✓ 자이썬 : 자바 가상 머신 용 인터프리터.
과거에는 제이파이썬(Jpython)이라고 불리기도 함
- ✓ IronPython : .NET 플랫폼 용 인터프리터
- ✓ PyPy : 파이썬으로 작성된 파이썬 인터프리터.

❖ 사용 가능 플랫폼

- ✓ 마이크로소프트 윈도우(9x/NT 계열은 최신판, 3.1 및 MS-DOS는 옛 버전만)
- ✓ 매킨토시(맥 OS 9 이전, 맥 OS X 이후 포함)
- ✓ 각종 유닉스
- ✓ 리눅스
- ✓ 팜 OS
- ✓ 노키아 시리즈 60

1. Python

❖ 파이썬의 활용 분야

- ✓ GUI Programming: 기본 모듈인 Tkinter 이용, PyQt
- ✓ Web Programming: Django, Flask 프레임워크
- ✓ Game Programming: PyOpenGL
- ✓ Database Programming
- ✓ Text 처리
- ✓ 수치 연산 Programming: Numeric Python 모듈이나 nextworkx모듈
- ✓ 병렬 연산 Programming
- ✓ 사물인터넷 - 라즈베리 파이
- ✓ C, C++, Java 와 결합한 프로그래밍
- ✓ 데이터 분석 분야: NumPy, Pandas, Scipy, Matplotlib

❖ 빠르게 실행되어야 하는 애플리케이션이나 여러 개의 스레드를 한꺼번에 처리해야 하는 애플리케이션에서는 부적합

2. 파이썬 설치

❖ 파이썬의 버전

- ✓ 파이썬은 2.x 버전과 3.x 버전이 있는데 파이썬은 2.x버전 과 3.x버전 간의 호환성을 유지하고 있지 않습니다.
- ✓ 변화
 - ◆ print가 함수로 변경
 - ◆ long 형이 없어지고 정수는 int로 통일
 - ◆ int / int 의 결과는 float
 - ◆ 문자열이 string과 unicode로 구분되었는데 string 과 bytes로 구분됩니다.
- ✓ 2to3.py를 이용해서 2.x 버전을 3 버전으로 변환가능(파이썬 설치 디렉토리의 Tools□Scripts 디렉토리에 존재)

❖ 파이썬 다운로드 및 설치

<https://www.python.org/downloads/>

윈도우 용을 설치할 때 파이썬 명령어가 있는 디렉토리를 path에 추가하는 옵션을 체크해야 이클립스에서 파이썬 설정을 자동으로 할 수 있습니다.

2. 파이썬 설치



2. 파이썬 설치

파이썬 배포판 - 아나콘다: <https://www.anaconda.com/downloads>

- 파이썬 배포판: Ipython, Jupyter notebook, Qt Console, Python 포함
- 데이터 분석, 수학, 과학 관련 다양한 라이브러리 포함
: numpy, pandas, scipy, matplotlib etc

Anaconda Installers

Windows 	MacOS 	Linux 
Python 3.9	Python 3.9	Python 3.9
64-Bit Graphical Installer (594 MB)	64-Bit Graphical Installer (591 MB)	64-Bit (x86) Installer (659 MB)
32-Bit Graphical Installer (488 MB)	64-Bit Command Line Installer (584 MB)	64-Bit (Power8 and Power9) Installer (367 MB)
	64-Bit (M1) Graphical Installer (316 MB)	64-Bit (AWS Graviton2 / ARM64) Installer (568 MB)
	64-Bit (M1) Command Line Installer (305 MB)	64-bit (Linux on IBM Z & LinuxONE) Installer (280 MB)

2. 파이썬 설치

Anaconda3 2021.11 (64-bit) Setup



Select Installation Type

Please select the type of installation you would like to perform for Anaconda3 2021.11 (64-bit).

Install for:

☒ Just Me (recommended)

☐ All Users (requires admin privileges)

Anaconda, Inc.

< Back

Next >

Cancel

2. 파이썬 설치

Anaconda3 2021.11 (64-bit) Setup



Advanced Installation Options

Customize how Anaconda integrates with Windows

Advanced Options

☒ Add Anaconda3 to my PATH environment variable

Not recommended. Instead, open Anaconda3 with the Windows Start menu and select "Anaconda (64-bit)". This "add to PATH" option makes Anaconda get found before previously installed software, but may cause problems requiring you to uninstall and reinstall Anaconda.

☒ Register Anaconda3 as my default Python 3.9

This will allow other programs, such as Python Tools for Visual Studio, PyCharm, Wing IDE, PyDev, and MSI binary packages, to automatically detect Anaconda as the primary Python 3.9 on the system.

Anaconda, Inc.

< Back

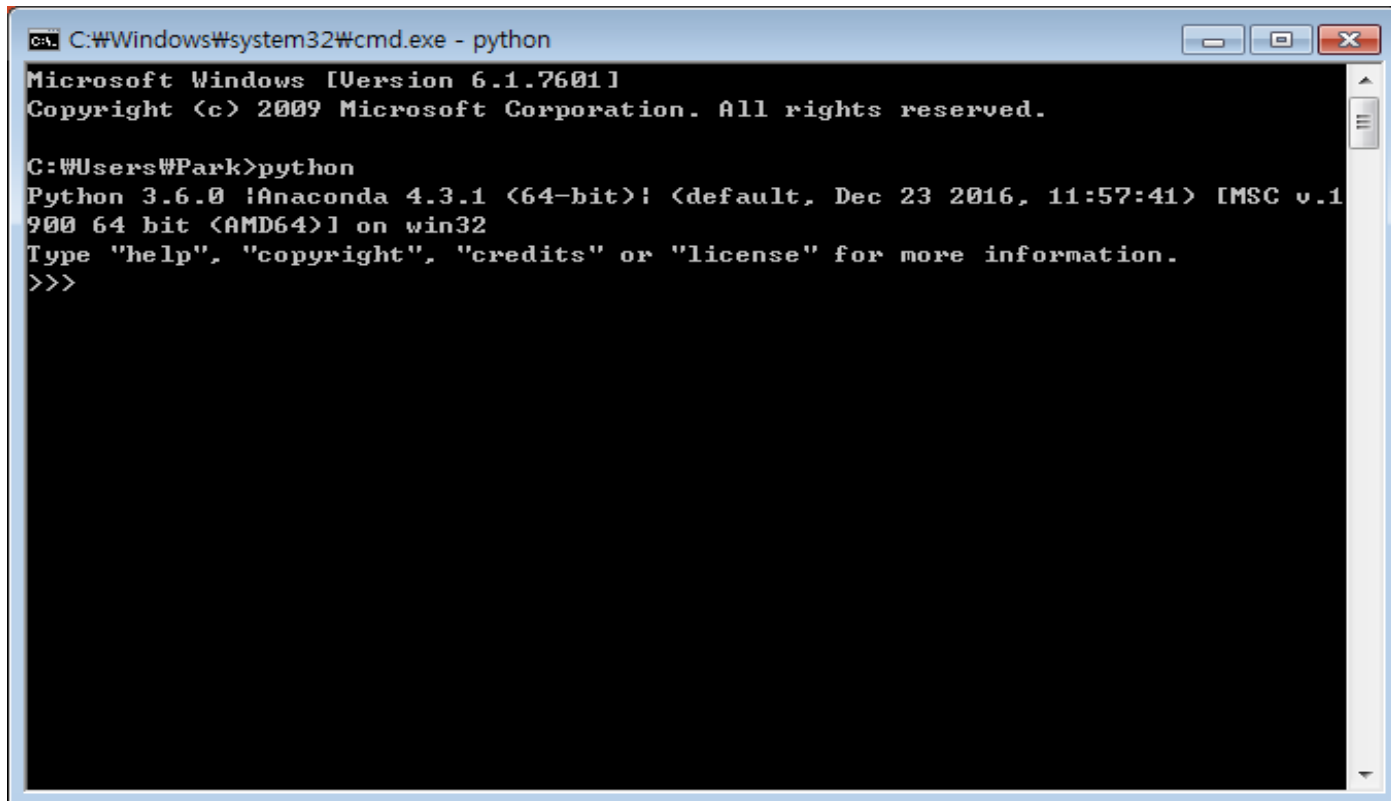
Install

Cancel

2. 파이썬 설치

❖ 설치 확인

- ✓ cmd를 실행
- ✓ python 이라고 입력 한 후 버전 확인



```
C:\Windows\system32\cmd.exe - python
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Park>python
Python 3.6.0 |Anaconda 4.3.1 (64-bit)| (default, Dec 23 2016, 11:57:41) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

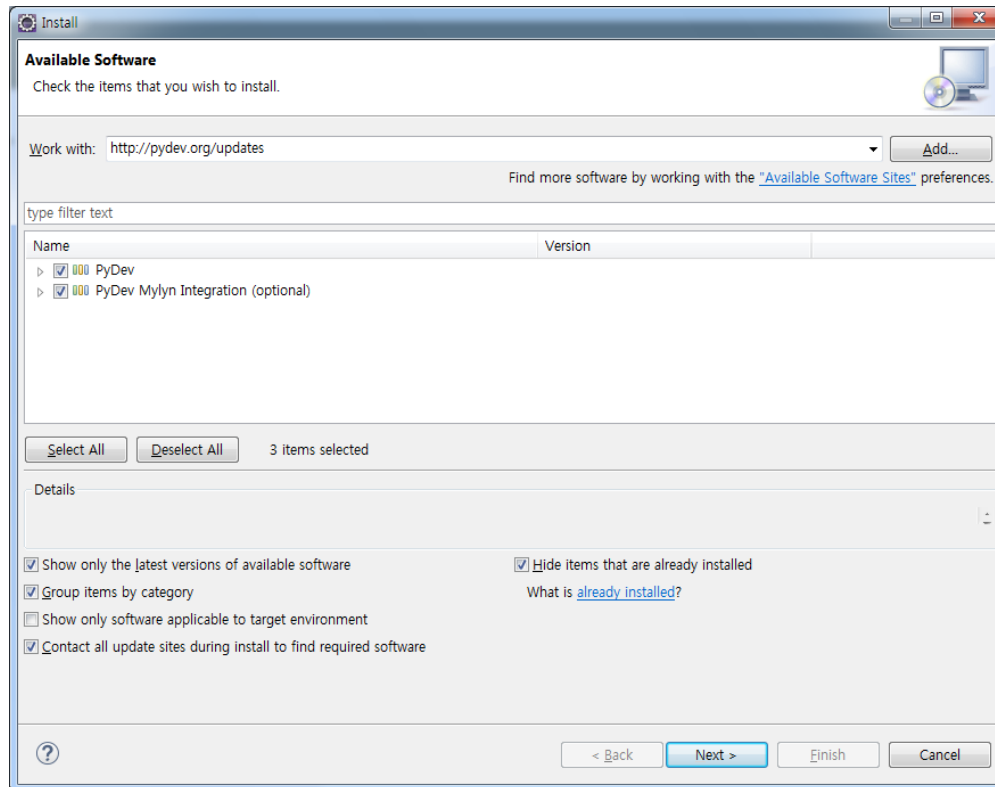

3. IDE

IDE

- ❖ ipython: 아나콘다를 설치하면 같이 설치
- ❖ PyDev 플러그 인을 이용한 이클립스
- ❖ Visual Studio의 파이썬 도구
- ❖ PyCharm
- ❖ Spyder

3. IDE

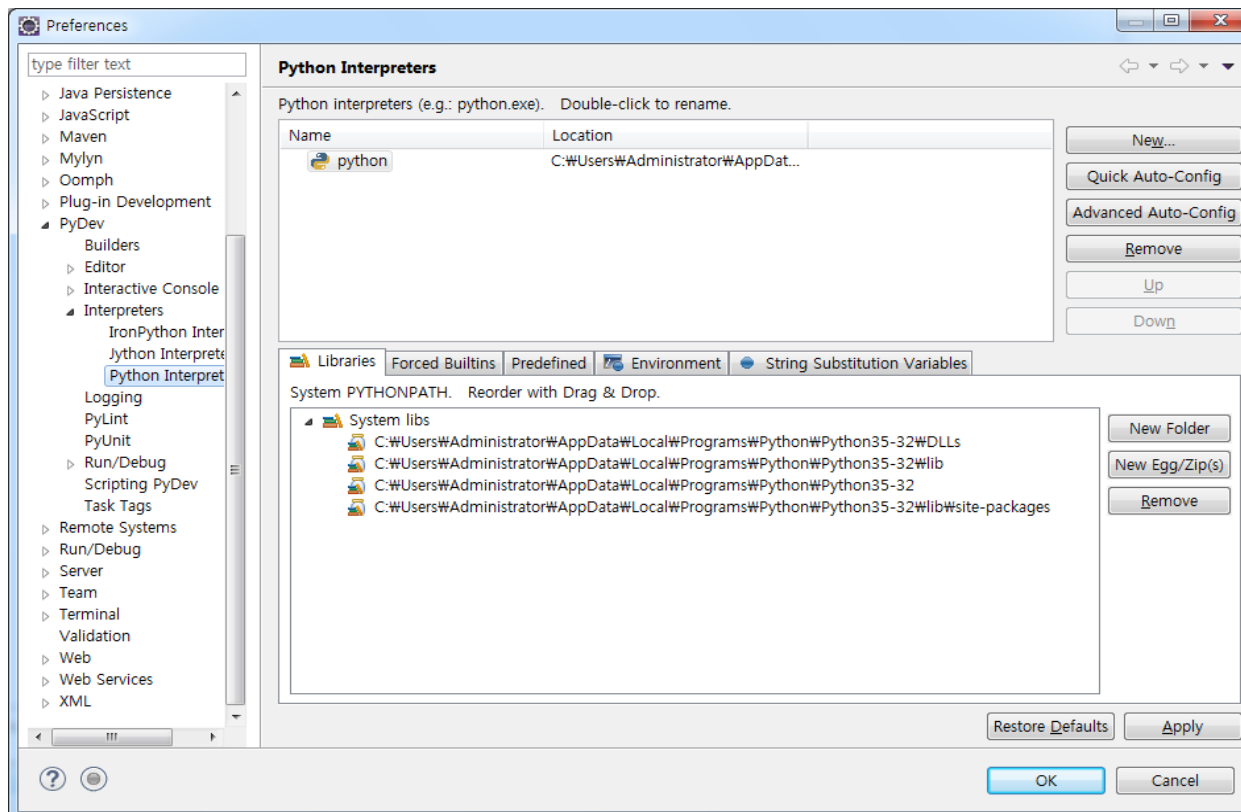
- ❖ Eclipse 플러그 인을 이용한 python
- ❖ 이클립스의 메뉴에서 "Help > Install New Software..." 를 선택하고 <http://pydev.org/updates> 또는 Eclipse Marketplace에서 python을 검색한 후 pydev 선택해서 설치



3. IDE

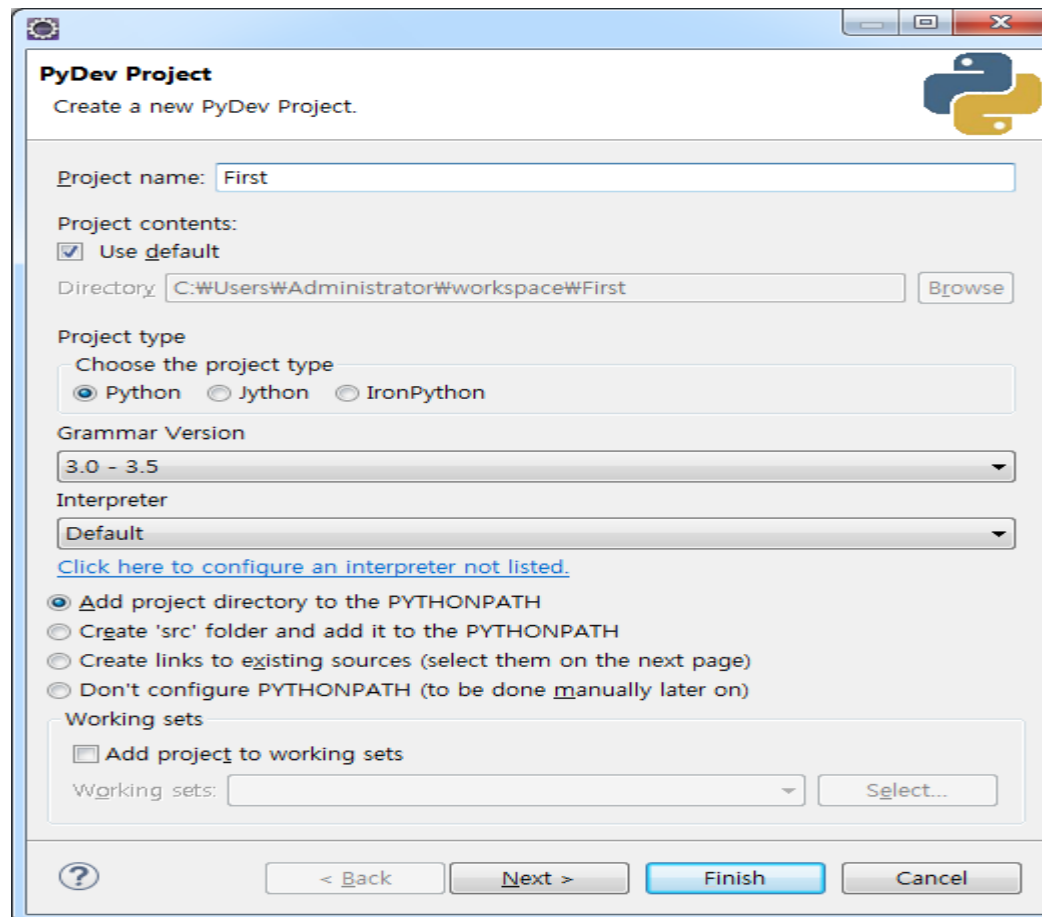
❖ 인터프리터 설정하기

- 'Windows > Preferences'를 선택
- 'PyDev > Interpreters > Python Interpreter' 선택하고 'Quick Auto-Config'



3. IDE

- ❖ perspective를 PyDev로 변경
- ❖ [File] - [New] - [PyDev Project]

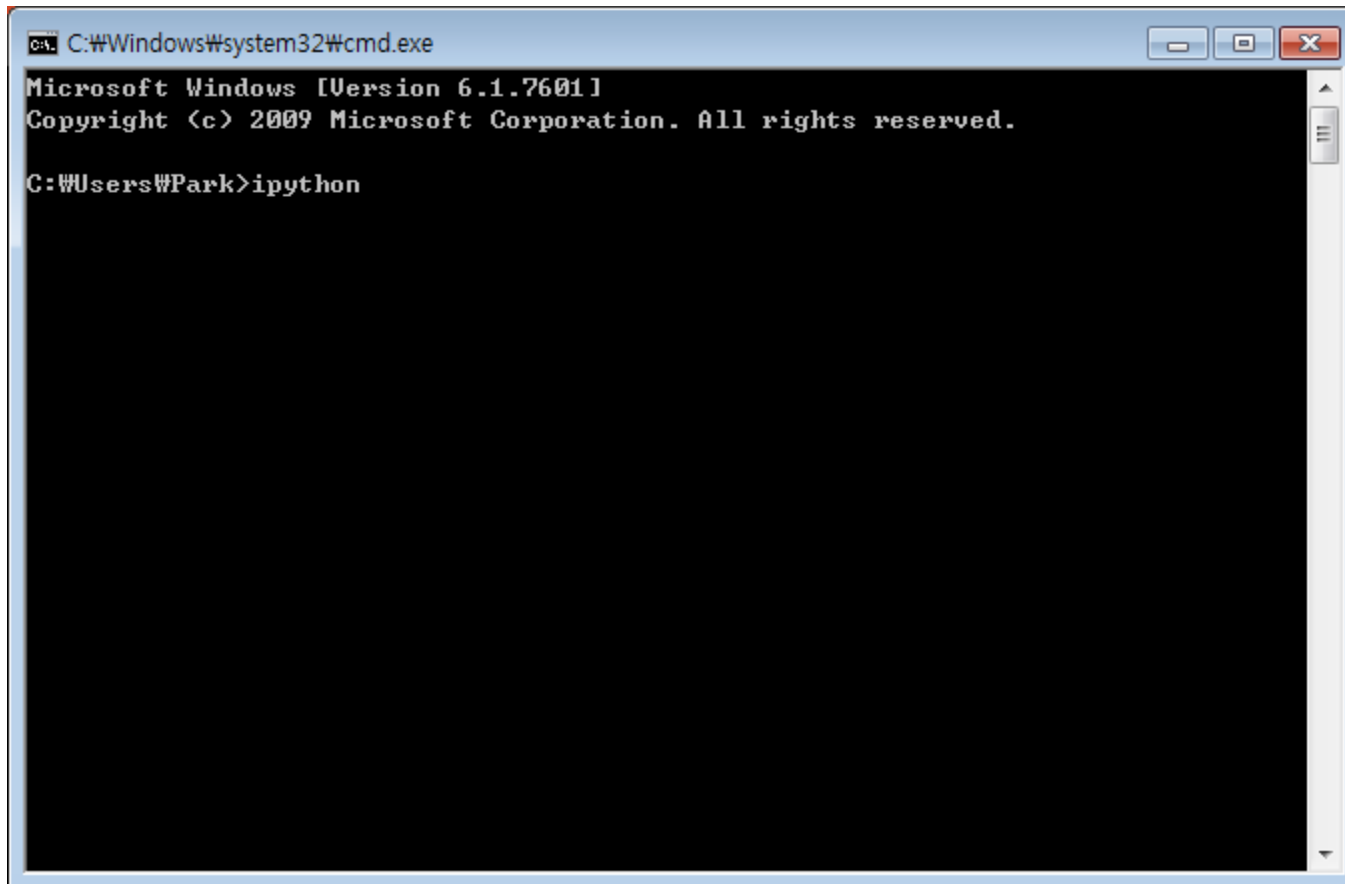


3. IDE

- ❖ 프로젝트를 선택하고 마우스 오른쪽 클릭 후 [New] – [File]을 선택하고 파일명을 pythontest.py를 입력 한 후 파일 생성
- ❖ 코드 작성
`print('Hello Python')`
- ❖ 실행
`ctrl+F11`
[Run] – [Run]

3. IDE

- ❖ ipython을 이용한 코딩 및 실행

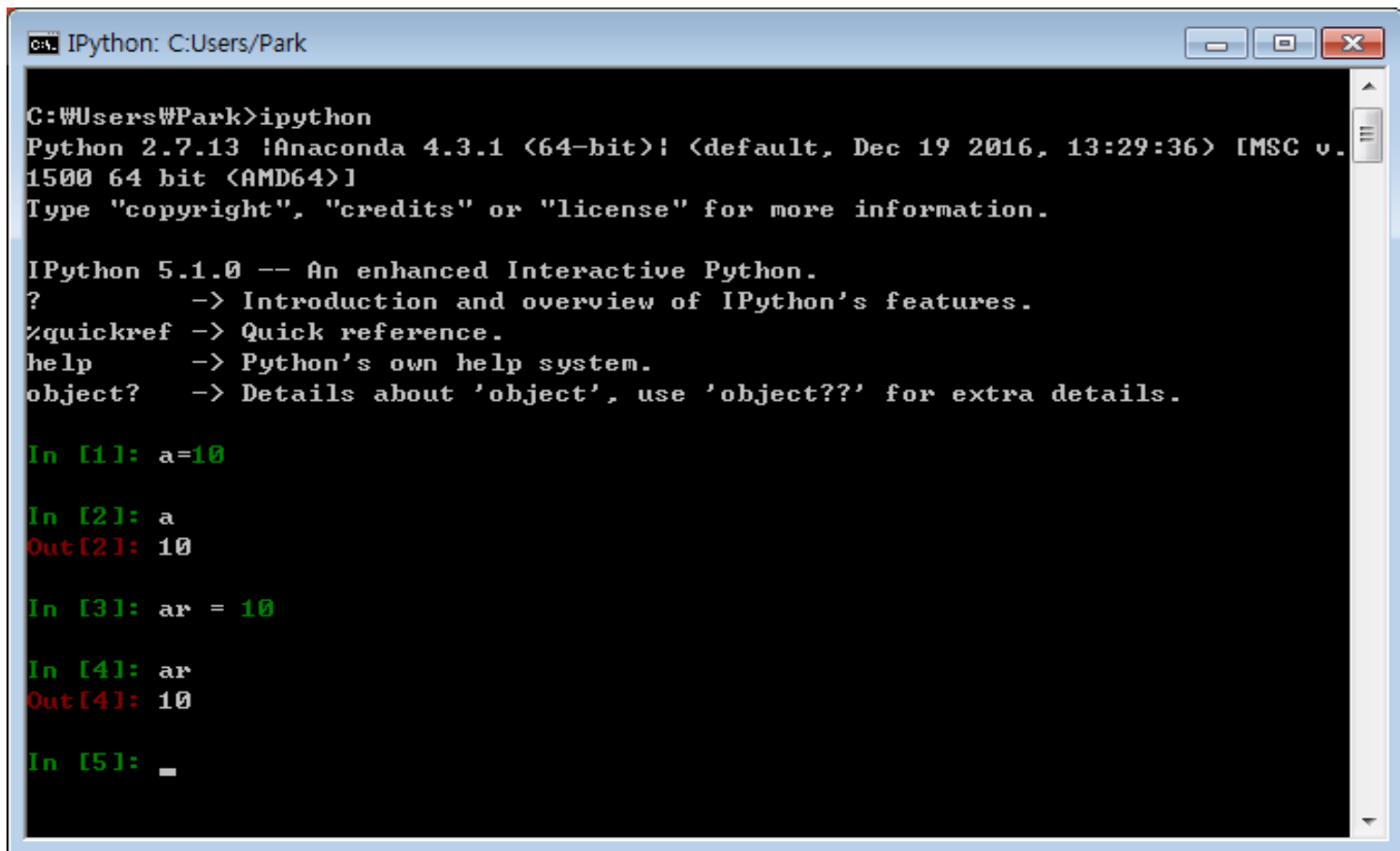


A screenshot of a Windows command prompt window. The title bar shows the path `C:\Windows\system32\cmd.exe`. The window contains the following text:

```
Microsoft Windows [Version 6.1.7601]  
Copyright (c) 2009 Microsoft Corporation. All rights reserved.  
  
C:\Users\Park>ipython
```

3. IDE

- ❖ 객체의 이름만 입력하면 바로 출력



```
IPython: C:\Users\Park

C:\Users\Park>ipython
Python 2.7.13 |Anaconda 4.3.1 (64-bit)| <default, Dec 19 2016, 13:29:36> [MSC v.
1500 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 5.1.0 -- An enhanced Interactive Python.
?                -> Introduction and overview of IPython's features.
%quickref        -> Quick reference.
help             -> Python's own help system.
object?         -> Details about 'object', use 'object??' for extra details.

In [1]: a=10

In [2]: a
Out[2]: 10

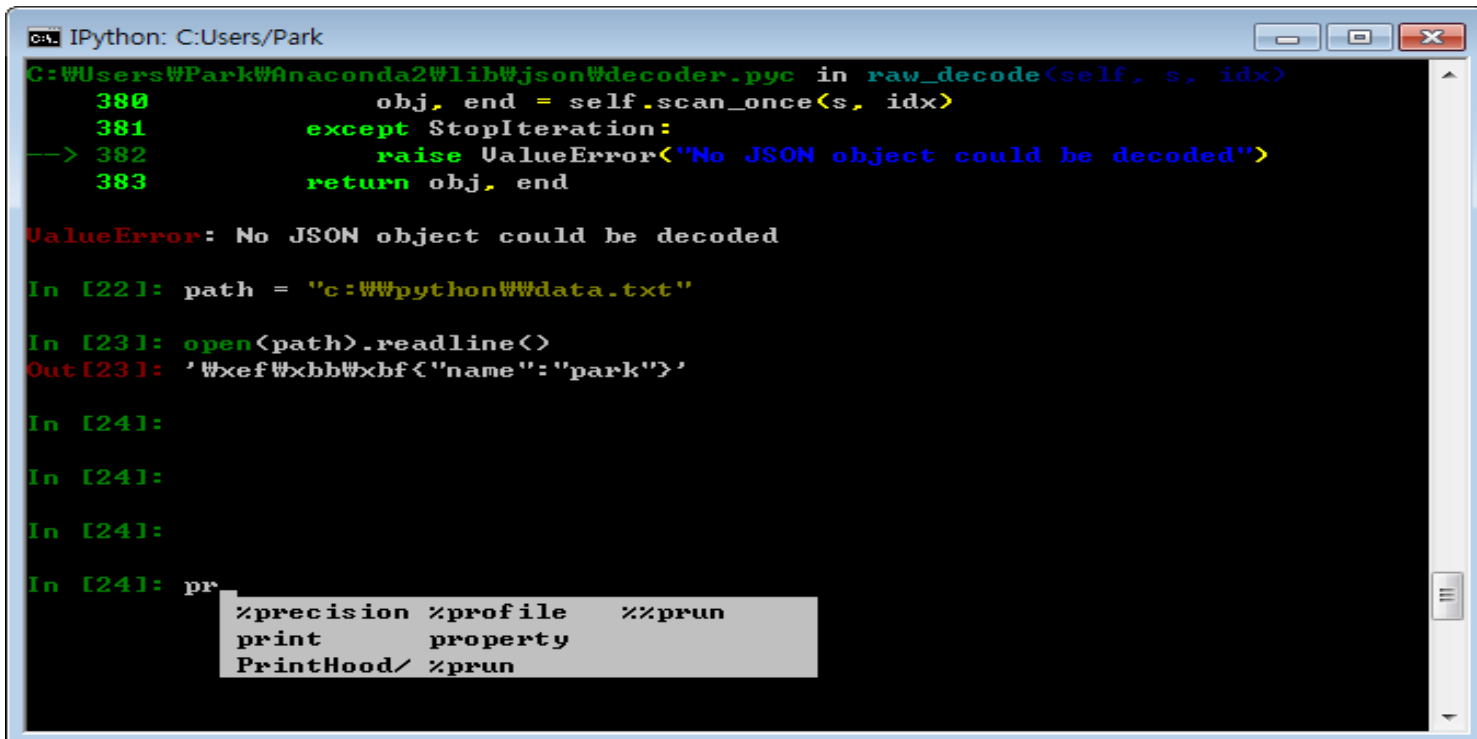
In [3]: ar = 10

In [4]: ar
Out[4]: 10

In [5]: _
```

4.ipython

- ❖ 탭을 이용한 자동완성 기능 제공: 명령을 입력하는 동안 Tab을 누르면 현재 위치에서 사용할 수 있는 변수를 화면에 출력
- ❖ 변수 이름 앞이나 뒤에 ?를 입력하면 변수에 대한 정보를 출력
- ❖ 함수 이름 뒤에 ??를 입력하면 함수의 소스 코드 출력



```
IPython: C:\Users\Park
C:\Users\Park\Anaconda2\lib\json\decoder.py in raw_decode(self, s, idx)
    380         obj, end = self.scan_once(s, idx)
    381     except StopIteration:
--> 382         raise ValueError("No JSON object could be decoded")
    383     return obj, end

ValueError: No JSON object could be decoded

In [22]: path = "c:\\python\\data.txt"

In [23]: open(path).readline()
Out[23]: ' {\xef\xbb\xbf {"name": "park"}}'

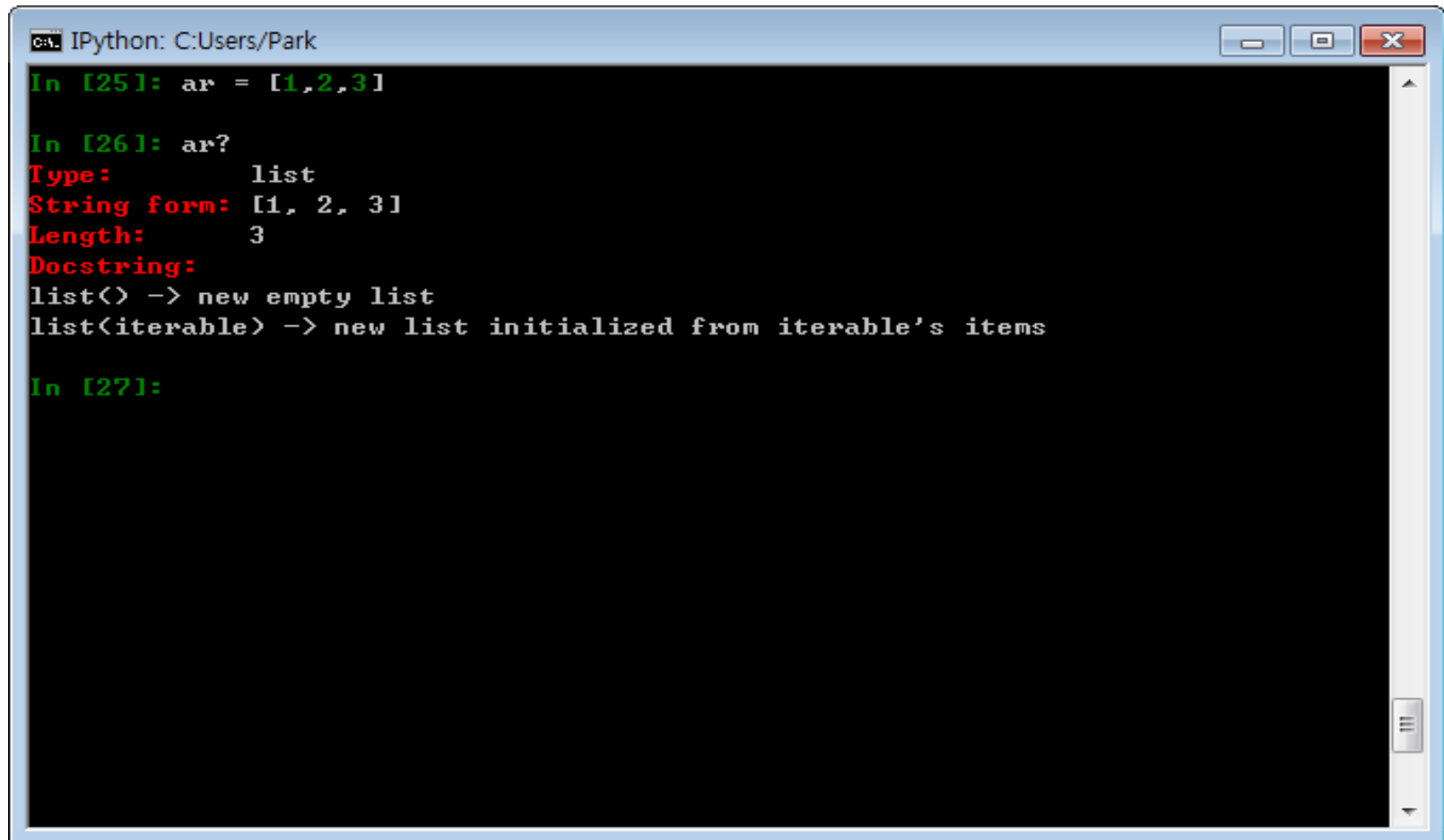
In [24]:

In [24]:

In [24]:

In [24]: pr
    %precision %profile    %%prun
    print      property
    PrintHood/ %prun
```


4.ipython



The screenshot shows an IPython terminal window with the title bar "IPython: C:\Users\Park". The terminal has a black background with green text for input and red text for output. The user has entered three commands: "In [25]: ar = [1,2,3]", "In [26]: ar?", and "In [27]:". The output for the second command shows the type, string form, length, and docstring of the list object.

```
IPython: C:\Users\Park
In [25]: ar = [1,2,3]
In [26]: ar?
Type:      list
String form: [1, 2, 3]
Length:    3
Docstring:
list() -> new empty list
list(iterable) -> new list initialized from iterable's items
In [27]:
```

4.ipython

- ❖ 작업 디렉토리 확인

```
In [1]: import os
```

```
In [2]: os.getcwd()
```

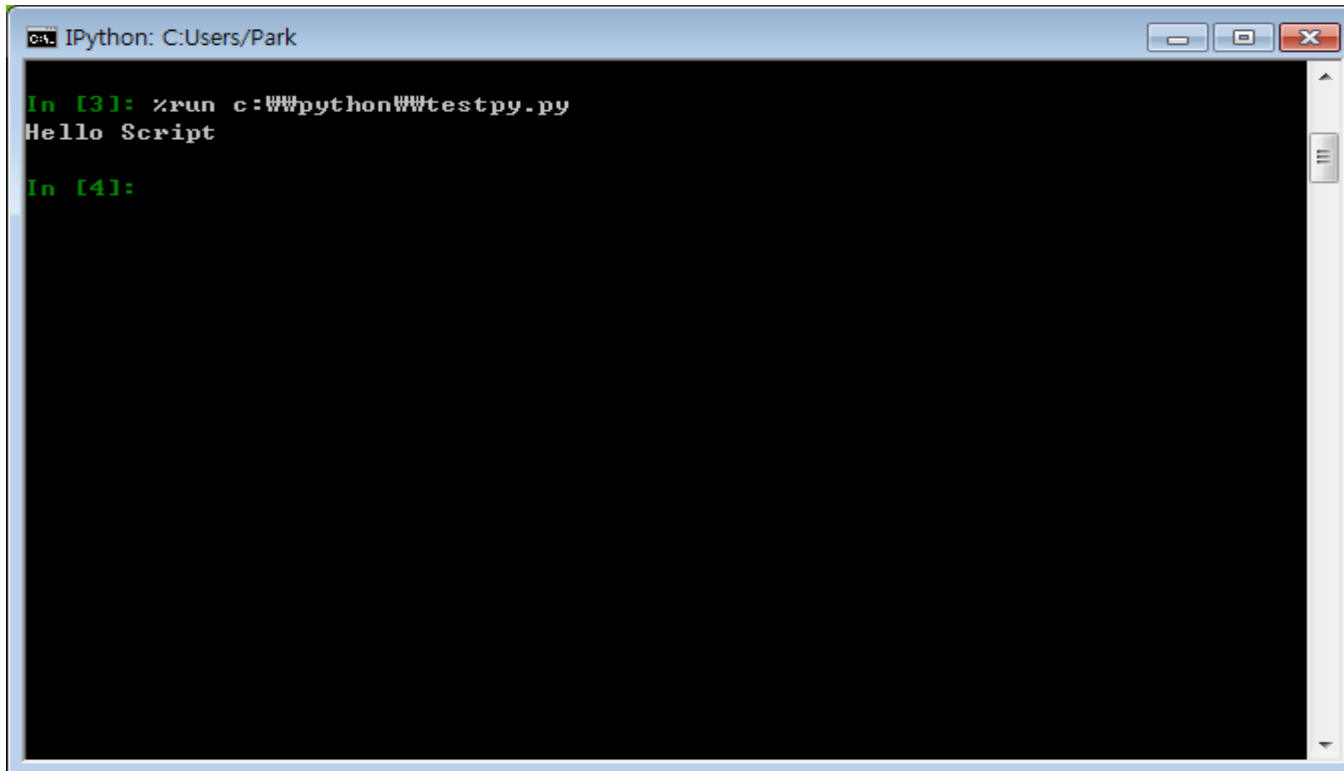
```
Out[2]: 'C:\Users\Administrator'
```

- ❖ 작업 디렉토리 변경

```
In [4]: os.chdir('c:\python')
```

4.ipython

- ❖ %run 파일경로: 파일 경로에 해당하는 파이썬 파일 실행
- ❖ 아래 코드를 갖는 파이썬 파일을 작성하고 실행
`print("Hello Script")`



The screenshot shows an IPython terminal window titled "IPython: C:\Users\Park". The terminal has a black background with green text. It shows the command `In [3]: %run c:\www\python\testpy.py` being entered, followed by the output `Hello Script`. Below this, the prompt `In [4]:` is visible, indicating the next command is ready to be entered.

```
IPython: C:\Users\Park

In [3]: %run c:\www\python\testpy.py
Hello Script

In [4]:
```

4.ipython

❖ 단축키

- Ctrl+P 또는 위 화살표 키: 명령어 히스토리를 역순으로 검색하기
- Ctrl+N 또는 아래 화살표 키: 명령어 히스토리에서 최근 순으로 검색하기
- Ctrl+R: readline 명령어 형식의 히스토리 검색(부분 매칭)하기
- Ctrl+Shift+V: 클립보드에서 텍스트 붙여넣기
- Ctrl+C: 현재 실행 중인 코드 중단하기
- Ctrl+A: 커서의 줄의 처음으로 이동하기
- Ctrl+E: 커서의 줄의 끝으로 이동하기
- Ctrl+K: 커서가 놓은 곳부터 줄의 끝까지 텍스트 삭제하기
- Ctrl+U: 현재 입력된 모든 텍스트 지우기
- Ctrl+F: 커서를 앞으로 한 글자씩 이동하기
- Ctrl+B: 커서를 뒤로 한 글자씩 이동하기
- Ctrl+L: 화면 지우기

4.ipython

❖ 매직 명령어: ipython이 제공하는 특수 명령어

%quickref: ipython의 빠른 도움말 표시

%magic: 모든 매직 함수에 대한 상세 도움말 출력

%debug: 최근 예외 트레이스백의 하단에서 대화형 디버거로 진입

%hist: 명령어 입력(그리고 선택적 출력) history 출력

%pdb: 예외가 발생하면 자동으로 디버거로 진입

%paste: 클립보드에서 들여쓰기가 된 채로 파이썬 코드 가져오기

%cpaste: 실행 파이썬 코드를 수동으로 붙여 넣을 수 있는 프롬프트 표시

%reset: 대화형 네임스페이스에서 정의된 모든 변수와 이름을 삭제

%page OBJECT: 객체를 pager를 통해 보기 좋게 출력

%run script.py: ipython 내에서 파이썬 스크립트 실행

%prun statement: cProfile을 통해 statement를 실행하고 프로파일링 결과를 출력

%time statement: 단일 statement 실행 시간을 출력

%timeit statement: statement를 여러차례 실행한 후 평균 실행 시간을 출력. 매우 짧은 시간 안에 끝나는 코드의 시간을 측정할 때 유용

%who, %who_ls, %whos: 대화형 네임스페이스 내에서 정의된 변수를 다양한 방법으로 표시

%xdel variable: variable을 삭제하고 ipython 내부적으로 해당 객체에 대한 모든 참조를 제거

%%writefile filename: filename인 파일을 생성

4.ipython

In [10]: `import numpy as np`

In [11]: `a = np.random.randn(100,100)`

In [12]: `%time np.dot(a, a)`

Wall time: 710 ms

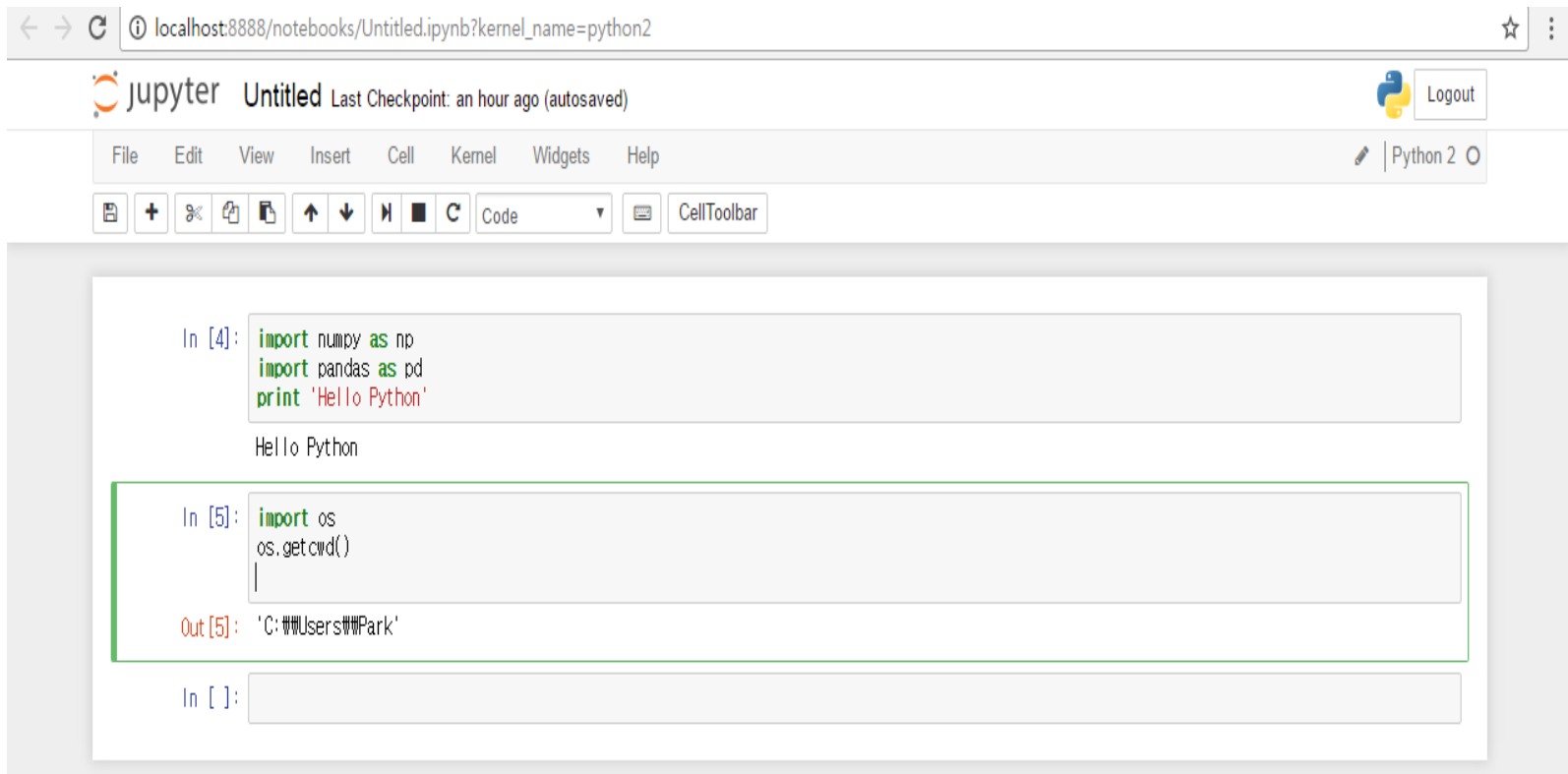
Out[12]:

```
array([[ 19.58800137,  2.62837613, -8.28386112, ..., -4.0982812 ,
        -13.99239531,  5.67487065],
       [ -5.55024439, 10.85218715,  6.14092077, ..., -13.73882312,
        -0.72756494,  2.39935928],
       [ -3.60873191, -15.89902742,  8.33705822, ..., 10.37198751,
        -2.66928403, -4.6852677 ],
       ...,
       [ -1.53904249, -1.2630197 , 25.4836311 , ...,  7.55673741,
         1.06642476,  3.21558587],
       [  8.5091311 , -14.67747325,  9.04744578, ..., -9.93647063,
         4.96999403, -8.40284447],
       [ -6.94798702, -5.71814232, 11.46917944, ...,  5.43738466,
        -8.7403922 , -6.61607408]])
```

5. jupyter notebook

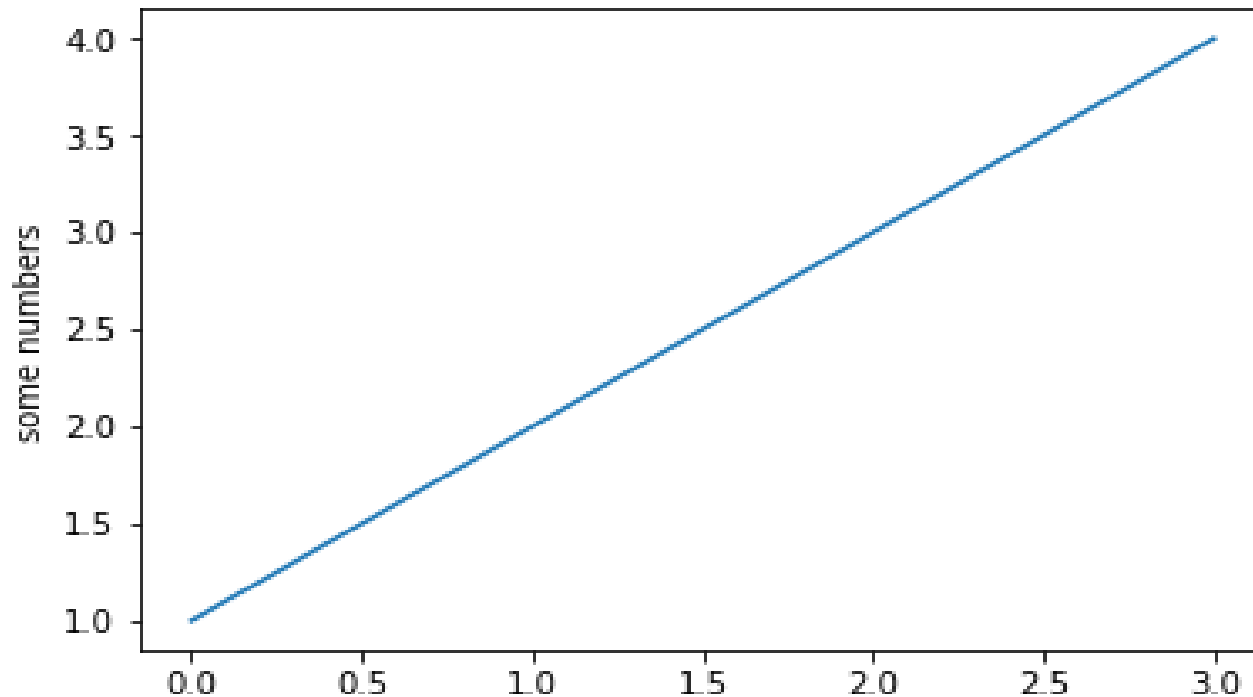
C:\Users\Administrator >> jupyter notebook

웹브라우저에서 <http://localhost:8888>



5. jupyter notebook

```
import matplotlib.pyplot as plt  
plt.plot([1,2,3,4])  
plt.ylabel('some numbers')  
plt.show()
```



6. pycharm

- ❖ Pycharm은 IntelliJ, CLion 등을 제공하는 JetBRAINS에서 배포하는 프로그램입니다.
- ❖ <http://www.jetbrains.com/pycharm/download/#section=windows> 에서 Community 버전을 선택해서 다운로드 후 설치

Download PyCharm

Windows

macOS

Linux

Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

Download

Free 30-day trial available

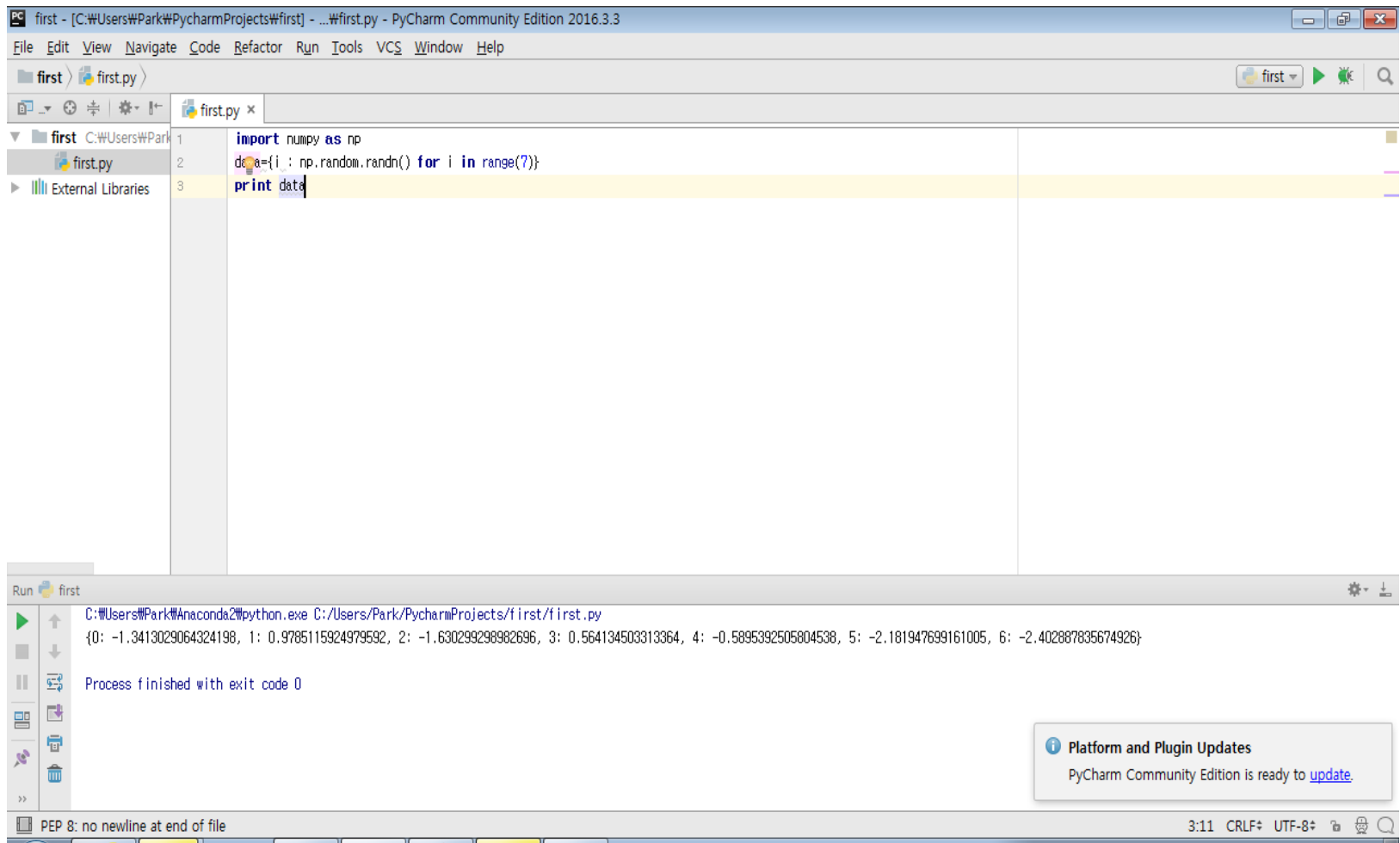
Community

For pure Python development

Download

Free, built on open-source

6. pycharm

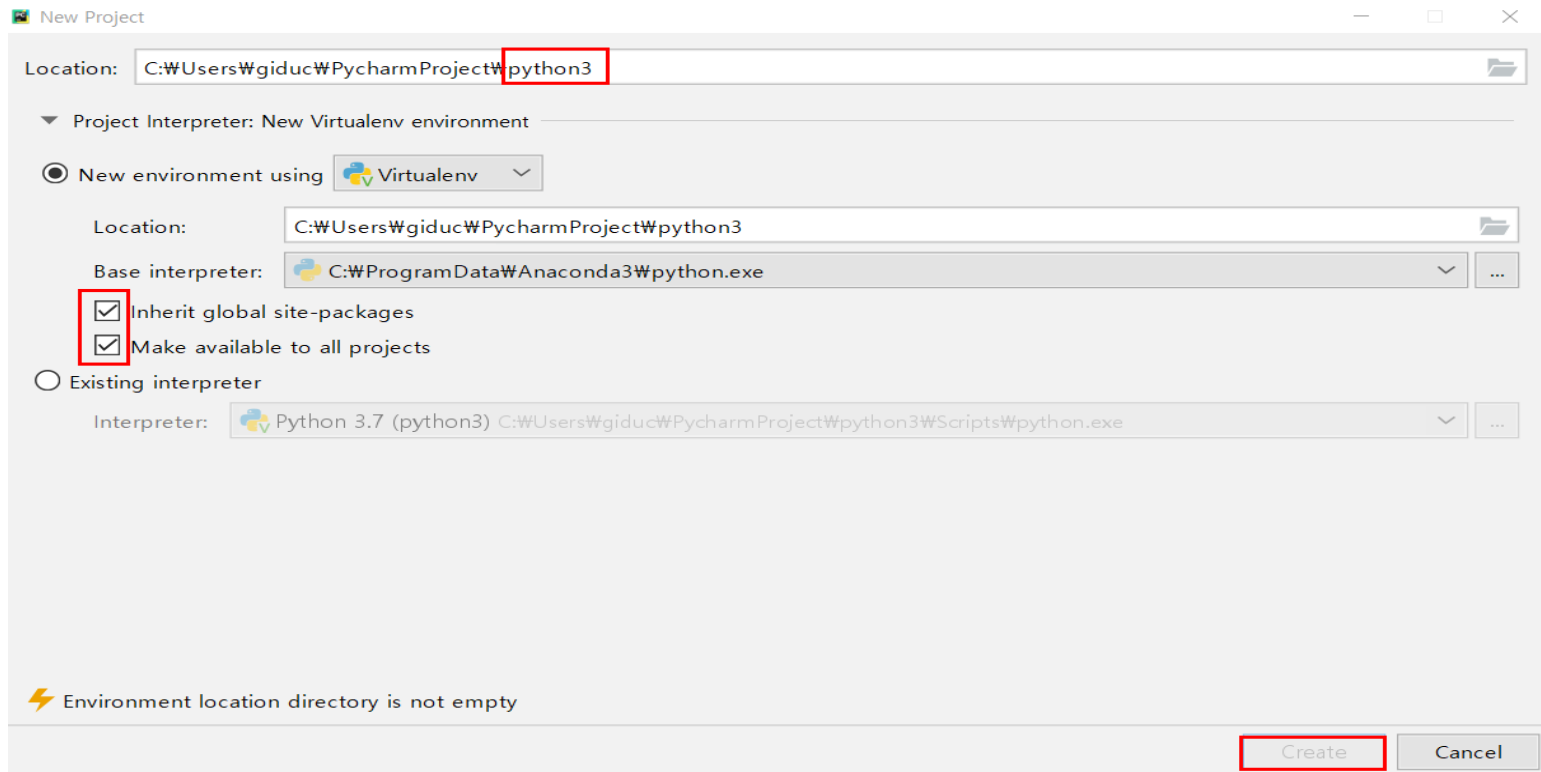


6. pycharm

- ❖ 가상 환경(Virtual Environment): 사용자가 정한 임의의 디렉토리 밑에 Python과 관련 패키지 등을 함께 넣어 그 안에서 독립적인 파이썬 개발 환경을 할 수 있도록 한 것으로 Lightweight, Self-contained 파이썬 개발 환경으로서 필요한 경우 한 개발 머신 안에 여러 개의 가상 환경을 만들고 각 가상 환경에서 다른 파이썬 버전이나 다양한 패키지들을 독립적으로 설치 사용할 수 있는 가상적 개발 환경
- ❖ Python 3 (3.4+)는 기본적으로 가상환경을 생성하는 유틸리티를 포함하고 있습니다.
- ❖ 가상 환경을 만들기 위해서는 pyenv 라는 유틸리티를 사용하는데, 각 OS 마다 (윈도우즈, Mac, 리눅스) 사용법이 약간씩 다릅니다.
- ❖ Pycharm에서는 pyenv 유틸리티가 번들로 포함되어 있으므로 가상환경 생성이 쉽습니다.

7. Pycharm 가상환경 구축

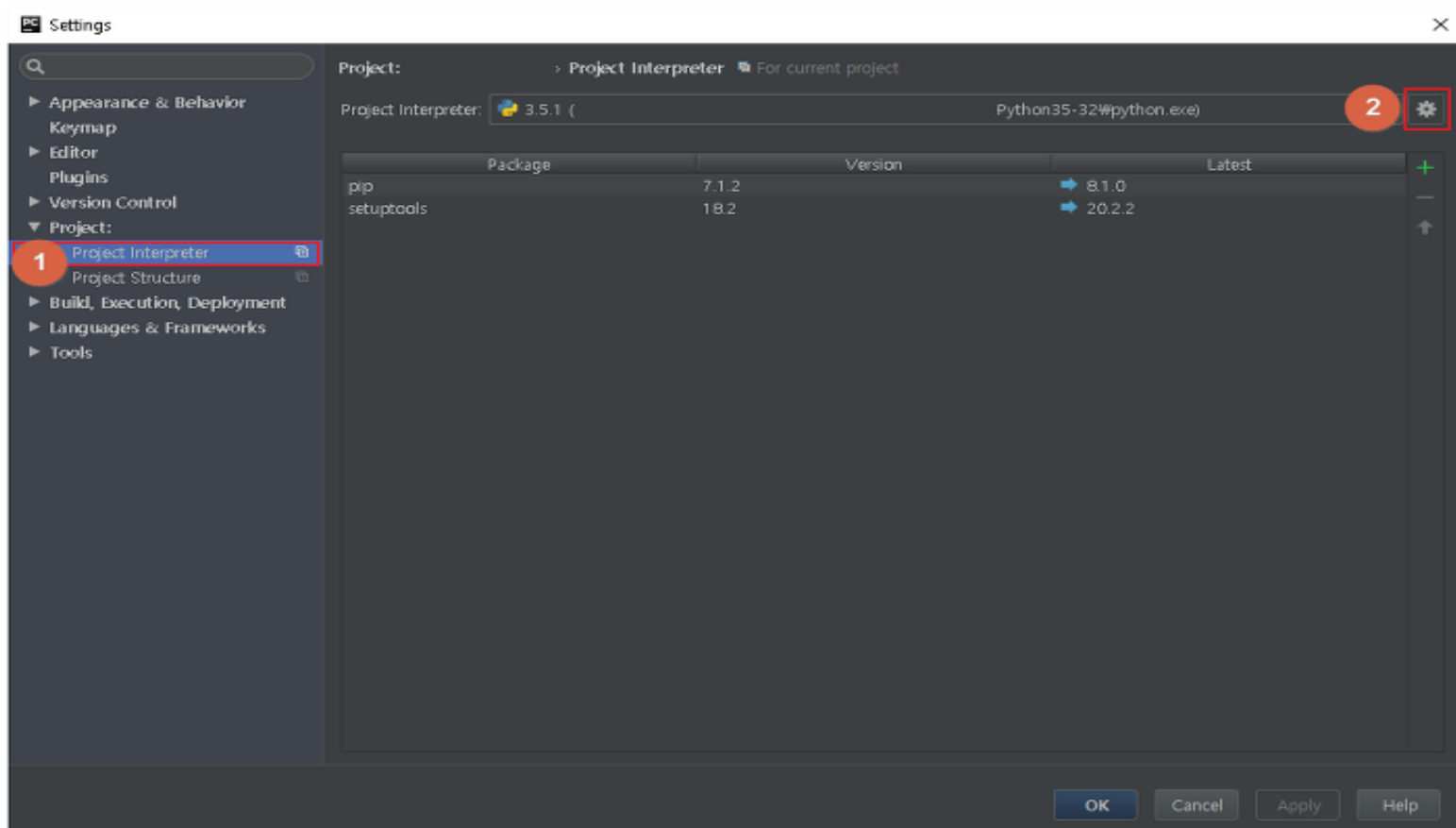
❖ 프로젝트 생성 [File] – [New Project]



- ✓ Inherit global site-packages: 기본 인터프리터의 site-packages를 상속받아 사용할 경우 체크
- ✓ Make available to all projects: 모든 프로젝트에서 사용할 경우 체크

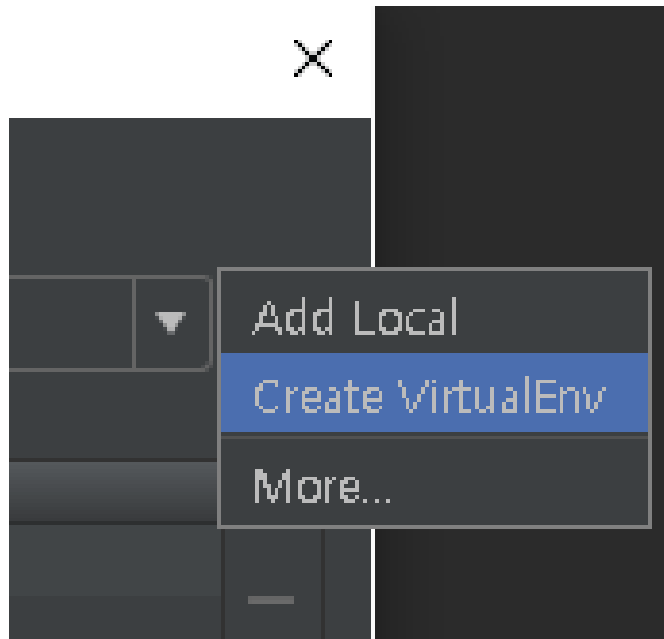
7. Pycharm 가상환경 구축

❖ 프로젝트 생성 [File] – [Settings]



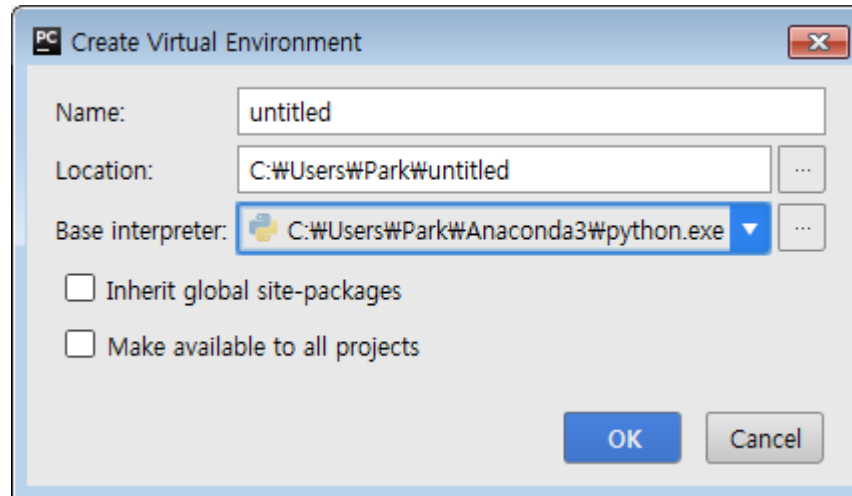
7. Pycharm 가상환경 구축

❖ Create VirtualEnv 선택



7. Pycharm 가상환경 구축

❖ 옵션 설정



- ✓ Name: 가상 환경 이름
- ✓ Location: 가상 환경이 생성될 경로
- ✓ Base interpreter: 사용할 파이썬 버전
- ✓ Inherit global site-packages: 기본 인터프리터의 site-packages를 상속받아 사용할 경우 체크
- ✓ Make available to all projects: 모든 프로젝트에서 사용할 경우 체크

8. Python Basic

❖ 파이썬과 다른 언어의 다른 차이점은 코딩 구조

- ✓ 구조적 프로그래밍 언어의 대부분은 특정한 부분을 구분 짓기 위해서 특정한 기호를 사용하여 그 시작과 끝을 표시하는 엄격한 구조를 따라야만 했는데 파이썬은 이러한 명령어가 없습니다.
- ✓ 파이썬에서는 코드의 구조를 정의하기 위해 기호 대신에 들여쓰기를 사용
- ✓ Java 나 C와 같은 언어에서 코드 블록을 여닫기 위하여 사용하는 중괄호 대신 파이썬에서는 공백을 사용하며 이는 코드를 읽기 쉽게하고 코드 내에서 불필요한 기호의 사용을 줄여줍니다.
- ✓ 코드의 정렬과 구성이 엄격하게 제한되는데, 네 칸를 들여쓰는 것이 표준이기는 해도 프로그래머가 들여쓰기의 규칙을 정할 수 있습니다.
- ✓ 하나의 문장을 종료할 때 종료기호는 없음
- ✓ 하나의 줄에 2개 이상의 명령어를 사용할 때 명령어를 구분하기 위한 용도로 ;을 사용

8. Python Basic

C 언어

```
#include <stdio.h>

int main()
{
    int i, sum=0;
    for(i=1; i<=10; i++)
    {
        sum+=i;
    }
    printf("%d", sum);
}
```

파이썬 언어

```
sum=0
for i in range(1,11):
    sum+=i
print(sum)
```

8. Python Basic

❖ 자바코드

```
int x = 100;
if(x > 0){
    System.out.println("x가 0보다 큼니다.");
} else {
    System.out.println("x가 0보다 작습니다.");
}
```

❖ 파이썬 코드(2.x에서는 #coding:utf-8을 상단에 추가, 기본: ascii code)

```
x = 100
if x > 0:
    print('x가 0보다 큼니다.')
else:
    print('x가 0보다 크지 않습니다.')
```

8. Python Basic

- ❖ 도움말을 실행하고자 하는 경우에는 [F1] 키를 눌러도 되고 IDLE 창에서 help(도움말을 얻고자 하는 명령)
- ❖ Python의 예약어
 - ✓ 예약어 (Reserved Words, keyword):파이썬에서 이미 문법적인 용도로 사용되고 있는 단어 또는 문자
 - ✓ 사용자 정의 식별자로 사용하면 예약어의 기능을 잃어버리는 단어들
 - ✓ 아래처럼 작성하면 예약어 확인 가능

```
import keyword
```

```
print(keyword.kwlist)
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def',  
'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is',  
'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with',  
'yield']
```

8. Python Basic

- ❖ 파이썬에서 주석을 달려면 다음과 같이 # 이런 샤프 기호를 사용하면 됩니다. 그러면 # 기호 뒤에 있는 모든 문자열들이 주석으로 무조건 무시됩니다.
- ❖ 어떤 특정 구역을 모두 주석화시키려면 """ 에서 """ 까지, 이렇게 외따옴표 또는 큰따옴표(쌍따옴표) 3개를 사용하면 됩니다.
- ❖ 소스 첫줄의 #! 기호는 주석문이 아닙니다. "유닉스 Shebang"입니다.
- ❖ 둘째 줄의 # 는 한글 확장완성형 인코딩 지정문입니다.
- ❖ 주석처리 단축키는 Ctrl + / 로 처리한다.

8. Python Basic

❖ 파이썬 코드

```
#!/usr/bin/python
# -*- coding: cp949 -*-
```

```
# 이줄은 라인 코멘트입니다
print("Hello World!")
```

```
print("Hello World!")      # 이것도 라인 코멘트입니다
print("Hello World!")      # 이것도 라인 코멘트입니다
print("Hello World!")      # 이것도 라인 코멘트입니다
```

```
"""
```

```
이것은 블럭 코멘트입니다.
그래서 여러 줄의 주석을
이렇게 달 수 있습니다.
큰따옴표 3개를 연속으로 적으면 됩니다.
```

```
"""
```

```
print("Hello World!")      # 라인 코멘트
```