Python

Installation of python in WSL or Ubuntu

Open a terminal, then ...

```
>> sudo apt install python3.8                 // need to specify the version, otherwise outdated
>> where python
/usr/bin/python                               // this is a symbolic link
>> cd /usr/bin; ll python*
/usr/bin/python
/usr/bin/python

>> python -m pip install numpy                // matrix calculation
>> python -m pip install scipy                // maths  calculation
>> python -m pip install sympy                // symbol calculation
>> python -m pip install matplotlib
>> python -m pip install pandas
>> python -m pip install scikit-learn
>> python -m pip install scikit-image
>> python -m pip install googlefinance        // 15 minute delayed tick-by-tick data
>> python -m pip install yahoo-finance        // daily historical data
>> python -m pip list                         // list all installed modules and corresponding version
```

Lets test it ...

```
>> python
>>> import numpy
>>> import scipy
>>> import sympy
>>> import matplotlib
>>> import pandas
>>> import sklearn                            // different in name (it does not support '-')
>>> import skimage                            // different in name (it does not support '-')
>>> import googlefinance
>>> import yahoo_finance                      // different in name (it does not support '-')
>>> exit()
```

Installation of Visual Code

However, when we run python code containing `matplotlib` in WSL will fail. Instead we need to do it via Visual Studio Code. Please note that we should install Visual Studio Code in windows, ***NOT*** in WSL. Here are the steps :

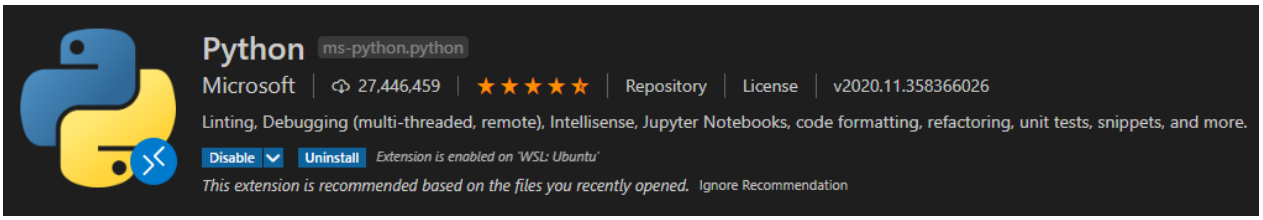
- download `VSCodeUserSetup-x64-1.51.1.exe` for windows
- execute `VSCodeUserSetup-x64-1.51.1.exe`, ensure that the `Add to PATH` option is selected
- open a `bash` terminal, go to the python code directory, type `code .` (don't miss the dot)

```
>> cd dev_python
>> code .                                     // vscode is launched with current directory as working directory
>> code                                       // vscode is launched without working directory
```

Step 1

Press `Ctrl-Shift-x` to launch *Extensions*, search and install the following :

- `Python` extension (although we already have Python interpreter in the machine, we need to install `Python` extension)
- `C++` extension (although we already have `gcc` compiler in the machine, we need to install `C++` extension)
- `Remote-WSL` extension (for remote access to *WSL*)
- `Remote-ssh` extension (for remote access to *Yubo office machine*)
- `vim` emulator

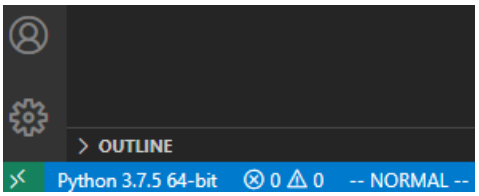


Now with `vscode` in windows, we can either :

- local access python in windows or
- remote access python in `WSL`

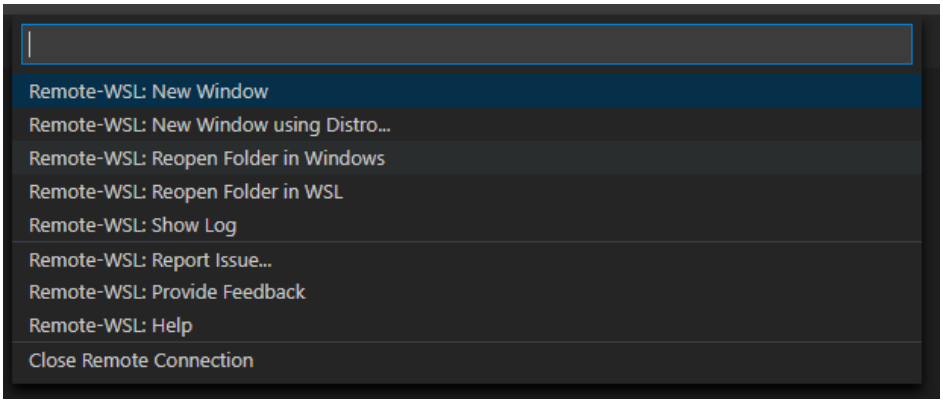
We can easily switch between the windows environment and `WSL` environment for development of the same Python project.

Step 2
Goto *LL* corner of *VSCode*, click the `><` icon (or press `F1` to invoke remote-explorer). It is using Windows Python, version `3.7.5`.

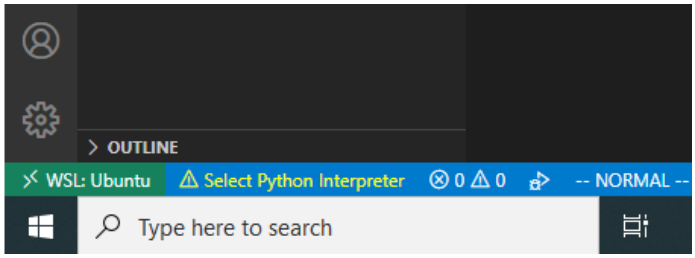


then in the pop-up menu, select either :

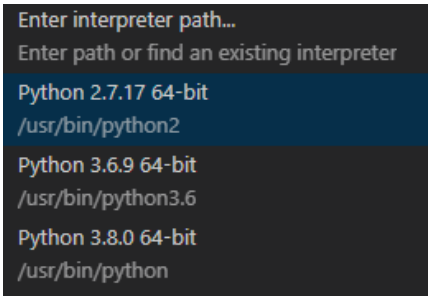
- `Remote-WSL : Reopen Folder in Windows`
- `Remote-WSL : Reopen Folder in WSL`
- `Remote-SSH : Reopen Folder to SSH remoted machine`



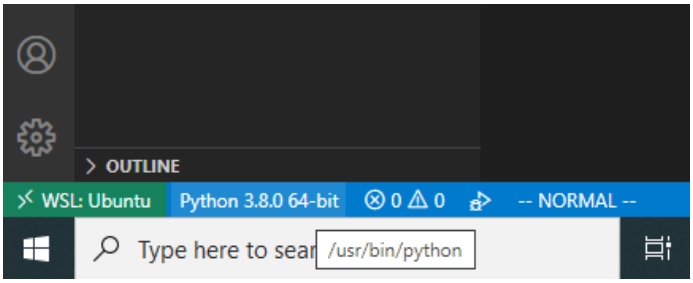
Suppose we select *WSL*, then *LL* corner becomes `>< WSL:Ubuntu` :



Now the Python interpreter is unknown, then we click `Select Python Interpreter` icon, menu pops up, showing that multiple Python versions can be found under `/usr/bin/python*` (`vscode` ables to locate all of them), select the one we want ...



finally it becomes ...



Then click run to execute the python code. Open one of the python file, then run it using either one of the following methods :

- click *Start-Run*, and select *Debug the currently active Python file*
- right click, select *Run file in terminal*
- right click, select *Run file in interactive mode* (in this mode, we can type extra code and run by `Shift-Enter`)
- ► we can set breakpoint (red circle on *LHS* of code)

Please note that if `matplotlib` is used to plot graph, only windows version can display it successfully, but ***NOT*** WSL. In other words, if no display is shown in terminal, no display can be shown in `vscode` remote mode. We can repeat the above steps for C++.

Setup for C++

How to open a new project? First of all, install `cmake tools` (not `cmake`) :

- press `ctrl-shift-x` (invoke `vscode` exchange) and install `cmake tools`
- press `ctrl-shift-p` (invoke prompt-command-line) and enter `cmake:quick start`
- enter project name
- enter project type : is it an executable or a library
- it then generates root `CMakeLists.txt`
- press `ctrl-shift-p` and enter `cmake:select a kit`, select from a list of `gcc` compilers
- press `ctrl-shift-p` and enter `cmake:select variant`, select from a list of build : `debug` / `release` / `release with debug info` etc
- press `ctrl-shift-p` and enter `cmake:config`, it generates `build/Makefile`
- press `ctrl-shift-p` and enter `cmake:build`, it generates `build/my_executable`
- press `ctrl-shift-p` and enter `cmake:debug`, it executes `build/my_executable`

The above is slightly different from my practice in `nvim`, I usually have :

```
>> mkdir build;   cd build
>> mkdir debug;   cd debug;   cmake -DCMAKE_BUILD_TYPE=Debug   ../..; make
>> mkdir release; cd release; cmake -DCMAKE_BUILD_TYPE=Release ../..; make
>> cd ../..
>> ./build/debug/my_executable
>> ./build/release/my_executable
```

where makefiles can be found under `build/debug/Makefile` and `build/release/Makefile`.