

Contents

- Contents
- Set up development environment
 - Terminal
 - * Git Bash
 - * Zsh
 - * Oh My Zsh
 - * Powerlevel10k theme
 - Python3 deployment
 - * Python3 installation
 - * Python3 virtual environment
 - Using conda
 - Using pip/pip3
 - Third-party libraries
 - Select python interpreter

Set up development environment

To start, here we first say little about different platforms:

1. Linux
2. MacOS
3. Windows

Linux and MacOS are platforms developers favor, for their Unix-based operating frameworks. Considering the usage habits of everyone, and the future usage of `cuda`, to be novice friendly, we set up our development environments on the most complicated platform for development – Windows.

Terminal

In Windows 10, you can download the app **Terminal Preview** from the Microsoft Store to use as your terminal. However, in Windows 11, you can use the default app **Terminal**.

In the top menu bar, click on the sign `V` next to the plus sign `+`, and you can find three shells: **Windows PowerShell**, **Command Prompt**, and **Azure Cloud Shell**.

Next let's go to another shell: `git bash`, which can be added into **Terminal**. With it, you could have almost the same experience as in Linux and MacOS.

Git Bash

Go to the website <https://git-scm.com/download/win>, and select the version **64-bit Git for Windows Setup**. During installation, select **Add a Git Bash Profile to Windows Terminal**; then you can use `git bash` in **Terminal**.

Now you can see another shell `git bash` under the above three shells. Click on the **Settings**, and in **Startup** select `git bash` as the default shell. Then you can see the `git bash` shell when you open **Terminal**.

Zsh

Next we go to a very advanced and programmable command interpreter (shell) for UNIX: `zsh`.

Go to the download link https://mirror.msys2.org/msys/x86_64/zsh-5.9-2-x86_64.pkg.tar.zst and gunzip the file. Then copy the two folders and four files to the path of your Git repository `C:\Git`. If prompted for permissions, grant them. In case of conflicts with duplicate names, simply overwrite the existing files.

First we digress a little to say where you are when you open **Terminal**. You can use the command `pwd` to see the current directory. In Windows, the default directory is `C:\Users\yourname`. And this time I highly recommend this powerful editor VScode to you. You can download it from the official website, quickly install it. Then you can open `.bashrc` file by VScode, and fill in the following content:

```
if [ -t 1 ]; then
    exec zsh
fi
```

Then you can see the **zsh** shell when you reopen **Terminal**. You may encounter warnings such as “`bash_profile` not found” or similar files. You can safely ignore them, and they should not appear again in the future.

Oh My Zsh

After installing **zsh** terminal, it may appear similar to the **bash** terminal without any significant differences, as we haven't made any configurations. However, **Oh My Zsh** can be used to manage **zsh** configurations. It bundles thousands of useful features, helpers, plugins, themes, and more.

You could install **Oh My Zsh** by running the following command in **zsh** by `curl`

```
sh -c "$(curl -fsSL https://gitee.com/mirrors/oh-my-zsh/raw/master/tools/install.sh)"
```

or by `wget`

```
sh -c "$(wget -O- https://gitee.com/pocmon/mirrors/raw/master/tools/install.sh)"
```

After installation, the default theme used is “`robbyrussell`”. You can modify the `ZSH_THEME` field in the `.zshrc` configuration file. Next, you need to install two plugins: **zsh-autosuggestions** and **zsh-syntax-highlighting**:

```
git clone https://github.com/zsh-users/zsh-autosuggestions
${ZSH_CUSTOM:--/.oh-my-zsh/custom}/plugins/zsh-autosuggestions
```

and

```
git clone https://github.com/zsh-users/zsh-syntax-highlighting.git
${ZSH_CUSTOM:--/.oh-my-zsh/custom}/plugins/zsh-syntax-highlighting
```

The **zsh-autosuggestions** plugin allows you to find and highlight history commands that match your current input, and you can use the right arrow key to complete them directly. The **zsh-syntax-highlighting** plugin recognizes shell commands and highlights them.

Then you can open `.zshrc` file by VScode, and fill in the following text near line 80:

```
plugins=(git
          zsh-syntax-highlighting
          zsh-autosuggestions)
```

Powerlevel10k theme

The default theme used is “`robbyrussell`”, which is not very beautiful. We can use the “`Powerlevel10k`” theme to make it more beautiful. To use the theme, font **MesloLGS NF** is recommended. After installation, it will be available to all applications on your system. Make sure that **Terminal** and integrated terminal

in VScode are using the same font. Then you can open `.zshrc` file by VScode, and fill in the following text near line 20:

```
ZSH_THEME="powerlevel10k/powerlevel10k"
```

After that, interactive information for `p10k` configuration will be displayed. Enter `y` and configure it according to your preferences. If you wish to reconfigure it in the future, you can execute the command

```
p10k configure
```

Here, I will use an image to demonstrate the successful configuration on all three platforms; see Figure 1.



Figure 1: Successful configuration of the final prompt if you have the same taste as Ang. Platforms from top to bottom: Linux(Ubuntu), MacOS, Windows.

Python3 deployment

Python3 installation

Here we talk about how to install `python3` on Windows. It's easy to install `python3` from the official website. But the command in the terminal is initially “python”. If you want to change it to “python3”, you should do as follows:

Use the following command to find the installation path of `python3`,

```
import sys
sys.executable
```

Once you have located the `python3` installation path, navigate to that directory, and rename the `python.exe` file to `python3.exe`, with the `pythonw.exe` file to `pythonw3.exe`.

After renaming, the `pip` or `pip3` command may encounter errors, and one potential solution is

```
> python3 -m pip install --upgrade --force-reinstall pip
```

Python3 virtual environment

We highly recommend you to have `venv` named `yourenv` (decided by you) available, since it's terse for you to manage those python packages. There are two methods to meet such needs: `conda` and `pip/pip3`.

Using conda We recommend Anaconda, using the free distribution installation. And then you can make a conda environment in the terminal:

```
(path of current directory).....(hh:mm:ss)
> conda create -n yourenv python=3.x
```

Then you can find the Conda Env `yourenv` to run both `*.py` and `*.ipynb` files in VScode. In terminal or integrated terminal, you can activate by

```
(path of current directory).....(hh:mm:ss)
> conda activate yourenv
```

and deactivate by

```
(path of current directory).....(yourenv) (hh:mm:ss)
> conda deactivate
```

But conda environments come with a set of pre-installed Python libraries, which can sometimes conflict with newly installed libraries. Thus we recommend the following method.

Using pip/pip3 Using pip/pip3 to make a virtual environment with:

```
(path of current directory).....(hh:mm:ss)
> python3 -m venv yourenv
```

In terminal or integrated terminal on Windows, you can activate by

```
(path of current directory).....(hh:mm:ss)
> source [path of yourenv]/yourenv/Scripts/activate
```

If can't, run the terminal as an administrator

```
(path of current directory).....(hh:mm:ss)
> set-ExecutionPolicy RemoteSigned
```

then Enter, and chose Y to confirm. And you can deactivate the venv by

```
(path of current directory).....(yourenv) (hh:mm:ss)
> deactivate
```

Third-party libraries When first deploying a venv, there're two libraries, and you can look them up with

```
(path of current directory).....(yourenv) (hh:mm:ss)
> pip3 list
```

and you can see

Package	Version
-----	-----
pip	22.0.4
setuptools	58.1.0

Here pip and setuptools are built-in with yourenv. If you want to install other third-party libraries like numpy, you can install them with

```
(path of current directory).....(yourenv) (hh:mm:ss)
> pip3 install numpy
```

or

```
(path of current directory).....(yourenv) (hh:mm:ss)
> pip3 install numpy -i https://pypi.tuna.tsinghua.edu.cn/simple/
```

The option -i is used to specify the index source from where the package will be installed. Here we can use the index source from tuna, Tsinghua University open source software mirror. By default, pip3 downloads packages from the Python Package Index (PyPI). Then you can see

Package	Version
-----	-----
numpy	1.25.0
pip	22.0.4
setuptools	58.1.0

To export the dependencies used in `yourenv` to a `requirements.txt` file, you can use the following command:

```
(path of current directory).....(yourenv) (hh:mm:ss)
> pip3 freeze -> requirements.txt
```

If you want to transfer `yourenv` to your code collaborators, just send them the `requirements.txt` file and tell them to use the command:

```
(path of current directory).....(hisenv) (hh:mm:ss)
> pip3 install -r requirements.txt -i https://pypi.tuna.tsinghua.edu.cn/simple/
```

Select python interpreter With VScode, we usually deal with two kinds of python files: `.py` file and `.ipynb` file.

When opening a `.py` file, you can

- Enter `shift+control+P` to open Command Palette
- Click Python: Select Interpreter and then + Enter interpreter path...
- Input [python3 path of yourenv]

Then you can enter the button ►, and run the `.py` file. After doing so, you can open a `.ipynb` file, and

- Click Select Kernel
- Select Select Another Kernel... and Python Environments...
- Choose yourenv

With this, the configuration process for Python starting from the Terminal is complete. Arm yourself with Python and embark on a wonderful journey!