Installing WSL and VSCode

Ethan Bull - ethan.bull@hdr.mq.edu.au

December 2025

1 WSL and Windows

WSL (Windows Subsystem for Linux) allows you to run many Unix based programs natively on windows by running a Linux subsystem on top of your Windows install. This is especially important when dealing with code written and/or compiled in Unix, as many python packages interact slightly differently between Unix and Windows installs.

1.1 Installation

WSL has become very easy to install, especially on recent updates of Windows 10, and Windows 11. The most simple way to install WSL is to run

```
wsl --install
```

in a PowerShell terminal ran in administrator mode. By default this will install WSL2, which is what we want.

The easiest Linux distribution to use with WSL is Ubuntu. To install Ubuntu to your WSL, run

```
wsl.exe --install Ubuntu
```

If this doesn't work, run

```
wsl.exe --list --online
```

to check the distro names, and replace Ubuntu with your desired distro.

Once installed, you will be prompted to set a User Name and Password. These do not need to be the same as your Windows username and password. DO NOT put a space in the username. This account will become the default administrator account, and will be used to run the `sudo` command, so make sure to set something you can type and you'll remember.

1.2 Usage

To run WSL, you can either search for "Linux", or "Ubuntu" in the windows search bar, or run `wsl` or `wsl.exe` in the windows terminal.

To close WSL, run

```
exit
```

to close the terminal and subsystem safely.

When using WSL, you will install all programs as you would if you were running a complete Linux install, usually through the command line. To access your Linux files through the Windows File Explorer, you can either run

```
explorer.exe .
```

in your Linux command line, or enter \\`wsl` as the file location in the Windows File Explorer. The only exception to this whole process is installing and using VSCode.

For more information, see *https://learn.microsoft.com/en-us/windows/wsl/install*

2 VSCode and Python (in WSL)

2.1 Installation

Installing VSCode is very straight forward: Just download the VSCode installer for Windows online or through the Windows store, and run the installation.

Launch VSCode through Windows as you would a normal application, and navigate to the extensions tab on the left. Install the WSL extension.

From here, there are multiple ways to launch VSCode in WSL. You can run

```
code .
```

in the Linux terminal, or you can launch VSCode as normal in windows, then click the ⟩⟨-looking symbol in the bottom left corner to open the Remote Window dialogue. This dialogue can also be opened from a new window using the “Connect to...” option in the Start menu. From there, select WSL. If you’ve had a workspace open in WSL previously, you can also open that from recent workspaces and VSCode will connect to WSL.

To run python in VSCode, you must have both python installed on your WSL distro and the python extension. I also highly recommend installing the Jupyter Notebook and ipykernel extensions to run Jupyter Notebook files too.

2.1.1 Installing Python with Miniconda

To install miniconda, run

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

to download the installer and then

```
bash  /Miniconda3-latest-Linux-x86_64.sh
```

to run it. Agree to the EULA and to the initialization, and let it install. When completed, run

```
source ~/.bashrc
```

to allow the system to recognise the install. Python is now installed, and you can run python code, including pip installations.

If you want to install a specific version of Python (I recommend 3.13 or 3.14), or you want to have multiple independent Python environments, you can make a conda environment. To make a conda environment, first make sure you aren’t in a current conda environment by running

```
conda deactivate
```

in a terminal. Then, run

```
conda create --name environment_name python=3.xx
```

where environment_name is a name of your choice (no spaces), and 3.xx is the Python version you’d like to install. To use the environment in the terminal, run

```
conda activate environment_name
```

To get VSCode to use this environment in a .py file, select the current Python kernel in the bottom right of VSCode, and then select your desired environment. To update a Jupyter Notebook’s environment, select the kernel in the top right of the Jupyter notebook file when opened in VSCode.

2.2 Usage

To open any files in VSCode, you must first open a workspace. The easiest way to do this is to simply open a folder, which you can do from the Start menu in a new VSCode window. You can also do this using File → Open Folder. If you have a saved workspace or previously opened workspace, you can open this from File → Open recent or Open workspace from file.