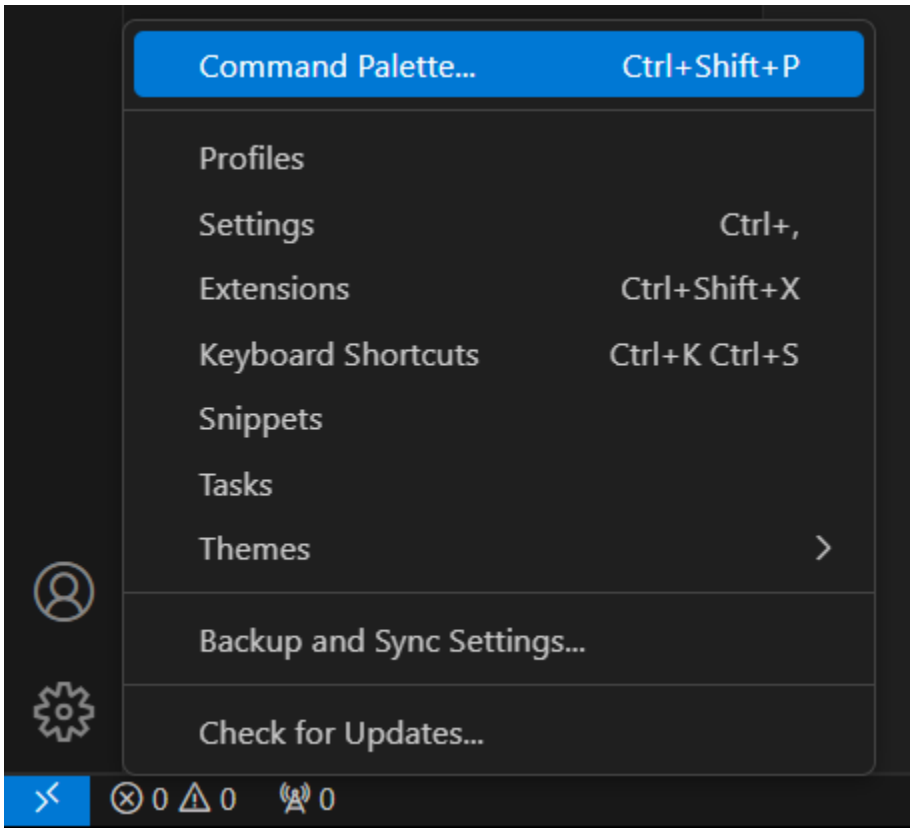
Virtual Environments: Cheat Sheet

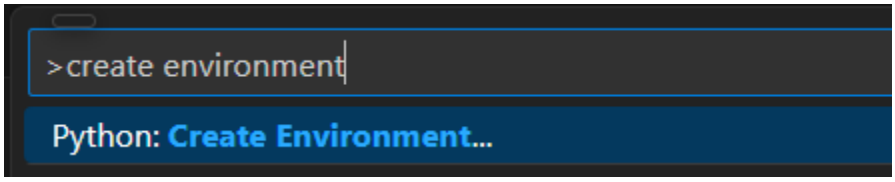
Table of contents

Creating a virtual environment

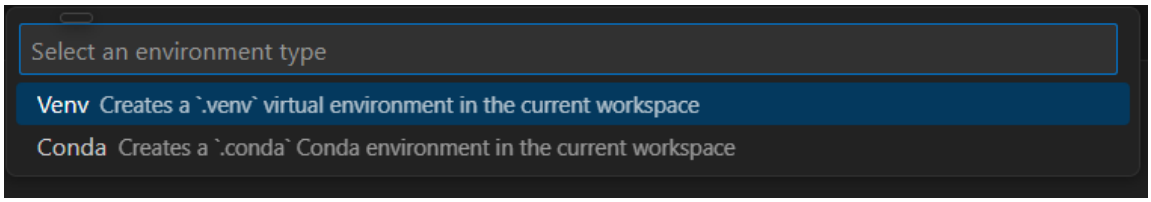
1. In your project on VS Code, open the command palette using the cog icon in the left bottom corner or F1.



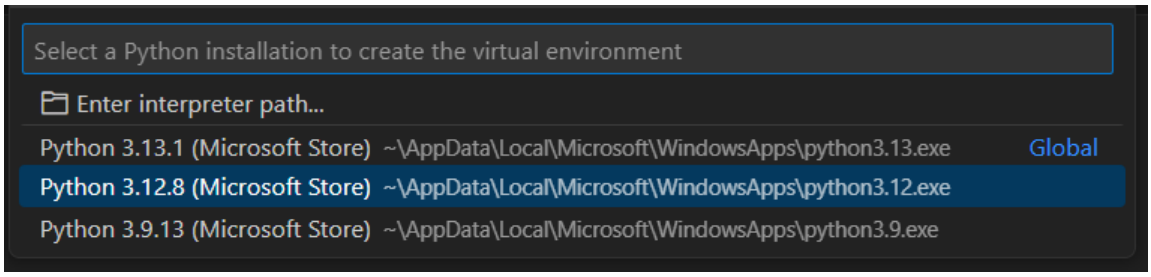
2. Type Create environment and select **Python: Create environment…**



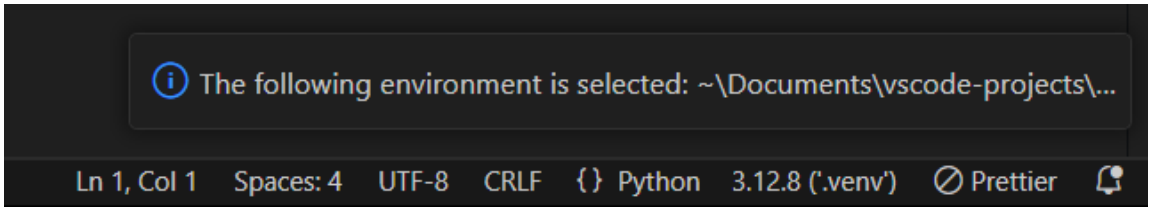
3. Select **Venv** as the environment type.



4. Select the Python version you want to use for this project. Unless otherwise specified, that is version **3.12.x** for our lessons.



5. Create a Python file in your project. On the right-hand side of the footer, you should now see that the environment is activated and which Python version it uses.



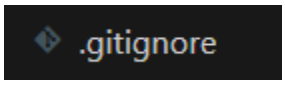
If this does not show automatically, please check the **Troubleshooting** steps below.

6. Open a terminal and check your Python version by running **python3 --version**. Depending on your system, this may not show the correct Python version you selected for the environment. If this is the case, run **python --version** instead.

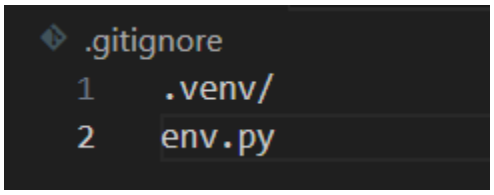
Note which version (**python3** or **python**) shows the correct Python version; you'll need

this any time you run a Python command in the terminal.

7. We do not need to add the .venv folder to GitHub. To tell **git** to ignore it, create a file called **.gitignore** - make sure to include the dot in that filename.



8. In **.gitignore**, add the **.venv/** folder (remember the / at the end that tells git it is a folder) and the file name **env.py**.

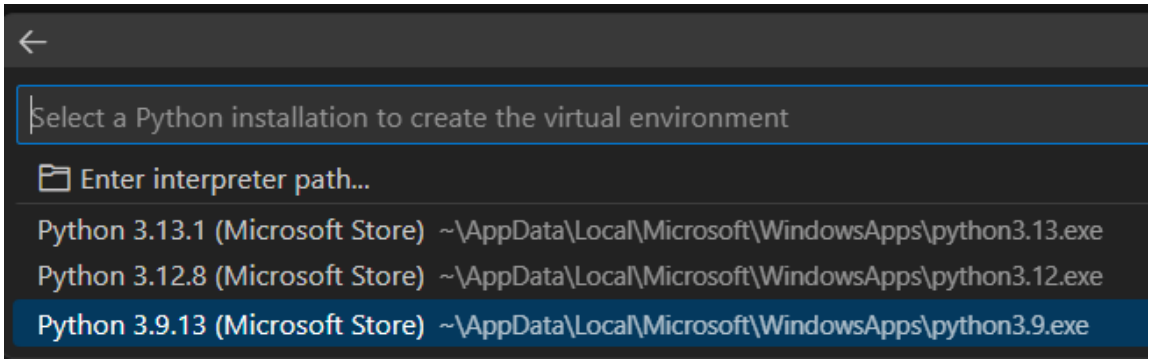


9. Well done! Your virtual environment is now up and running.

Installing a different Python version

1. Download the version you need onto your system from https://www.python.org/downloads/.

2. Create a new virtual environment in your project. Select the newly installed Python version to set it up, e.g. Python 3.9 in the screenshot below.



3. Check that the terminal was created using the correct Python version.

Using pip to manage packages

Installing packages

To install packages with pip, we have several commands at our disposal depending on which version we want to install in our projects.

Command	Use case
pip install package_name	Install the **latest** version of the package/library
pip install package_name**==**8.12.8	Install a **specific** version of the package; in this example, version number 8.12.8
pip install package_name**~=**8.12	Install any version that is **compatible** with the given version number
pip install -r requirements.txt	Install all packages listed in the **requirements.txt** file

Pip freeze uses

Command	What does it do?	When to use it
pip freeze	Lists the packages that are installed and their version numbers in the terminal.	Check whether your environment is active and can access the packages you've previously installed.
pip freeze > requirements.txt	Adds the list of installed packages and their version numbers to a file called requirements.txt	Reinstall your packages after starting a new workspace or virtual environment. Runs behind the scenes on deployment.

Troubleshooting FAQ

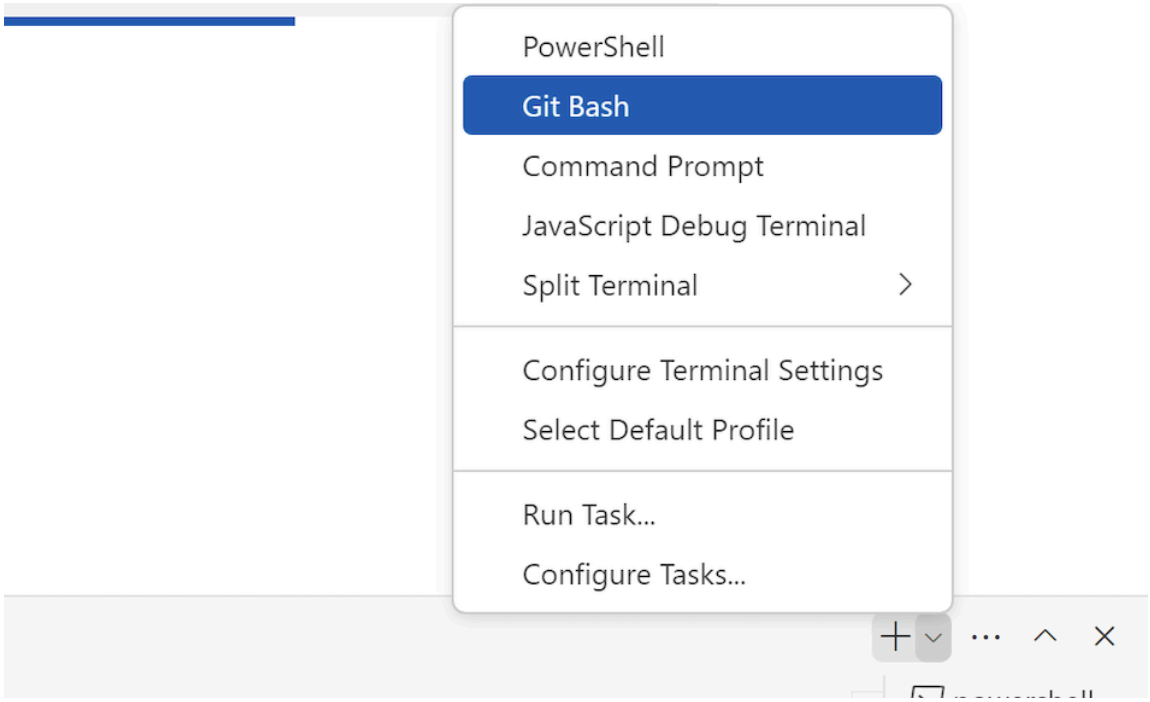
If VS Code fails to reactivate your virtual environment, you can do it manually.

1. Open a terminal and make sure you're in your project folder.
2. On Windows type: **.\.venv\Scripts\activate**
3. On Mac and Linux type: **source .venv/bin/activate**

NOTE: On some Windows systems, the **Activate** script will fail with the error "running scripts is disabled on this system".

If that happens, follow the steps below:

1. Click the dropdown arrow (caret) next to the plus icon at the top right of the terminal window and select **Git Bash** from the menu.



2. In the new terminal, make sure you're in the project directory. Then type: **source .venv/Scripts/activate**

You will now be able to open or create a Python file and see the ('venv': venv) indication in the status bar, which indicates that the virtual environment is activated.