



CODE BASED
LEARNING

PYTHON-DOCS

- IDE CONFIGURATIONS -

PROF. DR. RER. NAT.
ALEXANDER Voß

INFORM-PROFESSUR
FH AACHEN



CODE BASED
LEARNING

IDE CONFIGURATIONS

- PYTHON INTERPRETERS -

PyCHARM

DOCKER

VISUAL STUDIO CODE

Open python_course in PyCharm

There is no special project file, so open the `python_course` directory with PyCharm or choose `Open...` from the menu, if PyCharm is already open.

Assuming that there are global Python versions available you can select one at the bottom of the window.

Open `python_course` with PyCharm

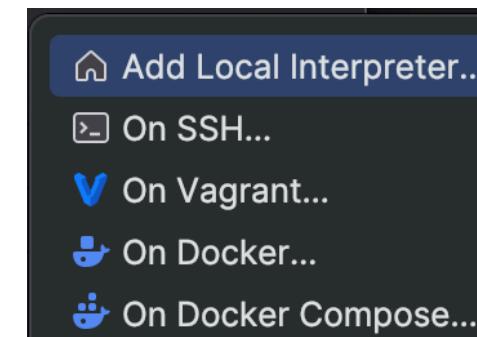
The screenshot shows the PyCharm interface with the title bar "PC python_course" and "main". The left sidebar displays a project tree with "Project Files" expanded, showing directories like `~/WorkInProgress/code_based`, `0x00`, and `0x01`, and files like `hello_world.py`. The main editor window contains a Python script with the following code:

```
# (C) 2025 A.Voß, a.voss@fh-aachen.de, https://a.voss.name
import platform
print(f"Hello World! (python {platform.python_version()})")
```

A context menu is open over the code, with the "Python Interpreter" option highlighted. The menu also includes "Add New Interpreter", "Interpreter Settings...", and "Manage Packages...". A blue arrow points to the "Python 3.9" option in the menu.

Configure Python in PyCharm

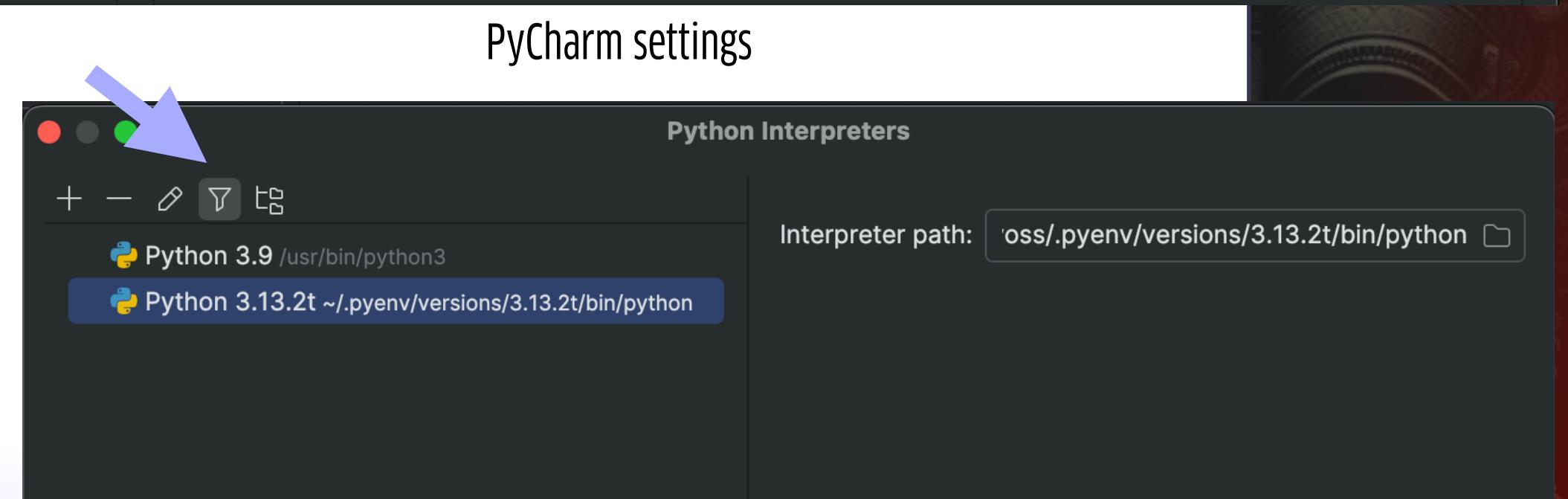
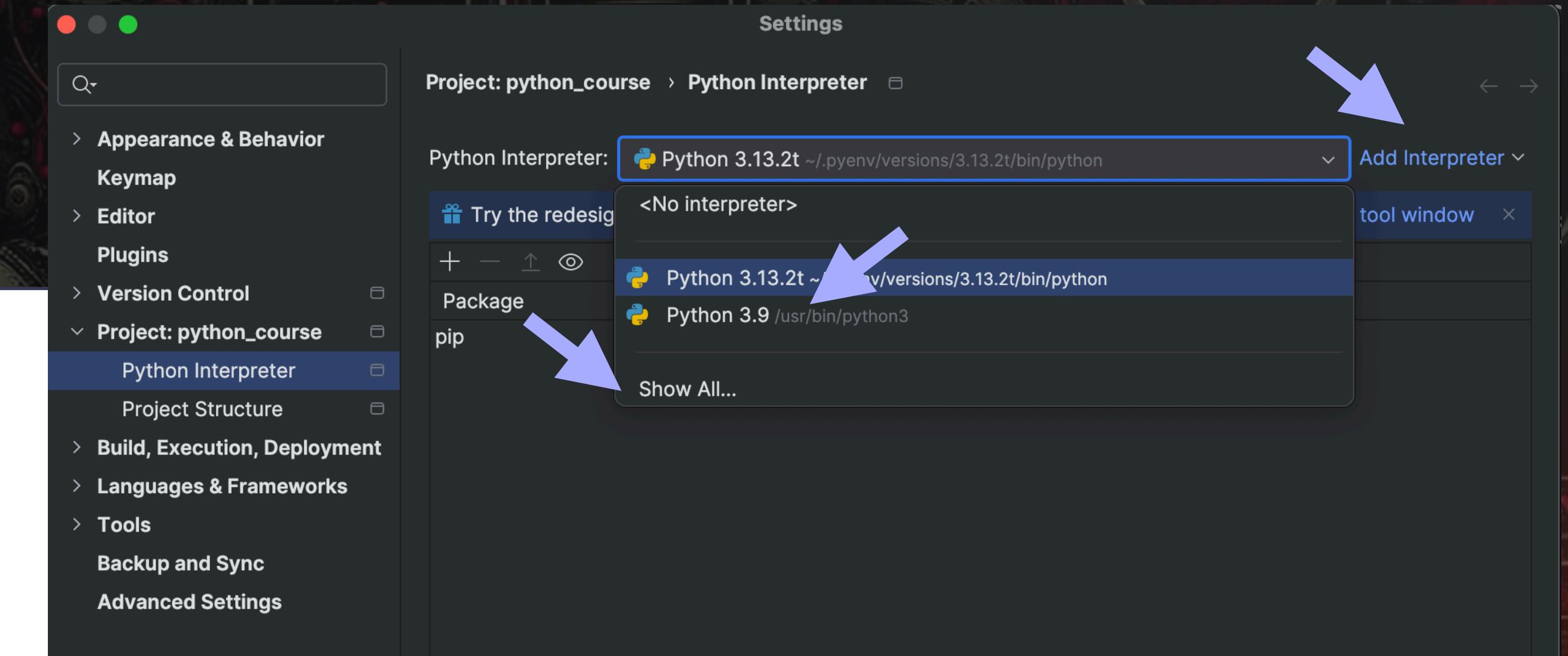
In PyCharm settings, you can also select the Python version/interpreter to be used or manage all interpreters via Show All... or Add Interpreter.



When you add a *local interpreter*, you can choose to select an existing interpreter, e.g. from any virtual environment (venv), or create a new virtual environment.

This gives you great flexibility to organise your versions and packages according to your needs and projects.

Note: Make sure to remove the *filter* if you want to see interpreters associated to other projects as well. You can also rename an entry here.



Configure Docker in PyCharm

Having introduced the interpreter setting in PyCharm, we would like to complete our discussion of Python environments with the remaining, the *Docker* variant.

To run our scripts in Docker, we will first create a *Docker image* with a specific Python version, including our `cbl` module.

All the files we need are in the `docker` folder, namely `Dockerfile` and the `requirements.txt`.

The latter contains the `cbl` module (see `pip freeze`).

```
# (C) 2024 A.Voß, a.voss@fh-aachen.de, python@codebasedLearning.dev

FROM python:3.12.2

# Create user to prevent installation of dependencies at root level.
RUN useradd -m pythonista

WORKDIR /home/pythonista/app
ADD . /home/pythonista/app
USER pythonista

# install dependencies from requirements
RUN pip install --no-cache-dir --user -r requirements.txt
```

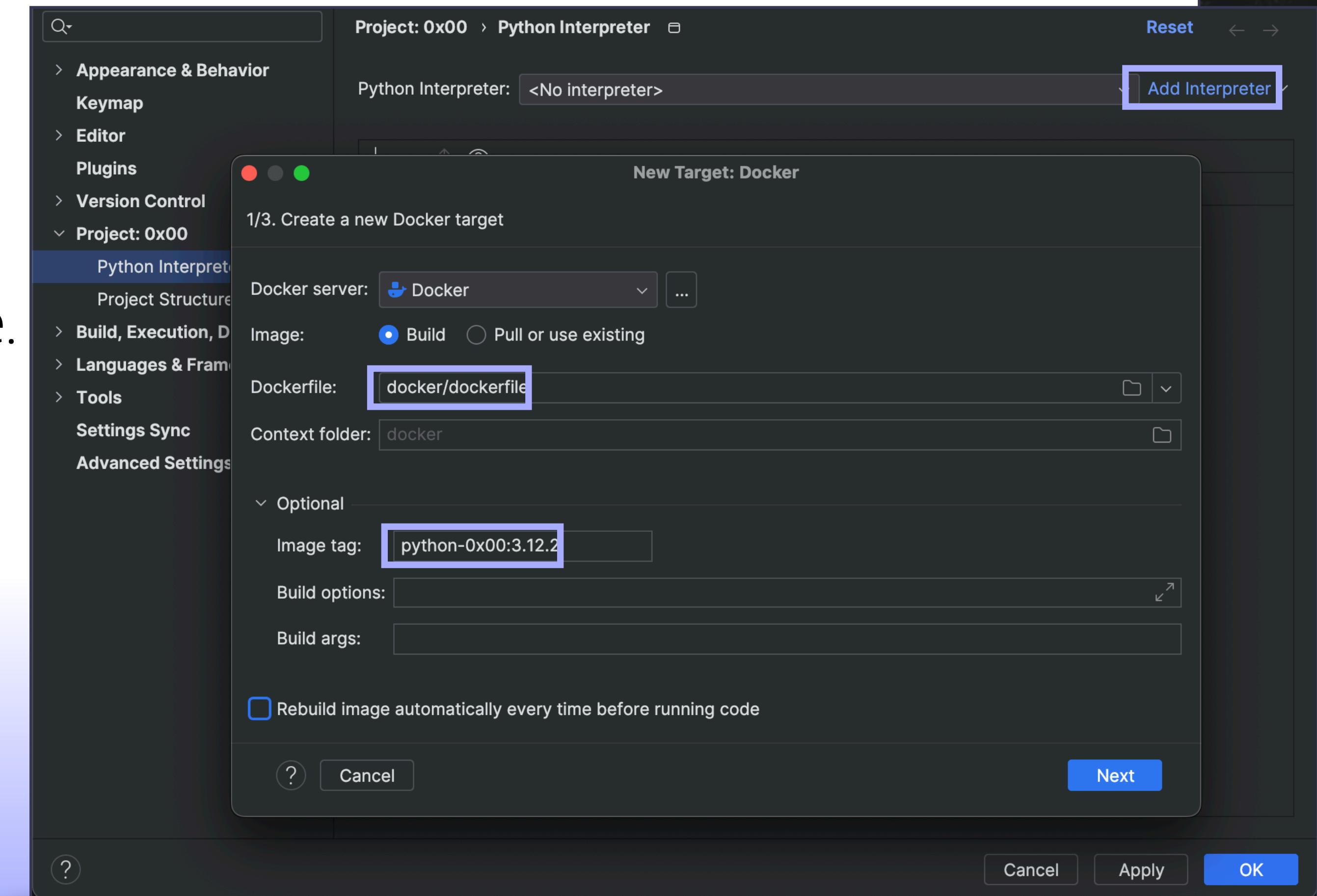
Dockerfile (or similar)



Configure Docker in PyCharm

IDE Configurations

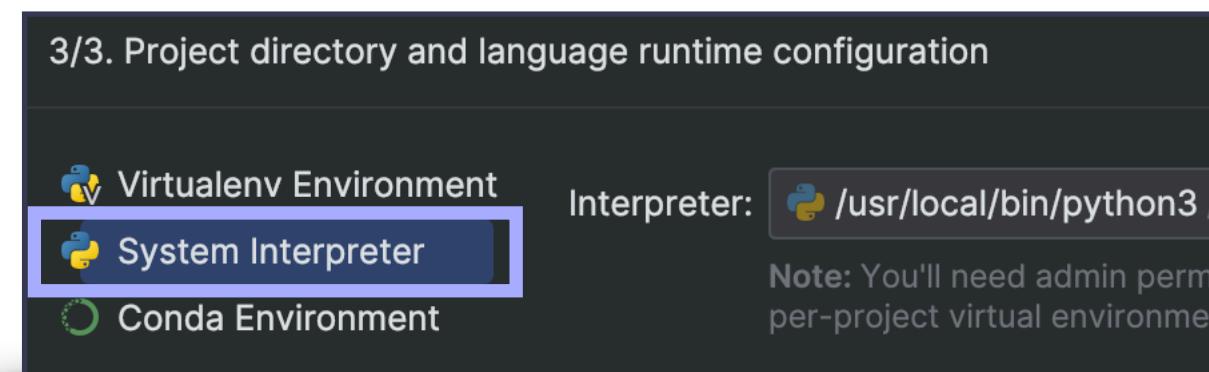
In the PyCharm settings, we Add Interpreter and select On Docker. Then we have to specify the dockerfile (docker folder) and supply a tag to label the image. After that the image will be built and Python and the modules from requirements.txt will be installed.



PyCharm settings

Configure Docker in PyCharm

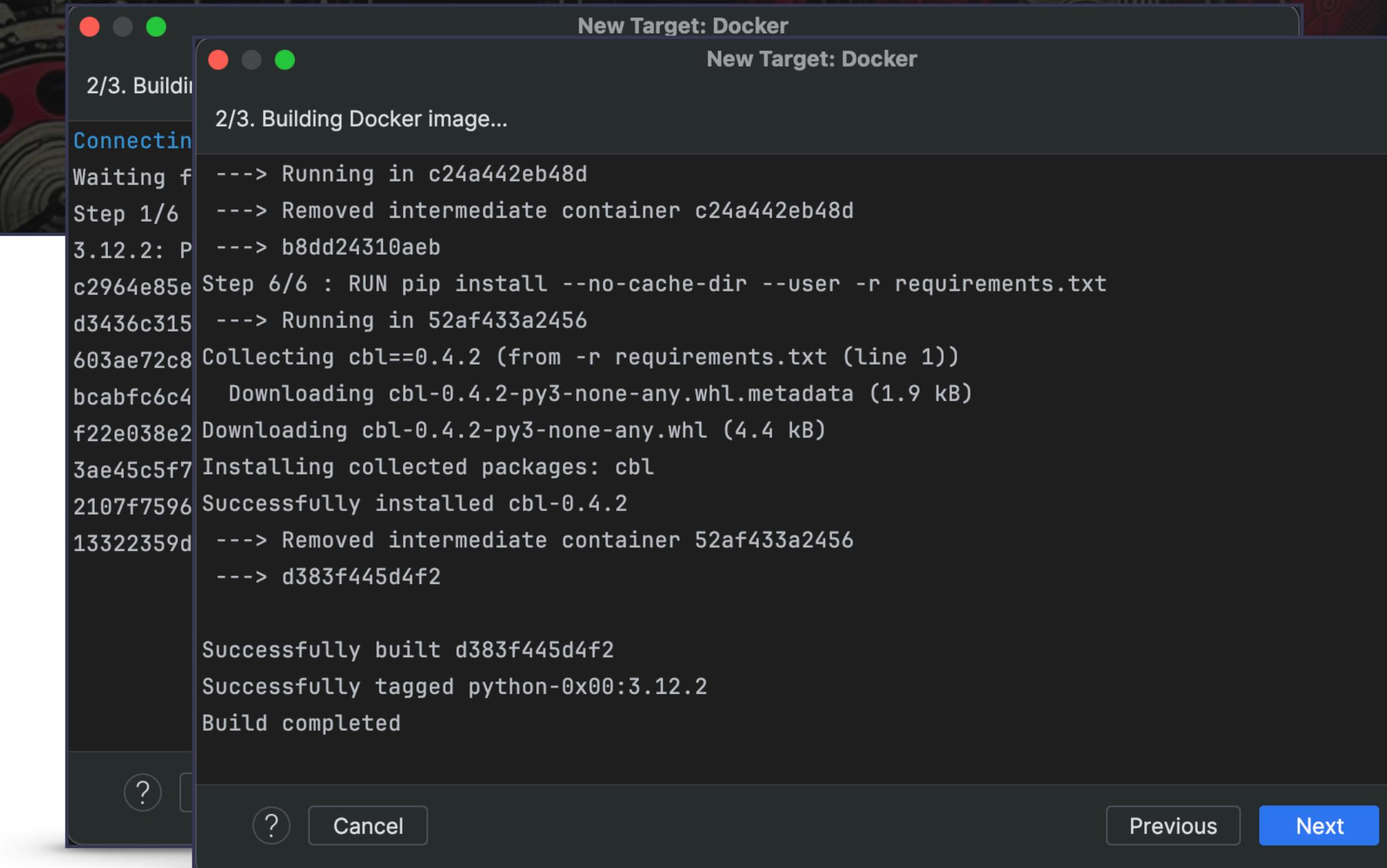
After building the image, we select the *Docker System Interpreter*.



select Python

You can also use virtual environments here, but as we are already isolated inside the Docker container, we skip this.

In the Docker Dashboard we can see our new image and the output shows the correct Python interpreter.



building the image

| | | |
|--|-----------------|--------|
| <input type="checkbox"/> pycharm_helpers 8e83133d1e7e | PY-233.14475.56 | In use |
| <input type="checkbox"/> python-0x00 d383f445d4f2 | 3.12.2 | Unused |
| <input type="checkbox"/> python d8be44680b2e | 3.12.2 | Unused |

Docker images

```
/usr/local/bin/python3 /opt/project/snippets/about_setup.py  
python 3.12.2 /usr/local/bin/python3  
cbl 0.4.2 ~/.local/lib/python3.12/site-packages/cbl
```

output



Configure Python in Visual Studio Code

IDE Configurations

The situation is very similar in Visual Studio Code.

The selection shows the variants known to the system.

A screenshot of the Visual Studio Code interface. The main area shows a Python script named `solution_grapevine.py` with the following code:

```
1 # ( 
2 import 
3     def 
4         if __name__ == "__main__":
5             sys.exit(task_name())
6         )
7     )
8
9
10
11
12
```

A context menu is open over the code, with the title "Select Interpreter". The "Selected Interpreter" dropdown shows the path: `~/Projects/code-based-learning/python/course/my_python/`. Below it is a list of available interpreters:

- + Enter interpreter path...
- ⚙️ Use Python from `python.defaultInterpreterPath` setting `~/Projects/cod...`
- ★ Python 3.12.0a3 64-bit ('3.12.0a3': pyenv) `~/pyenv/versi...` Recommended
- Python 3.11.1 ('venv') `~/Projects/code-based-learning/python/course/my_...` Venv
- Python 3.12.0a3 64-bit ('3.12.0a3') `~/pyenv/versions/3.12.0a3/bin/pyth...` Pyenv
- Python 3.11.1 64-bit ('3.11.1') `~/pyenv/versions/3.11.1/bin/python`
- Python 3.6.15 64-bit ('3.6.15') `~/pyenv/versions/3.6.15/bin/python`
- Python 3.9.13 64-bit `/usr/local/bin/python3` Global
- Python 3.9.6 64-bit `/usr/bin/python3`

A blue arrow points from the bottom right towards the status bar. The status bar displays the following information: "solution_grapevine.py", "Select Interpreter", "Selected Interpreter: ~/Projects/code-based-learning/python/course/my_python/", "Ln 1, Col 1", "Spaces: 4", "UTF-8", "LF", "Python", "3.11.1 ('venv': venv)", "Spell", and icons for Connect, Bell, and Help.



CODE BASED
LEARNING

IDE CONFIGURATIONS

- RUNNING A SCRIPT -

PyCHARM

DOCKER

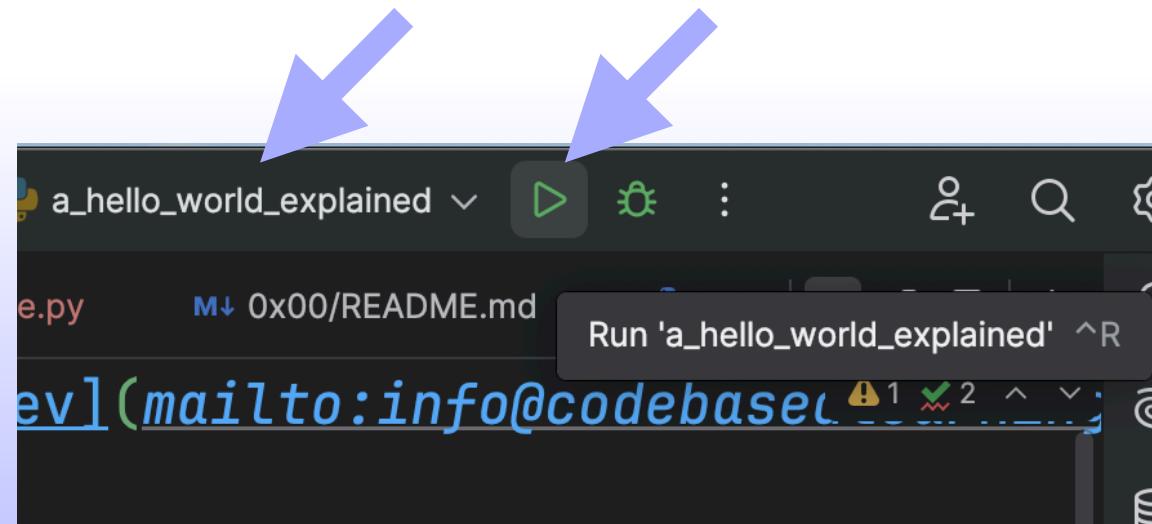
VISUAL STUDIO CODE

Run | Project Selection

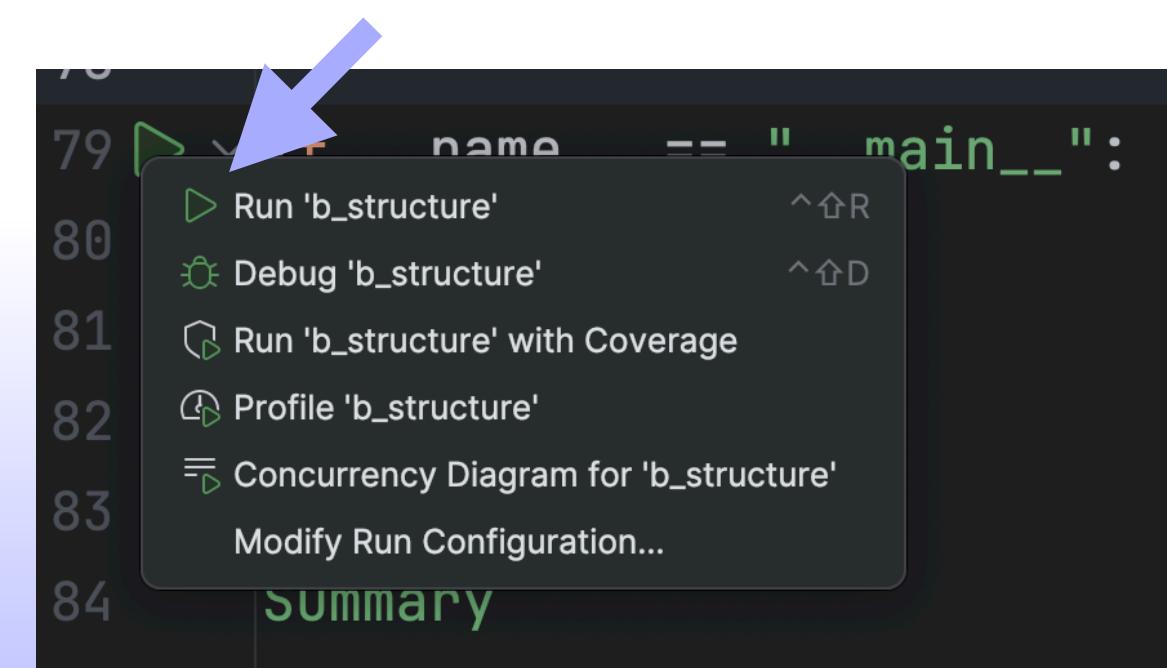
Running a Script

In PyCharm and Visual Studio Code you can run a script from the IDE without going into a terminal. The IDE will take care of using the associated Python interpreter.

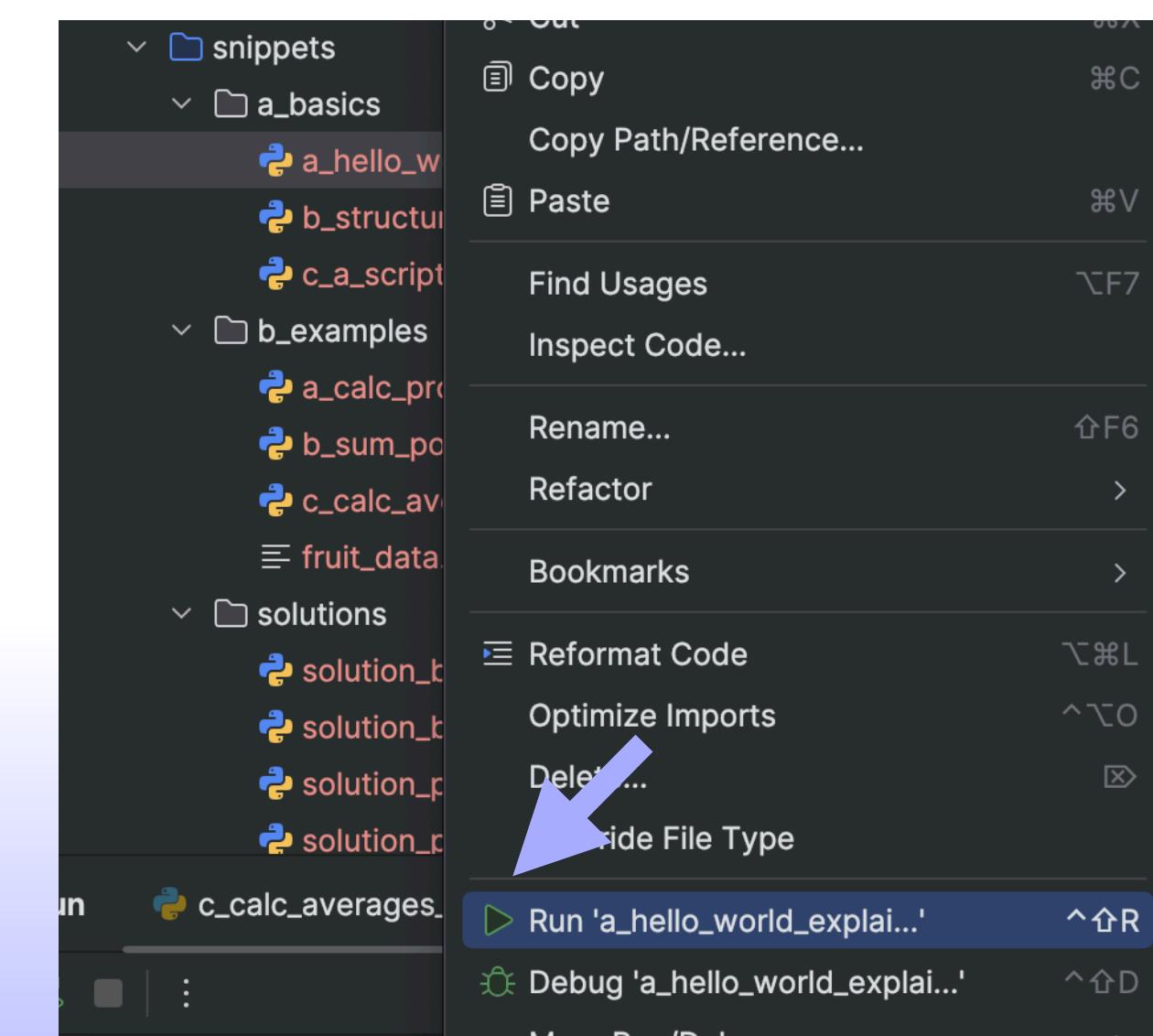
Note that just clicking the run button  may run a previous project, so you need to make sure that the project you are working on is selected. In PyCharm it looks like this.



run from top menu



run from main-guard



run from context menu





CODE BASED LEARNING

If you are missing something or have suggestions for improvement, please send me an [E-Mail](#).

!

Finished

