CE CoP Python Workshop
Jupyter Notebooks Setup

Jupyter (formerly IPython Notebook) is an open-source project that lets you easily combine Markdown text and executable Python source code on one canvas called a notebook. All of the workshop's modules will be distributed via this files, setting up the environment will be vital to continue from this point on. In the following sections, you will see a couple options that you can pick from: Intel devcloud setup (1) and VS Code setup (2).

In case anyone is stalled here, please do not hesitate to ask for help to the CE CoP core team.

1 Intel Devcloud Setup

The Intel Devcloud provides a strightforward way to create and use Jupyter Notebooks with no extra steps required to setup an environment. Once you have the access to the Intel Devcloud that is everything you need to start working on it.

1.1 Importing the files into the Devcloud

In order to have the files as part of the devcloud, the following steps will be required:

1. Unzip the .ipynb files provided for this workshop in a known location.
2. Go to https://jupyter.oneapi.devcloud.intel.com/hub/login?next=%2Fhub%2F.
3. Press "Click Here to Sign In" button.
4. Provide your intel's email and press "Continue" button
5. Select the file browser view by clicking the folder icon on the left as shown in Figure 1.

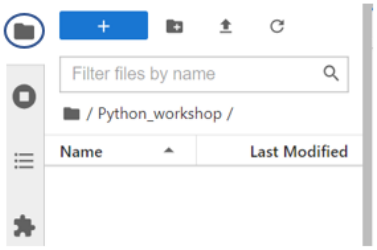


Figure 1: File browser

6. Select “New Folder” option as presented in Figure 2.

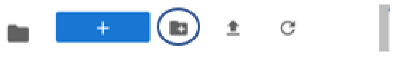


Figure 2: New folder creation

7. Enter “Python_workshop” as folder’s name.
8. Open the “Python_workshop” folder by double clicking over its name.
9. Select “Upload Files” option. See Figure 3 for reference.

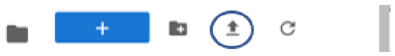


Figure 3: Uploading the files

10. Select all the .ipynb files and press the “Open” button.
11. Open the files by double clicking over its name.

2 VS Code Setup

Visual Studio Code supports working with Jupyter Notebooks natively, and through Python code files. To work with Python in Jupyter Notebooks, you must activate an Anaconda environment in VS Code, or another Python environment in which you’ve installed the Jupyter package.

2.1 Environment setup

2.1.1 Automatically setting up your environment

VS Code can handle the installation of those packages automatically. There are several ways for it to suggest what to install, let’s find the extension in the market place. In order to find the market, please click on Extensions, Figure 4 shows where it is.

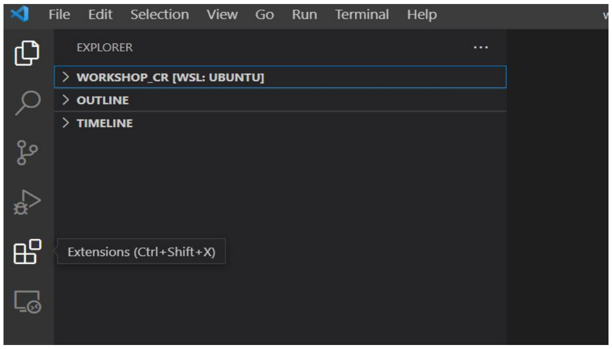
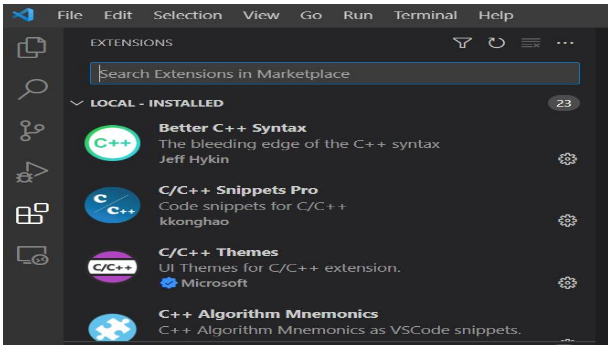


Figure 4: VS Code Extensions

Then, the market place will be displayed for you to install the extension. Figure 5 has a view of the market.



Better C++ Syntax
The bleeding edge of the C++ syntax
Jeff Hykin

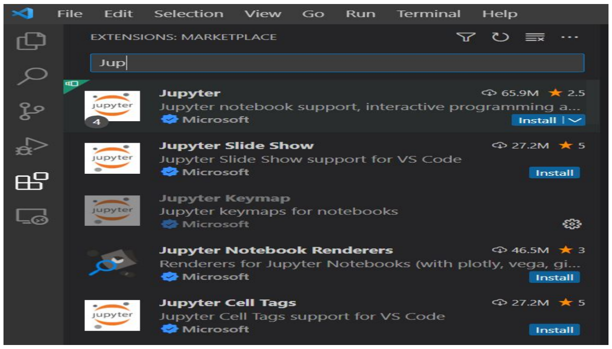
C/C++ Snippets Pro
Code snippets for C/C++
kkonghao

C/C++ Themes
UI Themes for C/C++ extension.
Microsoft

C++ Algorithm Mnemonics
C++ Algorithm Mnemonics as VSCode snippets.

Figure 5: VS Code Market

It is required for us to look for the extension so let's type "Jup". See Figure 6 as a reference.



Jup

Jupyter — 65.9M ★ 2.5
Jupyter notebook support, interactive programming a...
Microsoft — Install

Jupyter Slide Show — 27.2M ★ 5
Jupyter Slide Show support for VS Code
Microsoft — Install

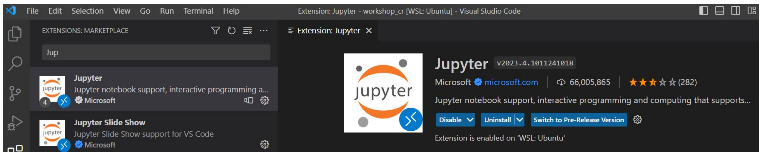
Jupyter Keymap
Jupyter keymaps for notebooks
Microsoft

Jupyter Notebook Renderers — 46.5M ★ 3
Renderers for Jupyter Notebooks (with plotly, vega, gi...
Microsoft — Install

Jupyter Cell Tags — 27.2M ★ 5
Jupyter Cell Tags support for VS Code
Microsoft — Install

Figure 6: VS Code Market: Jupyter

Proceed to install the Jupyter's extension, once completed, it will look just like Figure 7.



Extension: Jupyter - workshop_cr [WSL: Ubuntu] - Visual Studio Code

Jup

Jupyter
Jupyter notebook support, interactive programming a...
Microsoft

Jupyter Slide Show
Jupyter Slide Show support for VS Code
Microsoft

Jupyter v2023.4.1011241018
Microsoft microsoft.com | 66,005,865 | ★★★☆☆ (282)
Jupyter notebook support, interactive programming and computing that supports...
Extension is enabled on 'WSL: Ubuntu'

Figure 7: Jupyter Installed

Just to double check the kernels (Jupyter and Python), let's create and empty Jupyter file. In Figure 8 you'll be able to see where to find it.

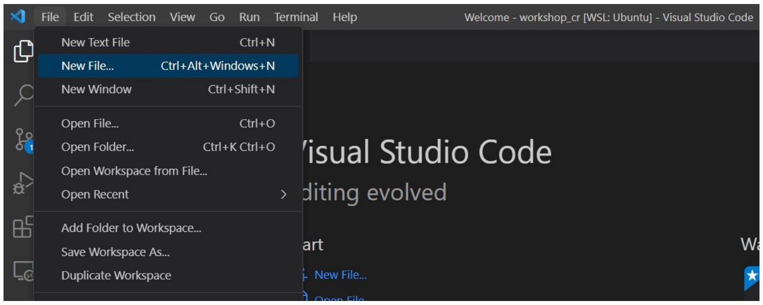


Figure 8: Creating a new file

Then you will require to select the file extension for VS Code to trigger the automatic setup. In Figure 9, once you type "Jup" to look for the file extension, it will suggest the type you are looking for.

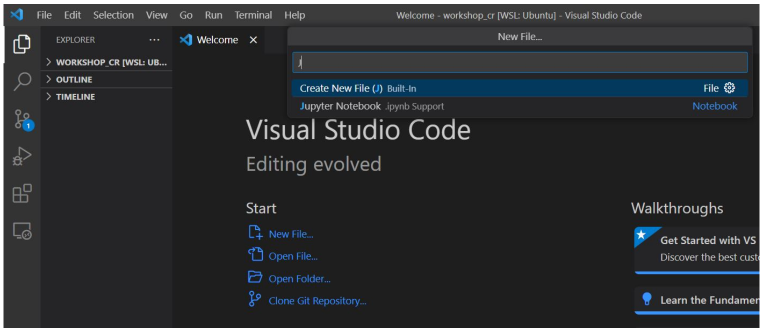


Figure 9: Picking the file extension

As soon as you click on the type, an empty file will be created such as the one presented in Figure 10. Selecting a Kernel is required for your files to be able to be executed.

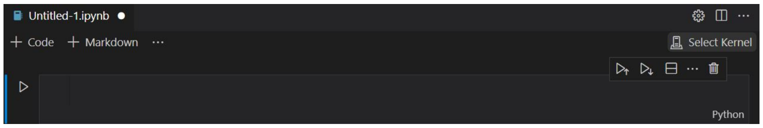


Figure 10: Empty file created, Select Kernel enabled

For VS Code to help you to look for the required packages, once you hit the "Select Kernel" button in Figure 10, the view presented in Figure 11 is shown for you to pick the right extension.

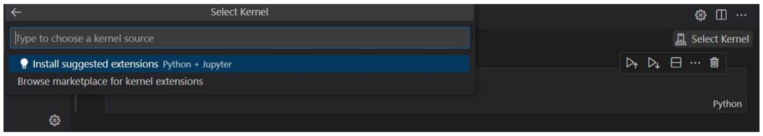

Select Kernel
Type to choose a kernel source
Install suggested extensions Python + Jupyter
Browse marketplace for kernel extensions
Select Kernel
Python

Figure 11: VS Code Market suggestions

Once you have Python and Jupyter set, it will be able to execute the files.

2.1.2 Manual installation/setup

To select an environment, use the Python: Select Interpreter command from the Command Palette (Ctrl+Shift+P). Once the appropriate environment is activated, you can create and open a Jupyter Notebook, connect to a remote Jupyter server for running code cells, and export a Jupyter Notebook as a Python file.

In case you want to have a better understanding on how this is managed, please visit Jupyter Notebooks in VS Code.

2.2 Importing the files into VS Code

It does not matter if you selected the automatic (2.1.1) or the manual (2.1.2) setup, you will require to do an import. Having the files placed into VS Code is quite simple, please follow these steps:

1. Unzip the .ipynb files provided for this workshop in a known location.
2. Have VS Code opened.
3. As shown in Figure 12, hit on “Open Folder...”.

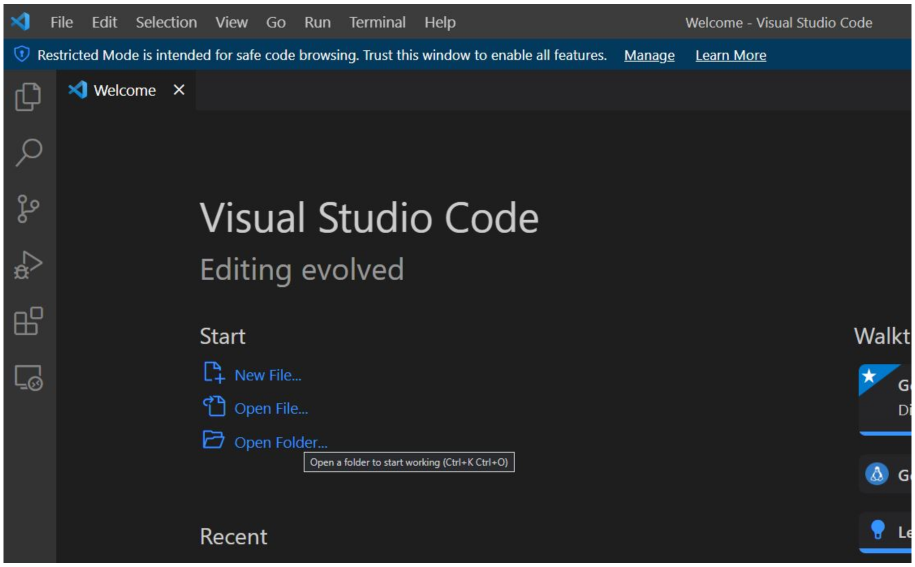
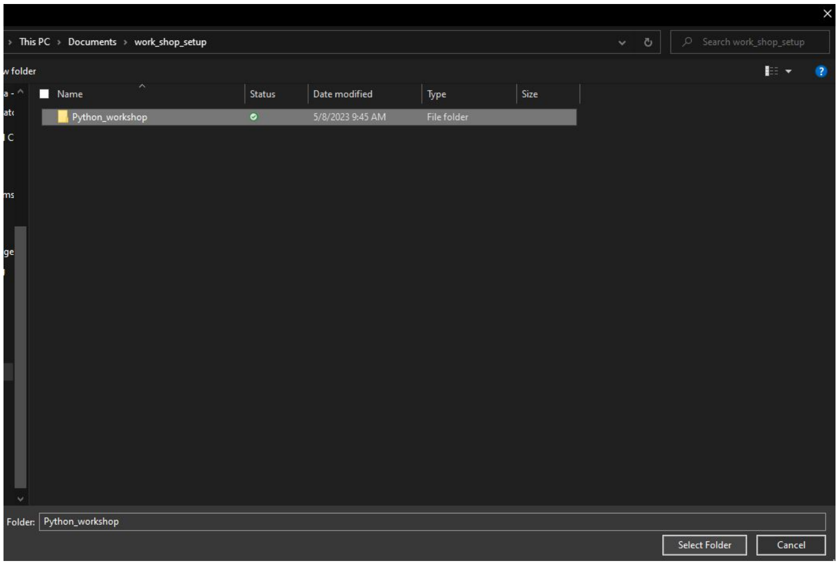

Restricted Mode is intended for safe code browsing. Trust this window to enable all features.

Figure 12: Importing the workshop directory.

4. On the “Open Folder” view, select the known path where the files were extracted. Figure 13 is a reference.

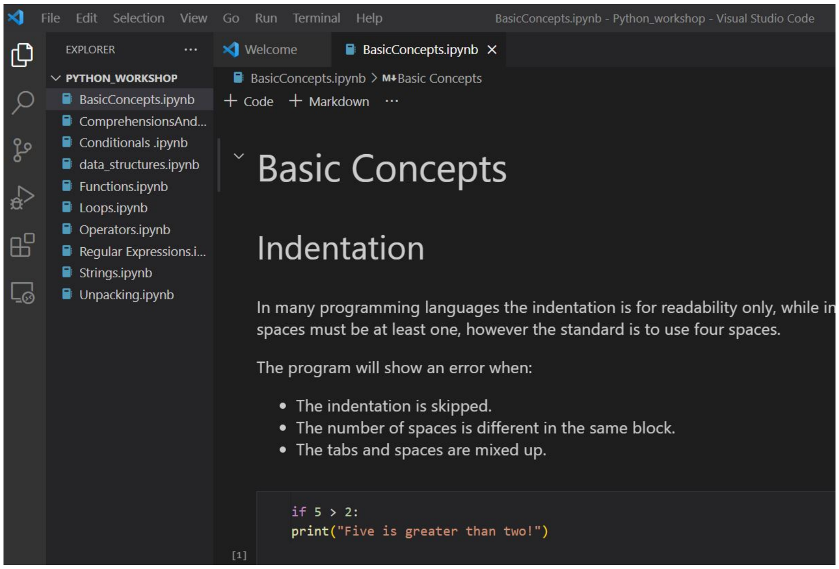


Name	Status	Date modified	Type	Size
Python_workshop	✓	5/8/2023 9:45 AM	File folder	

Folder: Python_workshop

Figure 13: Opening files.

5. Click on any file that has been imported to open it. See Figure 14.



Basic Concepts

Indentation

In many programming languages the indentation is for readability only, while in
spaces must be at least one, however the standard is to use four spaces.

The program will show an error when:

- The indentation is skipped.
- The number of spaces is different in the same block.
- The tabs and spaces are mixed up.

```
if 5 > 2:
print("Five is greater than two!")
```

Figure 14: Opening files.