Getting started with Python

Mirjeta Pasha and Connor Sanderford
*

July 2022

1 Installation

Python [4, 3, 2]is a free, open-source, and object oriented programming language that has become popular in scientific research in the recent decades. Python language is an integral part of a few courses in the school. Different platforms and compilers can be used for python. To get started, for general information, follow the following link.

`https://www.python.org/downloads/`

1.1 Python in different operating systems (OS)

1.1.1 Python for MAC OS

To install Python in MAC OS follow this link.

`https://www.python.org/downloads/macos/`

We recommend you install the latest version of Python 3.9 (3.9.13) or Python 3.8 (3.8.13) - some libraries do not yet support Python 3.10.

Check that Python is installed by opening a terminal session and running `python`.

1.1.2 Python for Windows OS

To install Python in Windows, follow the following link

`https://www.python.org/downloads/windows/`.

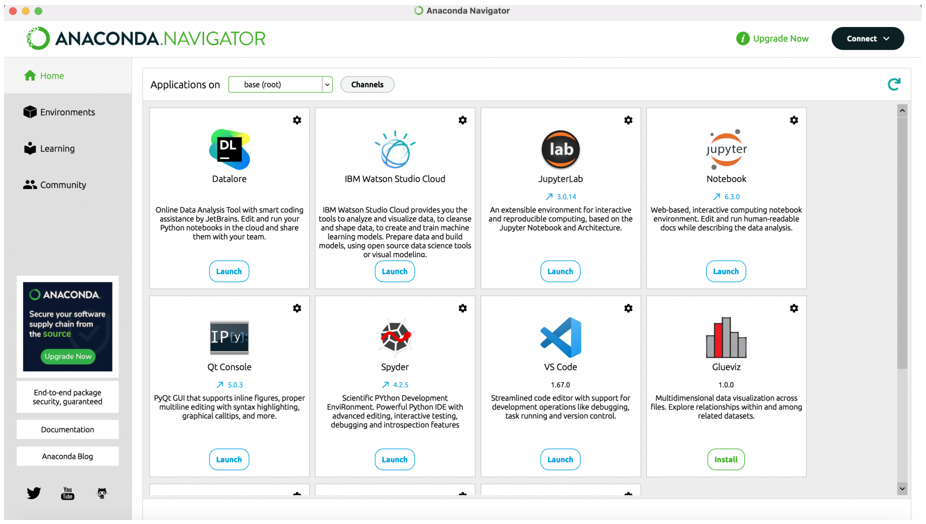
We recommend you install the latest version of Python 3.9 (3.9.13) or Python 3.8 (3.8.13) - some libraries do not yet support Python 3.10. Download the Windows installer compatible with your system architecture (this is usually 64-bit).

Check that Python is installed by opening a terminal session and running `python`.

*Tufts University (mirjeta.pasha@tufts.edu) and Arizona State University (csanderf@asu.edu)

1.1.3 Python through Anaconda Navigator

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands [1].



Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.

IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

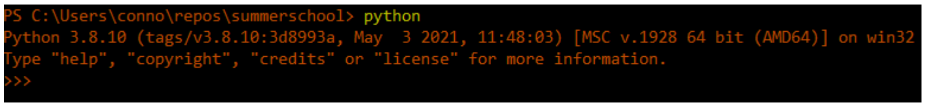
Streamlined code editor with support for development operations like debugging, task running and version control.

Multidimensional data visualization across files. Explore relationships within and among related datasets.

Figure 1: Anaconda Navigator

Follow the link `https://www.anaconda.com/products/distribution` to download an installer compatible with your OS and system architecture, then run the installer.

Check that Python is installed by opening Anaconda Prompt and running `python`. On Windows, open the Anaconda Prompt (Click Start, select Anaconda Prompt). On Mac, open the Launchpad, then open terminal (we illustrate the later in Figure 2).



```
PS C:\Users\conno\repos\summerschool> python
Python 3.8.10 (tags/v3.8.10:3d8993a, May  3 2021, 11:48:03) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Figure 2: Checking that you have access to a Python installation

1.2 Virtual environments

Unlike other languages, Python manages dependencies through virtual environments, where each contains an instance of the Python interpreter along with the installed dependencies. Python has several different tools for building and managing virtual environments, but this document will cover `venv` and `conda`, which correspond to the standard Python distribution, and the Anaconda distribution, respectively.

1.2.1 Using venv to build a virtual environment

`venv` environments are built at the project level, rather than globally. We recommend creating a directory that will store all your Python projects associated with the school, which we will call `summerschool/`. Once you have done this, navigate to the `summerschool/` directory in File Explorer, then click in the address bar, and type `cmd` to open a Command Prompt, or `powershell` to open Powershell at this location.

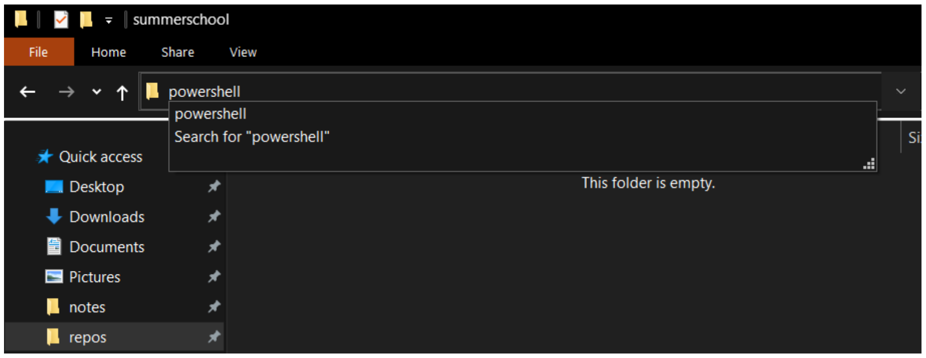
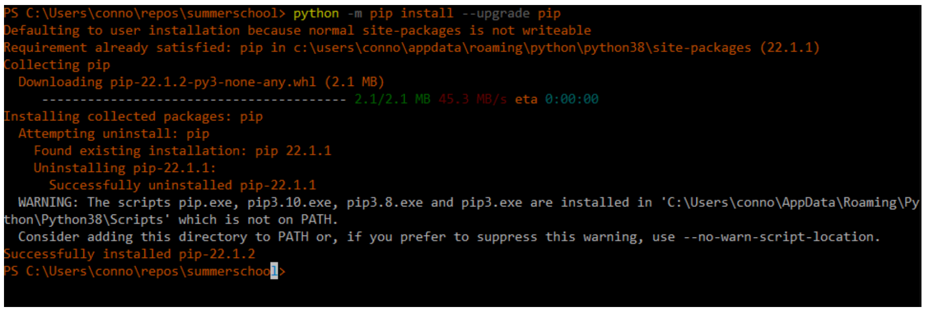


Figure 3: Opening powershell from the File Explorer

Before we build the virtual environment, we make sure that the Python package manager, `pip`, is up to date in the base Python environment. In your terminal session, run

```
python -m pip install --upgrade pip
```

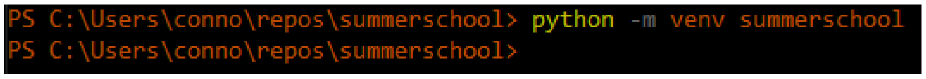
to update `pip`.



```
PS C:\Users\conno\repos\summerschool> python -m pip install --upgrade pip
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pip in c:\users\conno\appdata\roaming\python\python38\site-packages (22.1.1)
Collecting pip
  Downloading pip-22.1.2-py3-none-any.whl (2.1 MB)
     ---------------------------------------- 2.1/2.1 MB 45.3 MB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 22.1.1
    Uninstalling pip-22.1.1:
      Successfully uninstalled pip-22.1.1
  WARNING: The scripts pip.exe, pip3.10.exe, pip3.8.exe and pip3.exe are installed in 'C:\Users\conno\AppData\Roaming\Py
thon\Python38\Scripts' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed pip-22.1.2
PS C:\Users\conno\repos\summerschool>
```

Figure 4: Upgrade pip in the base python installation

We can now create a virtual environment inside the `summerschool/` directory. Run
`python -m venv summerschool`
to create a virtual environment with the name `summerschool`.

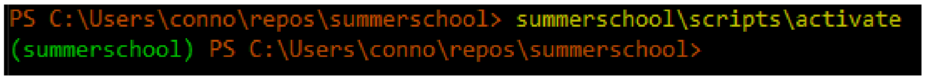


```
PS C:\Users\conno\repos\summerschool> python -m venv summerschool
PS C:\Users\conno\repos\summerschool>
```

Figure 5: Building a venv environment

We can now activate the virtual environment. On a Windows machine, run
`summerschool\scripts\activate` in order to activate the virtual environment - if this works, you will see that your prompt has changed, and is now preceeded by `(summerschool)`.

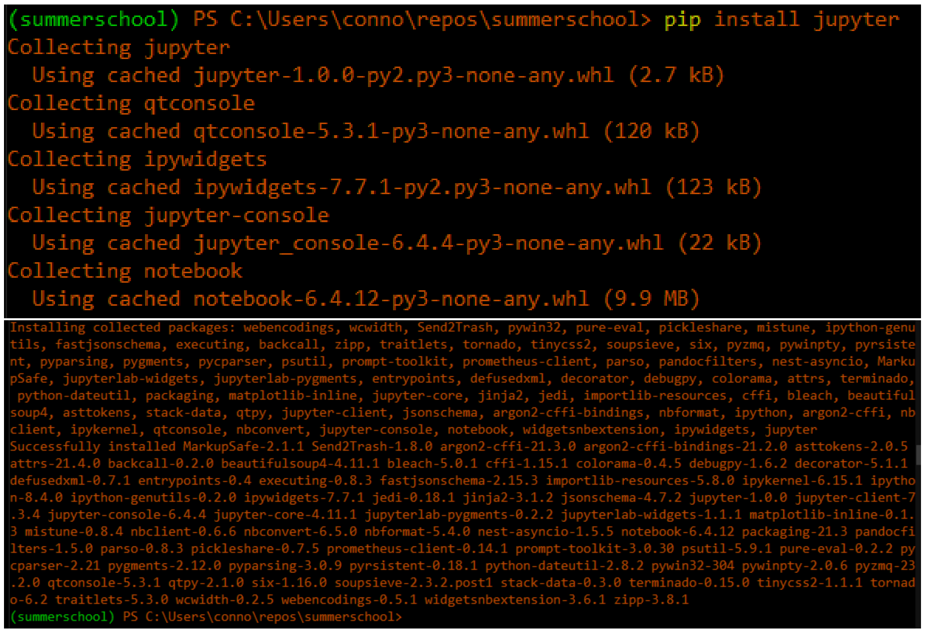
On a Mac or Linux machine, run
`source summerschool/bin/activate` in order to activate the virtual environment - if this works, you will see that your prompt has changed, and is now preceeded by `(summerschool)`.



```
PS C:\Users\conno\repos\summerschool> summerschool\scripts\activate
(summerschool) PS C:\Users\conno\repos\summerschool>
```

Figure 6: Activating a venv environment on Windows

With your virtual environment activated, any packages you `pip install` are now installed to the virtual environment rather than the base environment. To install a package to the virtual environment, run `pip install package_name`.
In order to execute Jupyter notebooks from this environment, run `pip install jupyter`.

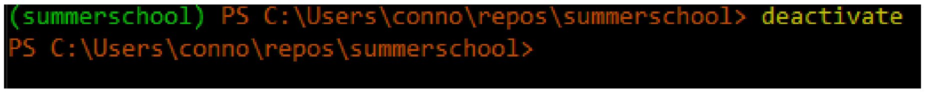


```
(summerschool) PS C:\Users\conno\repos\summerschool> pip install jupyter
Collecting jupyter
  Using cached jupyter-1.0.0-py2.py3-none-any.whl (2.7 kB)
Collecting qtconsole
  Using cached qtconsole-5.3.1-py3-none-any.whl (120 kB)
Collecting ipywidgets
  Using cached ipywidgets-7.7.1-py2.py3-none-any.whl (123 kB)
Collecting jupyter-console
  Using cached jupyter_console-6.4.4-py3-none-any.whl (22 kB)
Collecting notebook
  Using cached notebook-6.4.12-py3-none-any.whl (9.9 MB)
```

```
Installing collected packages: webencodings, wcwidth, Send2Trash, pywin32, pure-eval, pickleshare, mistune, ipython-genu
tils, fastjsonschema, executing, backcall, zipp, traitlets, tornado, tinycss2, soupsieve, six, pyzmq, pywinpty, pyrsiste
nt, pyparsing, pygments, pycparser, psutil, prompt-toolkit, prometheus-client, parso, pandocfilters, nest-asyncio, Marku
pSafe, jupyterlab-widgets, jupyterlab-pygments, entrypoints, defusedxml, decorator, debugpy, colorama, attrs, terminado,
 python-dateutil, packaging, matplotlib-inline, jupyter-core, jinja2, jedi, importlib-resources, cffi, bleach, beautiful
soup4, asttokens, stack-data, qtpy, jupyter-client, jsonschema, argon2-cffi-bindings, nbformat, ipython, argon2-cffi, nb
client, ipykernel, qtconsole, nbconvert, jupyter-console, notebook, widgetsnbextension, ipywidgets, jupyter
Successfully installed MarkupSafe-2.1.1 Send2Trash-1.8.0 argon2-cffi-21.3.0 argon2-cffi-bindings-21.2.0 asttokens-2.0.5
attrs-21.4.0 backcall-0.2.0 beautifulsoup4-4.11.1 bleach-5.0.1 cffi-1.15.1 colorama-0.4.5 debugpy-1.6.2 decorator-5.1.1
defusedxml-0.7.1 entrypoints-0.4 executing-0.8.3 fastjsonschema-2.15.3 importlib-resources-5.8.0 ipykernel-6.15.1 ipytho
n-8.4.0 ipython-genutils-0.2.0 ipywidgets-7.7.1 jedi-0.18.1 jinja2-3.1.2 jsonschema-4.7.2 jupyter-1.0.0 jupyter-client-7
.3.4 jupyter-console-6.4.4 jupyter-core-4.11.1 jupyterlab-pygments-0.2.2 jupyterlab-widgets-1.1.1 matplotlib-inline-0.1.
3 mistune-0.8.4 nbclient-0.6.6 nbconvert-6.5.0 nbformat-5.4.0 nest-asyncio-1.5.5 notebook-6.4.12 packaging-21.3 pandocfi
lters-1.5.0 parso-0.8.3 pickleshare-0.7.5 prometheus-client-0.14.1 prompt-toolkit-3.0.30 psutil-5.9.1 pure-eval-0.2.2 py
cparser-2.21 pygments-2.12.0 pyparsing-3.0.9 pyrsistent-0.18.1 python-dateutil-2.8.2 pywin32-304 pywinpty-2.0.6 pyzmq-23
.2.0 qtconsole-5.3.1 qtpy-2.1.0 six-1.16.0 soupsieve-2.3.2.post1 stack-data-0.3.0 terminado-0.15.0 tinycss2-1.1.1 tornad
o-6.2 traitlets-5.3.0 wcwidth-0.2.5 webencodings-0.5.1 widgetsnbextension-3.6.1 zipp-3.8.1
(summerschool) PS C:\Users\conno\repos\summerschool>
```

Figure 7: Installing Jupyter and its dependencies

To deactivate the virtual environment and return to the base environment, run `deactivate`.

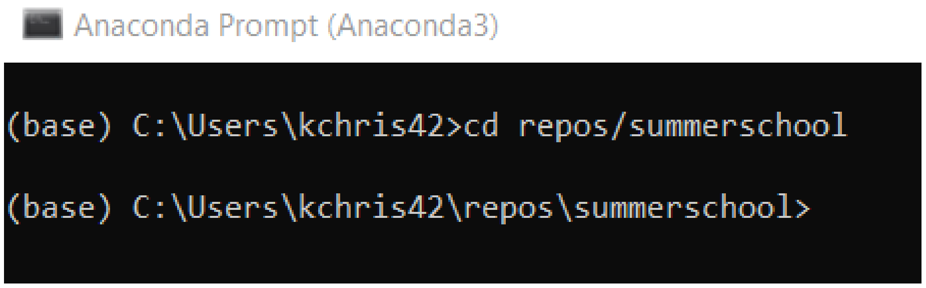


```
(summerschool) PS C:\Users\conno\repos\summerschool> deactivate
PS C:\Users\conno\repos\summerschool>
```

Figure 8: Deactivating the venv environment

1.2.2 Using conda to build a virtual environment

`conda` environments are built at the global level, rather than within projects. That said, we still recommend creating a directory that will store all your Python projects associated with the school, which we call `summerschool/`. Once you have done this, open Anaconda Prompt, and navigate to your `summerschool/` directory using the `cd` (change directory) command.



Anaconda Prompt (Anaconda3)

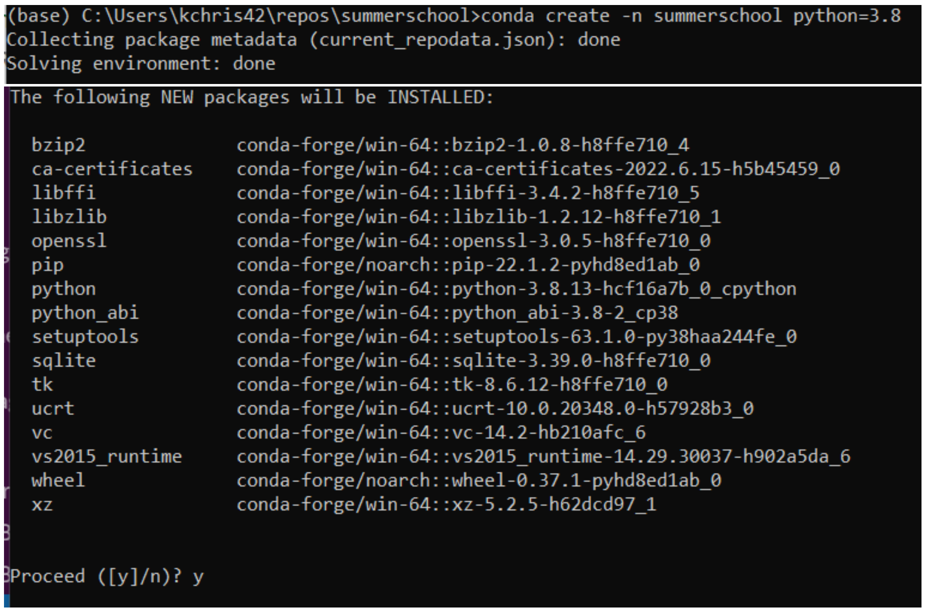
```
(base) C:\Users\kchris42>cd repos/summerschool

(base) C:\Users\kchris42\repos\summerschool>
```

Figure 9: cding to your directory

We can now create a conda environment specifically for projects associated with the summer school. Run

`conda create -n summerschool python=3.8`

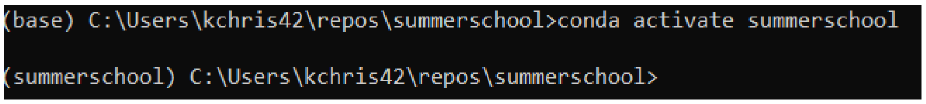
to create a conda environment with the name `summerschool` and associated python version of 3.8.



```
(base) C:\Users\kchris42\repos\summerschool>conda create -n summerschool python=3.8
Collecting package metadata (current_repodata.json): done
Solving environment: done
The following NEW packages will be INSTALLED:

  bzip2              conda-forge/win-64::bzip2-1.0.8-h8ffe710_4
  ca-certificates    conda-forge/win-64::ca-certificates-2022.6.15-h5b45459_0
  libffi             conda-forge/win-64::libffi-3.4.2-h8ffe710_5
  libzlib            conda-forge/win-64::libzlib-1.2.12-h8ffe710_1
  openssl            conda-forge/win-64::openssl-3.0.5-h8ffe710_0
  pip                conda-forge/noarch::pip-22.1.2-pyhd8ed1ab_0
  python             conda-forge/win-64::python-3.8.13-hcf16a7b_0_cpython
  python_abi         conda-forge/win-64::python_abi-3.8-2_cp38
  setuptools         conda-forge/win-64::setuptools-63.1.0-py38haa244fe_0
  sqlite             conda-forge/win-64::sqlite-3.39.0-h8ffe710_0
  tk                 conda-forge/win-64::tk-8.6.12-h8ffe710_0
  ucrt               conda-forge/win-64::ucrt-10.0.20348.0-h57928b3_0
  vc                 conda-forge/win-64::vc-14.2-hb210afc_6
  vs2015_runtime     conda-forge/win-64::vs2015_runtime-14.29.30037-h902a5da_6
  wheel              conda-forge/noarch::wheel-0.37.1-pyhd8ed1ab_0
  xz                 conda-forge/win-64::xz-5.2.5-h62dcd97_1


Proceed ([y]/n)? y
```

Figure 10: Creating a new conda environment

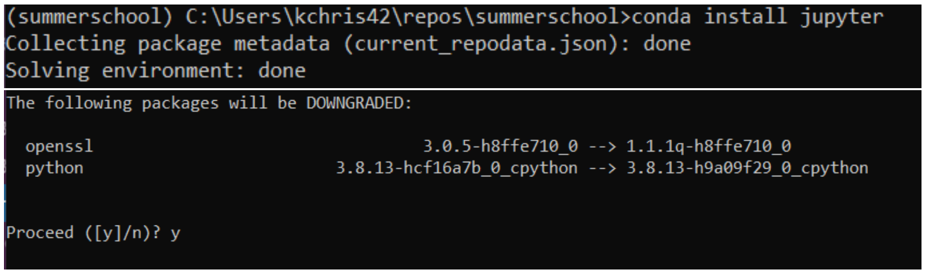
We can now activate the virtual environment. Run `conda activate summerschool` in order to activate the conda environment - if this works, you will see that your prompt has changed, and is now preceeded by `(summerschool)`.



```
(base) C:\Users\kchris42\repos\summerschool>conda activate summerschool

(summerschool) C:\Users\kchris42\repos\summerschool>
```

Figure 11: Activating a conda environment

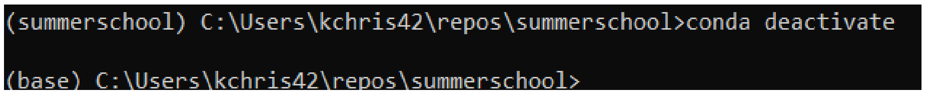
With your conda environment activated, any packages you `conda install` are now installed to the virtual environment rather than the base environment. To install a package to the virtual environment, run `conda install package_name`. In order to execute Jupyter notebooks from this environment, run `conda install jupyter`.



```
(summerschool) C:\Users\kchris42\repos\summerschool>conda install jupyter
Collecting package metadata (current_repodata.json): done
Solving environment: done
The following packages will be DOWNGRADED:

  openssl                             3.0.5-h8ffe710_0 --> 1.1.1q-h8ffe710_0
  python                      3.8.13-hcf16a7b_0_cpython --> 3.8.13-h9a09f29_0_cpython


Proceed ([y]/n)? y
```

Figure 12: Installing Jupyter to a conda environment

To deactivate the virtual environment and return to the base environment, run `conda deactivate`.



```
(summerschool) C:\Users\kchris42\repos\summerschool>conda deactivate

(base) C:\Users\kchris42\repos\summerschool>
```

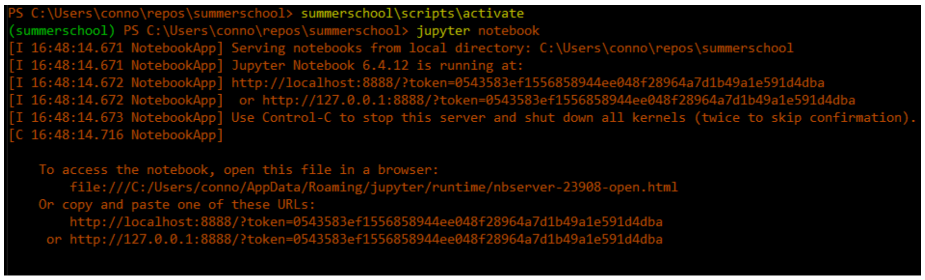
Figure 13: Deactivating a conda environment

1.3 Platforms for Python

1.3.1 Python in the Jupyter notebook

Because you have already installed `jupyter` to your `venv` or `conda` environment, you will be able to run Python in a Jupyter notebook without any further setup.

With your virtual environment activated, run `jupyter notebook` in your terminal.



```
PS C:\Users\conno\repos\summerschool> summerschool\scripts\activate
(summerschool) PS C:\Users\conno\repos\summerschool> jupyter notebook
[I 16:48:14.671 NotebookApp] Serving notebooks from local directory: C:\Users\conno\repos\summerschool
[I 16:48:14.671 NotebookApp] Jupyter Notebook 6.4.12 is running at:
[I 16:48:14.672 NotebookApp] http://localhost:8888/?token=0543583ef1556858944ee048f28964a7d1b49a1e591d4dba
[I 16:48:14.672 NotebookApp]  or http://127.0.0.1:8888/?token=0543583ef1556858944ee048f28964a7d1b49a1e591d4dba
[I 16:48:14.673 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 16:48:14.716 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/conno/AppData/Roaming/jupyter/runtime/nbserver-23908-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=0543583ef1556858944ee048f28964a7d1b49a1e591d4dba
     or http://127.0.0.1:8888/?token=0543583ef1556858944ee048f28964a7d1b49a1e591d4dba
```

Figure 14: Starting a Juypter notebook session

This should open a new tab in your browser which displays the Jupyter server. Create a new notebook by clicking on `new` in the upper right hand corner of the screen. You will be able to create a new notebook with your choice of kernel here, in particular the one you currently have activated in your terminal session.

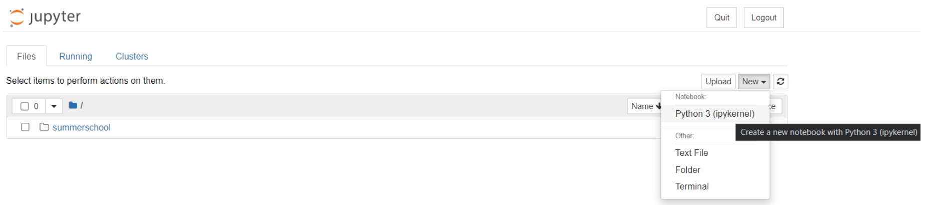


Figure 15: Creating a new Jupyter notebook

When ending your Jupyter session, be sure to shut down Jupyter rather than just closing the tab in your browser. Click the "Quit" button in the upper right to shut down the Jupyter server.

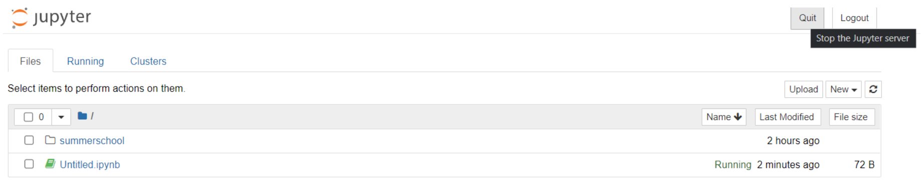


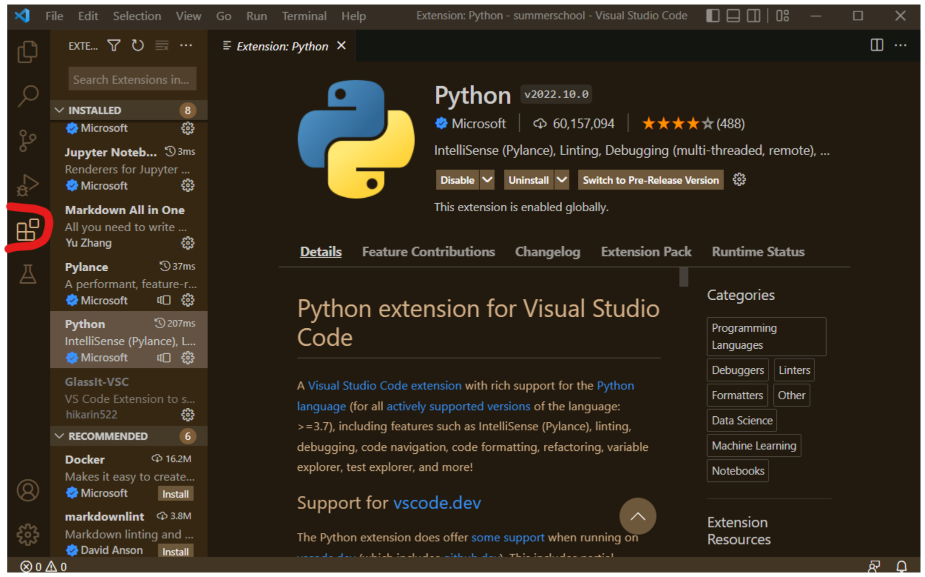
Figure 16: Ending a Jupyter notebook session

1.3.2 Python through Visual Studio

Download and install VSCode:

`https://code.visualstudio.com/`

Open VSCode, and click on the extensions view. In the extensions view, type Python, and install the official Python extension from Microsoft. This should also install the Jupyter extension from Microsoft which enables notebook support.



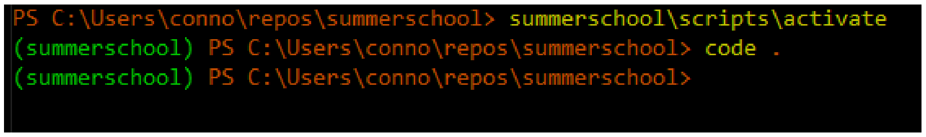
A Visual Studio Code extension with rich support for the Python language (for all actively supported versions of the language: >=3.7), including features such as IntelliSense (Pylance), linting, debugging, code navigation, code formatting, refactoring, variable explorer, test explorer, and more!

The Python extension does offer some support when running on

Figure 17: The VS Code extensions page, and Python extension.

Once these are finished installing, close VSCode.

With your virtual environment activated, run `code .` in your terminal. VSCode will open, with the current directory of your terminal as your workspace.



```
PS C:\Users\conno\repos\summerschool> summerschool\scripts\activate
(summerschool) PS C:\Users\conno\repos\summerschool> code .
(summerschool) PS C:\Users\conno\repos\summerschool>
```

Figure 18: Opening VS Code from a terminal with a working directory.

You can create new Python files by clicking the new file button and using the file ending `.py`, or new notebooks by clicking the new file button and using the file ending `.ipynb`.

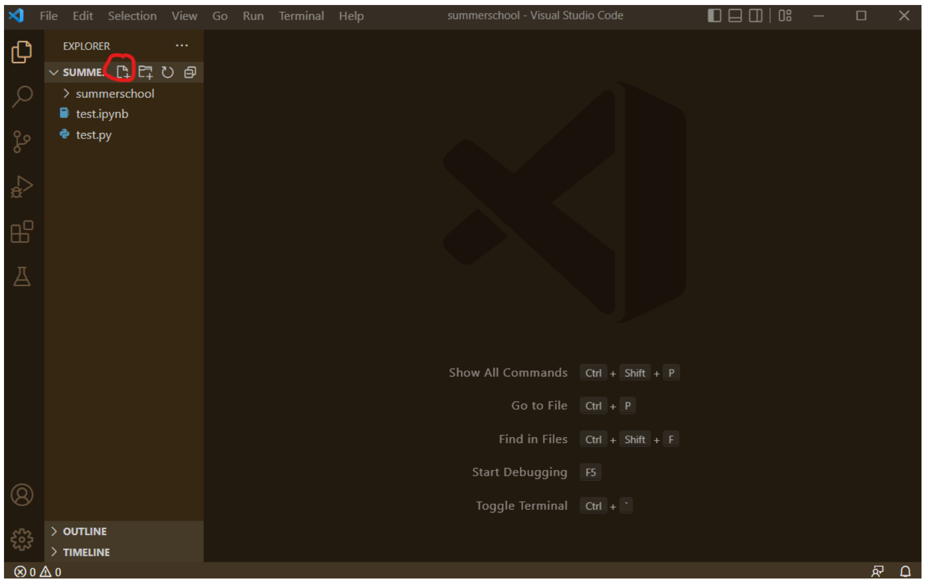

File Edit Selection View Go Run Terminal Help
summerschool - Visual Studio Code
EXPLORER
SUMME...
summerschool
test.ipynb
test.py
Show All Commands Ctrl + Shift + P
Go to File Ctrl + P
Find in Files Ctrl + Shift + F
Start Debugging F5
Toggle Terminal Ctrl + `
OUTLINE
TIMELINE

Figure 19: The "new file" button.

If you have a `.py` file open, look at the Status Bar in the lower right corner. You should see that the interpreter associated with your virtual environment is selected (it will say something like `3.8.10 ('summerschool':venv)` for a venv environment). If it is not, then click there and select the interpreter that you would like to use.

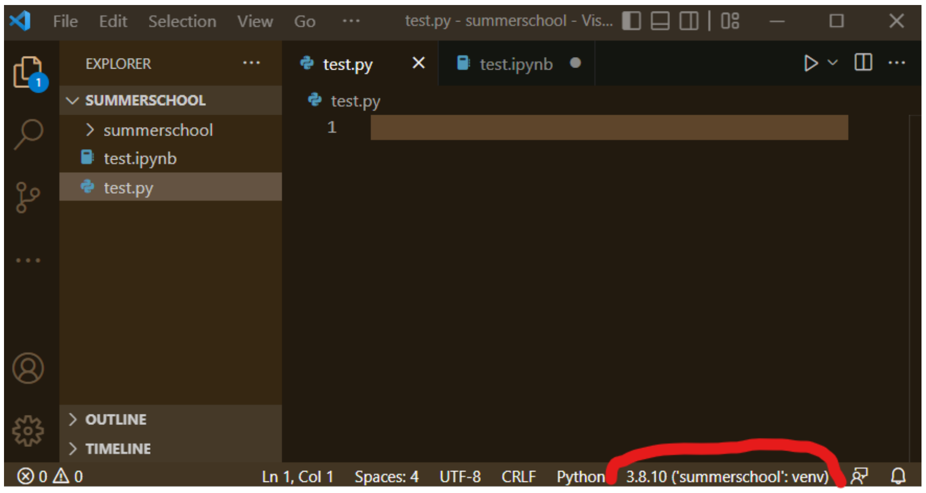


Figure 20: Selecting a virtual environment for a Python script.

Similarly, if you have a `.ipynb` file open, look in the upper right corner. You should see that the kernel associated with your virtual environment is selected (it will say something like `summerschool (Python 3.8.10)` for a venv environment). If it is not, then click there and select the interpreter that you would like to use.

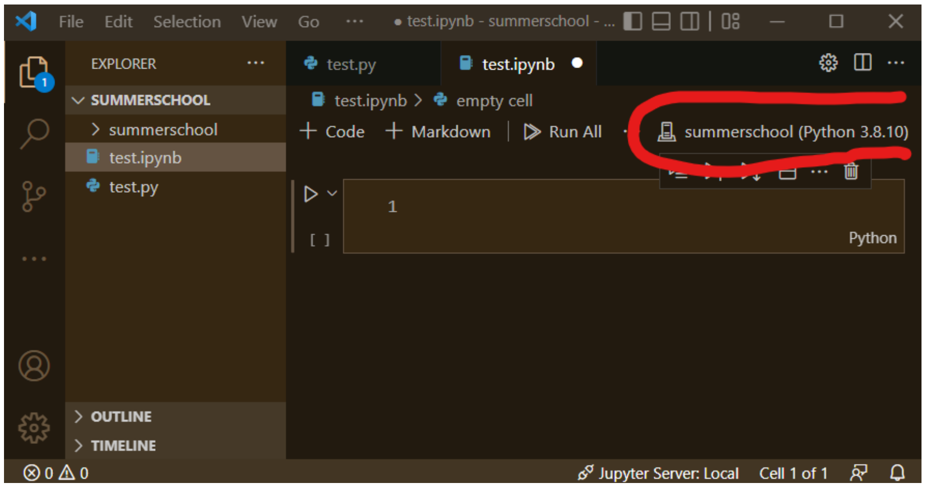
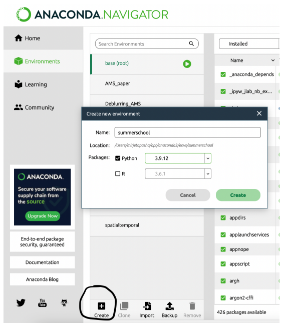


Figure 21: Selecting a kernel for a Jupyter notebook.

1.3.3 Other

You can create python (.py and .ipynb) projects directly from Anaconda as well. Once you install and open Anaconda Navigator as shown in Figure 1, you can click on Environments icon, that is located on the top left corner and then click Create button to create an environment. After selecting the name and the python version from the dropdown menu, click the green button Create.


Create new environment

Name: summerschool

Location: /Users/mirjetapasha/opt/anaconda3/envs/summerschool

Packages: ☑ Python 3.9.12

☐ R 3.6.1

Figure 22: Create a new environment in Anaconda Navigator

After your environment is created, you can go to Home (located on the upper left corner of Anaconda) and select your environment from the dropdown menu as shown in Figure 23.

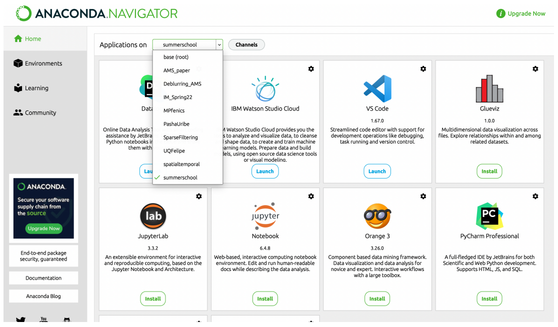


Figure 23: Select a new environment in Anaconda Navigator

With the new environment activated, you can install or launch Notebook, JupyterLab, or VsCode from Anaconda Navigator.

Remarks Other platforms can be used for Python, for instance Spyder, PyCharm, Atom, and Sublime Text, to mention a few. In this short tutorial, we limit our discussion to the above mentioned platforms. If you are interested in other platforms, we suggest exploring the following links:

1. Spyder: `https://www.spyder-ide.org`

2. PyCharm: `https://www.jetbrains.com/pycharm/`
3. Atom: `https://atom.io/`
4. Sublime Text: `https://www.sublimetext.com/3`

References

[1] D. Rolon-Mérette, M. Ross, T. Rolon-Mérette, and K. Church. Introduction to Anaconda and Python: Installation and setup. *Python for research in psychology*, 16(5):S5–S11, 2016.

[2] M. F. Sanner et al. Python: A programming language for software integration and development. *J Mol Graph Model*, 17(1):57–61, 1999.

[3] G. Van Rossum et al. Python programming language. In *USENIX annual technical conference*, volume 41, pages 1–36. Santa Clara, CA, 2007.

[4] G. VanRossum and F. L. Drake. *The python language reference.* Python Software Foundation Amsterdam, Netherlands, 2010.