Getting started with ILWISPy, Python and Jupyter Notebooks

Ben Maathuis, Bas Retsios, Martin Schouwenburg, Willem Nieuwenhuis and Lichun Wang.
Faculty ITC. University of Twente. May 2025.

Corresponding authors: Ben Maathuis (b.h.p.maathuis@utwente.nl) and Bas Retsios (v.retsios@utwente.nl)

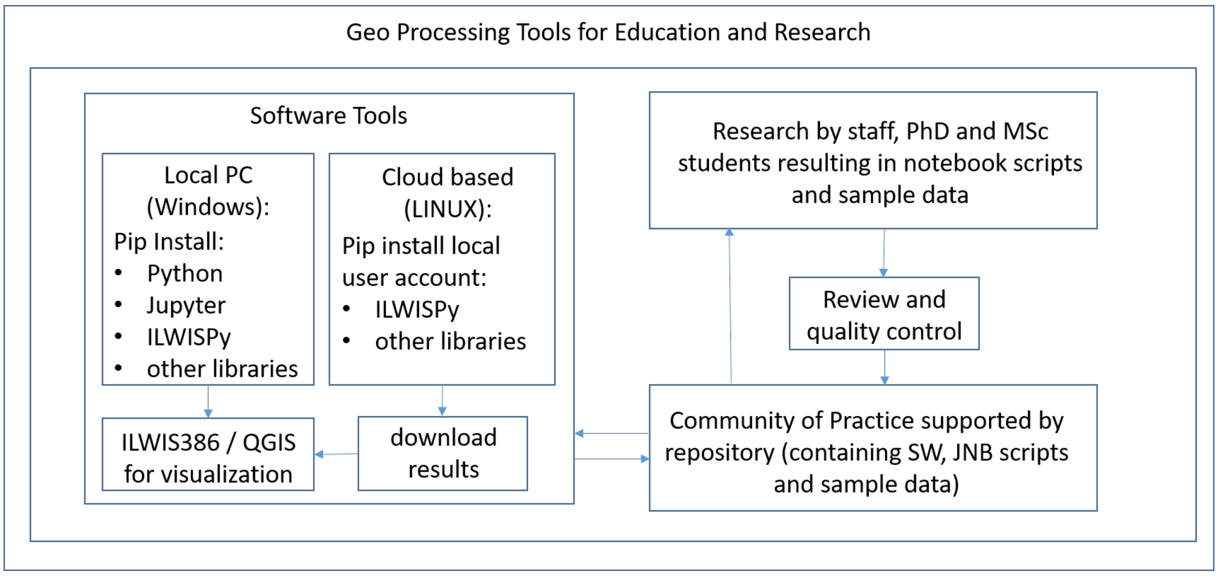
1. Introduction: Why ILWISPy?

ILWISPy is a Geo-Processing Tool which can be used as a site package under Python. The main objective of ILWISPy is to '*achieve more with less coding*' and '*increase flexibility*', e.g. also being able to operate ILWISPy within a Virtual Research Environment using Linux as operating system. ILWISPy has a large library of commonly used GIS and RS operations which can be easily coded through single line statements. The core is based on C++, enabling good computing performance. This capability, together with other Python site packages, offers great flexibility in data retrieval, (pre-) processing and analysis, far exceeding those offered through current Graphical User Interface (GUI) Geo-processing tools.

ILWISPy can be directly imported in Python, but can also be used within a Jupyter Notebook, offering the capability of markdown text with explanations and subsequent coding fields to execute certain operations, which for educational purposes offers nice opportunities. For research applications these notebooks and data used also contribute to 'open science' showing all processing steps conducted.

For quick display of results or more advanced visualizations use can be made of exiting free GUI based software tools like ILWIS386 or QGIS. The overall concept is provided in Figure 1.

Figure 1: Role of ILWISPy in education and research


Geo Processing Tools for Education and Research
Software Tools
Local PC (Windows):
Pip Install:
• Python
• Jupyter
• ILWISPy
• other libraries
Cloud based (LINUX):
Pip install local user account:
• ILWISPy
• other libraries
ILWIS386 / QGIS for visualization
download results
Research by staff, PhD and MSc students resulting in notebook scripts and sample data
Review and quality control
Community of Practice supported by repository (containing SW, JNB scripts and sample data)

This document describes installation from scratch, without previous installation of python, conda or mini-conda. If (mini-) conda has been installed already then a slightly different installation procedure should be followed. In case of questions, additional information can be obtained from the authors.

ILWIS under Python (ILWISPy) is available for Python versions 3.6 and higher, for both Windows and Linux.

All ILWISPy resources are available at: https://filetransfer.itc.nl/pub/52n/ilwis_py/. The following structure is adhered to:

- **/notebooks_V2**: Notebooks in Jupyter Notebook format and as HTML. The JNB format files can be directly used, just copy and paste these in your active Jupyter notebook folder. The HTML files show the full processed content of the notebook, to show the user what is to be expected and the output that will be created by execution of the notebook. Currently a number of notebooks are available providing an introduction to ILWISPy, and the use of ILWISPy for image and time series processing as well as retrieving and processing of data from openEO portals;
- **/sample_data_V2**: data used within these notebooks

Below the procedure to install Python (example version user here is Python version 3.8.10), Jupyter Notebook and ILWISPy are described.

Additional information is available on GitHub, see: https://github.com/Ilwis/IlwisObjects#readme, including also known installation issues under Linux.

2. Download and install Python for Windows

Visit the Python download pages at: https://www.python.org/downloads/. Here the latest Python full bug fix release of Python 3.8 with binary installers is used as example (Python3.8.10).

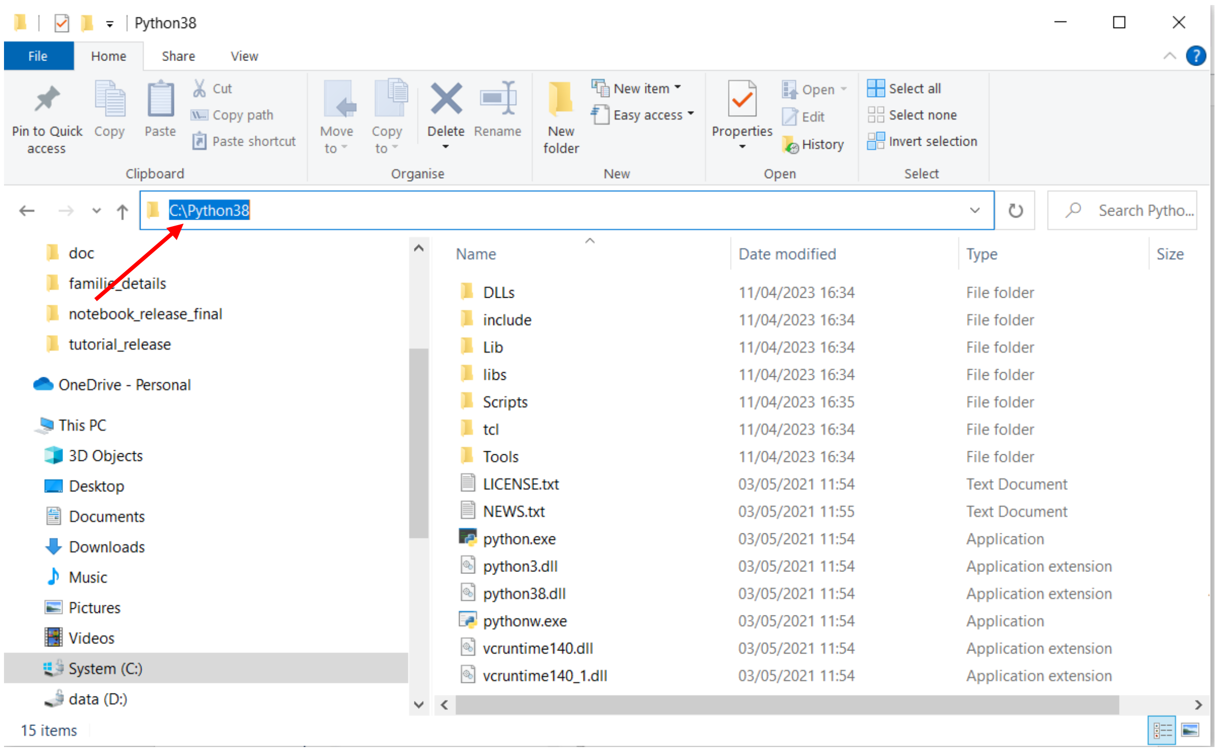
At the moment of preparation of this document select under “Looking for a specific release?” Python Release version “Python 3.8.10” (https://www.python.org/downloads/release/python-3810/) and download the ‘Windows x86-64 executable installer’.

To install Python on your system (assumed here a Windows – 64 bits operating system) :

1. Double click with the mouse the file “python-3.8.10-amd64.exe”, activate the option “Add Python 3.8 to PATH” and select the installation option “Customize installation”;
2. In the next window “Optional Features” continue with “Next”;
3. Use the default settings in the “Advanced Options” window, but change under the “Customize Install Location” the folder settings to: “C:\Python38”.
4. Press “Install” and accept the system request to make changes to your configuration;
5. After installation Python will indicate if the installation was successful and press “Close”.

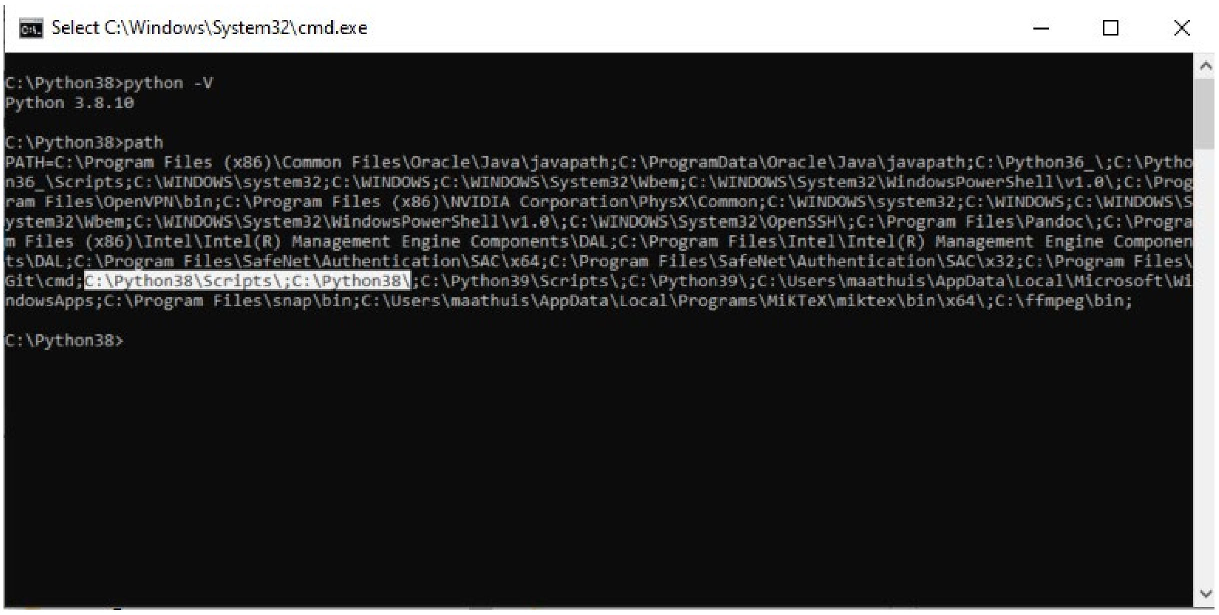
Navigate with your system “Explorer” to the newly created folder “C:\Python38” and click on the icon in front of the folder settings (see figure 2), this should now receive a blue color, subsequently type, using the keyboard “cmd”, to open the command line interpreter in the folder “C:\Python38”. Type the following syntax: “python -V” (note case sensitivity!) and you should get the information about the version installed. Next type “path”, to check if the path is set to the python folder, here: ”c:\python38”, see also figure 3.

Figure 2: Newly created Python 3.8 folder content showing the executable “python.exe”



Name	Date modified	Type	Size
DLLs	11/04/2023 16:34	File folder	
include	11/04/2023 16:34	File folder	
Lib	11/04/2023 16:34	File folder	
libs	11/04/2023 16:34	File folder	
Scripts	11/04/2023 16:35	File folder	
tcl	11/04/2023 16:34	File folder	
Tools	11/04/2023 16:34	File folder	
LICENSE.txt	03/05/2021 11:54	Text Document	
NEWS.txt	03/05/2021 11:55	Text Document	
python.exe	03/05/2021 11:54	Application	
python3.dll	03/05/2021 11:54	Application extension	
python38.dll	03/05/2021 11:54	Application extension	
pythonw.exe	03/05/2021 11:54	Application	
vcruntime140.dll	03/05/2021 11:54	Application extension	
vcruntime140_1.dll	03/05/2021 11:54	Application extension	

Figure 3: Check Python version installed and your system path settings



Select C:\Windows\System32\cmd.exe

```
C:\Python38>python -V
Python 3.8.10

C:\Python38>path
PATH=C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\ProgramData\Oracle\Java\javapath;C:\Python36_\;C:\Pytho
n36_\Scripts;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\Prog
ram Files\OpenVPN\bin;C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\S
ystem32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\WINDOWS\System32\OpenSSH\;C:\Program Files\Pandoc\;C:\Progra
m Files (x86)\Intel\Intel(R) Management Engine Components\DAL;C:\Program Files\Intel\Intel(R) Management Engine Componen
ts\DAL;C:\Program Files\SafeNet\Authentication\SAC\x64;C:\Program Files\SafeNet\Authentication\SAC\x32;C:\Program Files\
Git\cmd;C:\Python38\Scripts\;C:\Python38\;C:\Python39\Scripts\;C:\Python39\;C:\Users\maathuis\AppData\Local\Microsoft\Wi
ndowsApps;C:\Program Files\snap\bin;C:\Users\maathuis\AppData\Local\Programs\MiKTeX\miktex\bin\x64\;C:\ffmpeg\bin;

C:\Python38>
```

3. Installing ILWISPy for Windows users

Continue or open again the command prompt window, see figure 2, and ensure that you are in the python38 folder. From the command prompt, type the following command:

c:\python38> ***python –m pip install ilwis***

Note: replace ‘drive:\local folder’ with the location of your actual drive:\folder on your system

Press <Enter> and note the response provided in the command line window. If you get an 'error logging' because of an older 'PIP' version, to upgrade 'PIP', type following command:

c:\python38> ***python -m pip install --upgrade pip***

You can reinstall ILWISPy using the following command:

c:\python38> ***python -m pip install ilwis --force-reinstall***

If you want to upgrade your ILWISPy version and download the most recent version use the following statement.:

c:\python38> ***python -m pip install ilwis --upgrade***

To check if the installation was successful, start a new Python session. In the Python command line shell, also see figure 2 above and type the following expressions:

C:\Python38> 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.

➢ import ilwis

- ilwis.version()

    '1.0 build 20250513'

- quit()

To close Python press <Ctrl>Z and press <Enter> or type: exit() or quit()

Note that ILWISPy is installed under the Python38 folder, in '\Lib\site-packages\ilwis'.

To uninstall ILWISPy type the following command:

C:\Python38> ***Python –m pip uninstall Ilwis***

4. Installing ILWISPy for Linux users

Installation instructions to install ILWISPy for Linux, using Python version 3.8, is further elaborated upon below as example. To install ILWISPy use the following set of commands. Open a "Terminal" session and provide the following command:

python -m pip install ilwis

Could be that you see a logging error, this is likely due to an older version of 'PIP', to upgrade 'PIP' type:

python -m pip install --upgrade pip

If you want to re-install ILWISPy, using the upgraded PIP version, type:

python -m pip install ilwis --force-reinstall

To check if the installation procedure was successful, start a new Python session or restart the python kernel and import ILWISPy:

In the 'Terminal', type on the command line:

```
$ python

Python 3.8.10 (default, Jun 22 2022, 20:18:18)

[GCC 9.4.0] on linux

Type "help", "copyright", "credits" or "license" for more information.
```

- import ilwis
- ilwis. version() #check version installed

    1.0 build 20250513

- quit()

5. Installing frequently used Python libraries in conjunction with ILWISPy

Python makes use of some libraries which have to be installed separately. A library is a collection of pre-combined codes that can be used iteratively to reduce the time required to code. They are particularly useful for accessing the pre-written frequently used codes, instead of writing them from scratch every single time. Similar to the physical libraries, these are a collection of reusable resources, which means every library has a root source. This is the foundation behind the numerous open-source libraries available in Python.

Pip is a useful program to install additional libraries in Python. To ensure you are working with the latest version of pip, type the following command from within the python38 folder:

python –m pip install - - upgrade pip

To install additional libraries use the following command from within the python folder:

python –m pip install "some library"

Some useful libraries for data science are:

- **NumPy**: Numerical Python is the fundamental package for numerical computation in Python; it contains a powerful N-dimensional array object.
- **Matplotlib**: Matplotlib is a plotting library for Python. Because of the graphs and plots that it produces, it's extensively used for data visualization.
- **Pandas**: Python data analysis is a must in the data science life cycle. It is the most popular and widely used Python library for data science, along with NumPy in Matplotlib.
- **SciPy**: Scientific Python is another free and open-source Python library for data science that is extensively used for high-level computations.

If not already available, install the libraries mentioned above, replacing "some library" with the names mentioned above, like numpy, matplotlib, pandas, sklearn (short for 'scikit-learn'). If additional libraries are going to be used at a later stage, you should install them before you continue.

Now with Python, ILWISPy and the required libraries installed we continue with the installation of the Jupyter notebook.

Note that when using a virtual research environment on a cloud service, mostly the site packages mentioned above are already installed, as well as the Jupyter Notebook environment which is discussed below.

6. Installing Jupyter Notebook for Windows users

Installing Jupyter notebook under Python using pip install:

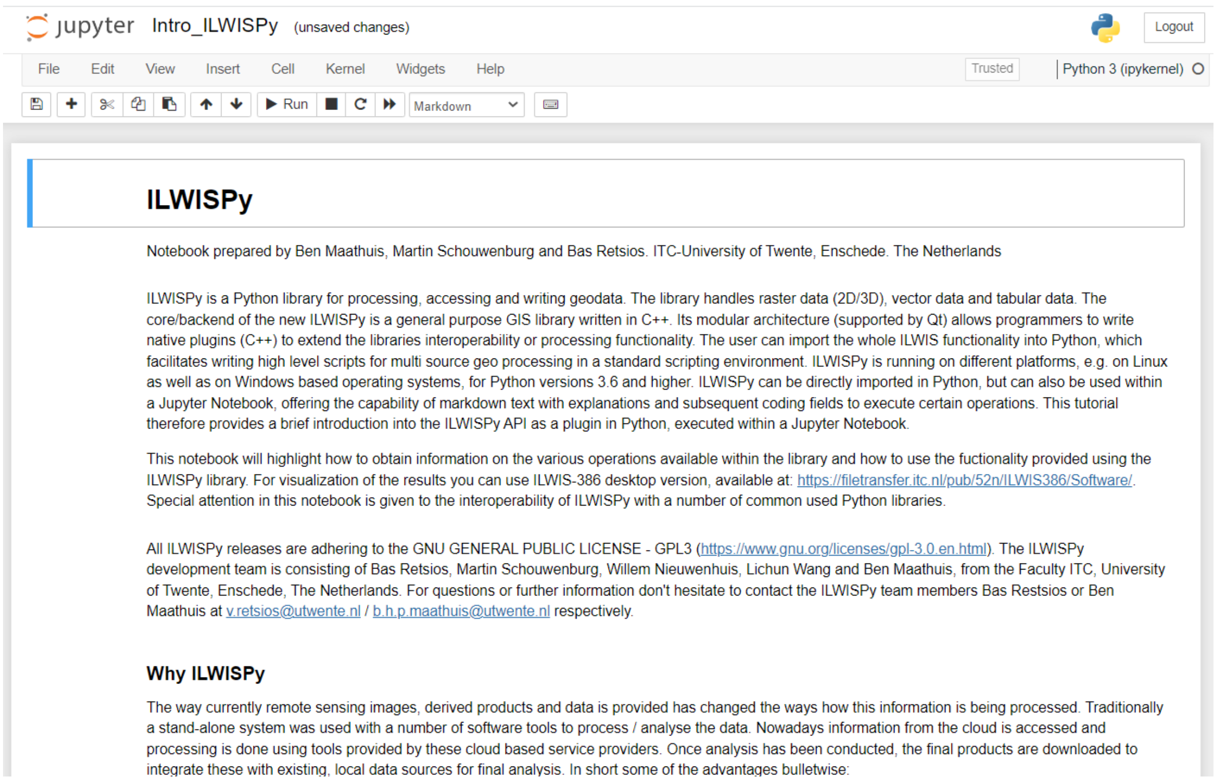
- The command to install Jupyter is similar as given above: "python -m pip install jupyter". Type this command in the cmd window in the folder "c:\python38";
- First a number of files are downloaded and then the Packages are installed;
- Once finished the installation create a new folder on the root of your c:\drive, e.g. "Jupyter" and navigate to this folder in an identical manner as shown in figure 1, but now using the folder "c:\jupyter";

- From the folder "c:\jupyter" type the following command to launch Jupyter notebook using command-line: "jupyter notebook". Your default browser is opened, e.g. "Chrome" and you can navigate to the respective JNB files or sub-folders were your notebooks are residing;

Navigate to: https://filetransfer.itc.nl/pub/52n/ilwis_py/notebooks/. Copy the sample notebooks, e.g. "Intro_ILWISPy.ipynb" to the folder "C:\jupyter".

Click on the notebook "Intro_ILWISPy.ipynb" to get started. You should get the content as of figure 4.

Figure 4: Starting the Jupyter notebook using the 'Intro_ILWISPy' notebook


jupyter Intro_ILWISPy (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Markdown

ILWISPy

Notebook prepared by Ben Maathuis, Martin Schouwenburg and Bas Retsios. ITC-University of Twente, Enschede. The Netherlands

ILWISPy is a Python library for processing, accessing and writing geodata. The library handles raster data (2D/3D), vector data and tabular data. The core/backend of the new ILWISPy is a general purpose GIS library written in C++. Its modular architecture (supported by Qt) allows programmers to write native plugins (C++) to extend the libraries interoperability or processing functionality. The user can import the whole ILWIS functionality into Python, which facilitates writing high level scripts for multi source geo processing in a standard scripting environment. ILWISPy is running on different platforms, e.g. on Linux as well as on Windows based operating systems, for Python versions 3.6 and higher. ILWISPy can be directly imported in Python, but can also be used within a Jupyter Notebook, offering the capability of markdown text with explanations and subsequent coding fields to execute certain operations. This tutorial therefore provides a brief introduction into the ILWISPy API as a plugin in Python, executed within a Jupyter Notebook.

This notebook will highlight how to obtain information on the various operations available within the library and how to use the fuctionality provided using the ILWISPy library. For visualization of the results you can use ILWIS-386 desktop version, available at: https://filetransfer.itc.nl/pub/52n/ILWIS386/Software/. Special attention in this notebook is given to the interoperability of ILWISPy with a number of common used Python libraries.

All ILWISPy releases are adhering to the GNU GENERAL PUBLIC LICENSE - GPL3 (https://www.gnu.org/licenses/gpl-3.0.en.html). The ILWISPy development team is consisting of Bas Retsios, Martin Schouwenburg, Willem Nieuwenhuis, Lichun Wang and Ben Maathuis, from the Faculty ITC, University of Twente, Enschede, The Netherlands. For questions or further information don't hesitate to contact the ILWISPy team members Bas Restsios or Ben Maathuis at v.retsios@utwente.nl / b.h.p.maathuis@utwente.nl respectively.

Why ILWISPy

The way currently remote sensing images, derived products and data is provided has changed the ways how this information is being processed. Traditionally a stand-alone system was used with a number of software tools to process / analyse the data. Nowadays information from the cloud is accessed and processing is done using tools provided by these cloud based service providers. Once analysis has been conducted, the final products are downloaded to integrate these with existing, local data sources for final analysis. In short some of the advantages bulletwise:

7. Create a Jupyter shortcut on your Windows system

From https://filetransfer.itc.nl/pub/52n/ilwis_py/temporary_tools/ copy the files "Jupyter.lnk" (shortcut) and "jnb_ico" to the folder 'c:\jupyter'. Ensure that the extension of the file Jupyter.lnk remains the same (could be renamed to download!)

Within your folder 'c:\jupyter', drag the shortcut "Jupyter.lnk" to the desktop. Right click the shortcut, inspect the properties, ensure that the "Start in" folder is correct, here: c:\jupyter. Also click the button "Change Icon" and navigate to the folder 'c:\jupyter' and select the file "jnb.ico".

Double click the shortcut to start a 'Jupyter Notebook' session. If you have installed the notebooks in a different folder, right click the shortcut with the mouse and change the properties accordingly, e.g. the properties under "Start in". Now close all active windows.

8. Using Jupyter Notebooks

A notebook integrates code and its output into a single document that combines visualizations, narrative text, mathematical equations, and other rich media. In other words: it's a single document where you can run code, display the output, and also add explanations, formulas, charts, and make your work more transparent, understandable, repeatable, and shareable.

Jupyter's Notebooks and dashboard are web apps, and Jupyter starts up a local Python server to serve these apps to your web browser, making it essentially platform-independent and opening the door to easier sharing your code developed on the web.

Start Jupyter NoteBook (JNB) by double clicking the shortcut created and open once more the notebook "1.1_Python_fist_steps.ipynb". The dashboard's interface is mostly self-explanatory. Each .ipynb file is a text file that describes the contents of your notebook in a format called JSON. Each cell and its contents, including image attachments that have been converted into strings of text, is listed therein along with some metadata.

There are two items appearing in the main menu that you should notice: cells and kernels. These are key both to understanding Jupyter and to what makes it more than just a word processor.

A kernel is a "computational engine" that executes the code contained in a notebook document. A cell is a container for text to be displayed in the notebook or code to be executed by the notebook's kernel. Cells form the body of a notebook. There are two main cell types:

- A *Code* cell contains code to be executed in the kernel. When the code is run, the notebook displays the output below the code cell that generated it.
- A *Markdown* cell contains text formatted using Markdown and displays its output in-place when the Markdown cell is run.

You can always tell the difference between Code and Markdown cells because code cells have that label on the left and Markdown cells do not. The "In" part of the label is simply short for "Input," while the label number indicates when the cell was executed on the kernel. In a Jupyter Notebook, there is always one "active" cell highlighted with a border whose color denotes its current mode:

- Green outline — cell is in "edit mode"
- Blue outline — cell is in "command mode"

To run a cell press the Shift + Enter keys simultaneously. There are plenty of other commands we can use. The best way to use them is with keyboard shortcuts. Keyboard shortcuts are a very popular aspect of the Jupyter environment because they facilitate a speedy cell-based workflow. Many of these are actions you can carry out on the active cell when it's in command mode.

Below, you'll find a list of some of Jupyter's keyboard shortcuts. You don't need to memorize them all immediately, but this list should give you a good idea of what's possible.

- Toggle between edit and command mode with Esc and Enter, respectively.
- Once in command mode:
    - Scroll up and down your cells with your Up and Down keys.
    - Press 'a' or 'b' to insert a new cell above or below the active cell.
    - 'm' will transform the active cell to a Markdown cell.
    - 'y' will set the active cell to a code cell.

    - ‘d’ + ‘d’ (d twice) will delete the active cell.
    - ‘z’ will undo cell deletion.
    - Hold Shift and press Up or Down to select multiple cells at once. With multiple cells selected, Shift + ‘m’ will merge your selection.
- Ctrl + Shift + -, in edit mode, will split the active cell at the cursor. If your keyboard has more than one key corresponding to -, and one zooms out instead of splitting cells, try using the other key.
- You can also click and Shift + Click in the margin to the left of your cells to select them.

From the “Help” menu also additional information about the various keyboard shortcuts can be obtained.

Markdown is a lightweight, easy to learn markup language for formatting plain text. Markdown is a lightweight and popular Markup language which is a writing standard for data scientists and analysts. It is often converted into the corresponding HTML by the Markdown processor which allows it to be easily shared between different devices and people.

Markup language is similar to Hypertext Markup Language(HTML) made of Markup tags, and it consists of the opening tag <tagname> and closing tag </tagname>. For additional information on Markdown, see: https://www.datacamp.com/community/tutorials/markdown-in-jupyter-notebook.

Behind every notebook runs a kernel. When you run a code cell, that code is executed within the kernel. Any output is returned back to the cell to be displayed. The kernel’s state persists over time and between cells — it pertains to the document as a whole and not individual cells.

For example, if you import libraries or declare variables in one cell, they will be available in another.

Most of the time when you create a notebook, the flow will be top-to-bottom. But it’s common to go back to make changes. When we do need to make changes to an earlier cell, the order of execution we can see on the left of each cell, such as ‘In [6]’ (in front of a cell), can help diagnose problems by seeing what order the cells have run in.

And if we ever wish to reset things, there are several useful options from the Kernel menu:

- Restart: restarts the kernel, thus clearing all the variables etc. that were defined.
- Restart & Clear Output: same as above but will also wipe the output displayed below your code cells.
- Restart & Run All: same as above but will also run all your cells in order from first to last.
- If your kernel is ever stuck on a computation and you wish to stop it, you can choose the Interrupt option.

Depending on the type of installation, Jupyter provides the option to change the kernel – programming languages. Many other languages, in addition to Python, may be used in the notebook. By creating a new notebook from the dashboard menu, by selecting Python, you are actually choosing which kernel to use. Here only the Python kernel is at your disposal.

Open the command line interpreter in the folder “C:\Python38”, see figure 2. Type the following syntax: “jupyter kernelspec list” and you should get the information about the kernel installed, here ‘python3’.

9. Use of sample ILWISPy Jupyter Notebooks

A number of notebooks have been prepared to demonstrate the capability offered by the ILWISPy library in conjunction with other python site packages, like numpy, matplotlib, etc. These notebooks can be downloaded from: https://filetransfer.itc.nl/pub/52n/ilwis_py/notebooks_V2/. For a new ILWISPy user best sequence of reviewing / running the current available notebooks is:

- Intro_ILWISPy - provides general introduction to ILWISPy
- Intro_RS_ILWISPy - how you can use ILWISPy for basic image processing
- Intro_RS_ILWIS_Py_classification - import and preprocessing of RS images and subsequent use in machine learning (using Science Kit)
- Intro_timeseries_ILWISPy - introduction in use of time series data and time series data processing
- Intro_openEO_ILWISPy - retrieve data from cloud services and conduct the necessary data processing using ILWISPy

The Notebooks are also available in a HTML format. These show you the results of the code fields executed, so you can compare your results obtained by execution of the notebook with those shown in the HTML version.

10. Python references and useful online resources

Allen Downey (2015): Think Python. How to Think Like a Computer Scientist. 2nd Edition, Version 2.4.0. Green Tea Press. Available at: https://greenteapress.com/thinkpython2/thinkpython2.pdf

Attard, Guillaume (2021): An Intro to the Earth Engine Python API. Available at: https://developers.google.com/earth-engine/tutorials/community/intro-to-python-api-guiattard

Data Carpentry. Programming with Python. Available at https://carpentries-incubator.github.io/python-novice-programming-gapminder/

Khan, S. (2020): Learn Basic Statistics with Python. Available at: https://medium.com/insights-school/learn-basic-statistics-with-python-cc0f45275929

Sat Kumar Tomer (2011): Python in Hydrology. Green Tea Press. Available at: https://www.greenteapress.com/pythonhydro/pythonhydro.pdf

Stojiljkovic, M. Python Statistics Fundamentals: How to Describe Your Data. Available at: https://realpython.com/python-statistics/

Data Carpentry: https://carpentries-incubator.github.io/python-novice-programming-gapminder/