Python installation guide

This document was created for the X-DIGI Hackathons and Python training. Please go through all the steps so that come prepared to the events

In case you need assistance there are some very well maintened Yammer groups where people can answer most of your questions:

- Python community in Shell
- Python baby steps
- GeoIO

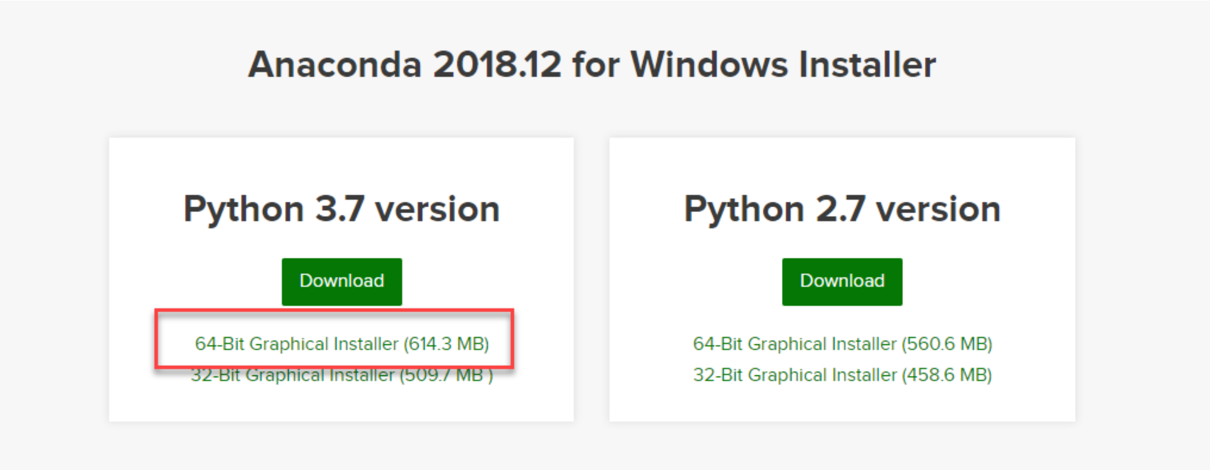
One of the reasons why Python is so popular is because of the massive user community. If you struggle with something, you'll most likely find the answer online.

In this guide we'll be using Python most popular distribution: Anaconda. If it's not clear to you what is the difference between Python Implementation vs. Python distribution vs. Python itself? Read the answer on this post

Install Anaconda

Anaconda is available for all major platforms, e.g. Windows, Linux and Mac.

- Go to https://www.anaconda.com/download



Anaconda 2018.12 for Windows Installer

Python 3.7 version

Download

64-Bit Graphical Installer (614.3 MB)

32-Bit Graphical Installer (509.7 MB)

Python 2.7 version

Download

64-Bit Graphical Installer (560.6 MB)

32-Bit Graphical Installer (458.6 MB)

- Choose platform and Python 3.7 and download
- Run the installer after accepting the license
- **You don't need admin privileges to install Anaconda on your machine**
- **Windows**: recommended location: C:/Apps/Software/Anaconda

Once Anaconda is installed you will find:

- **Anaconda Navigator** : a GUI (graphical user interface) used to manage environment and packages through **conda** a powerful package management tool
- **Jupyter Notebooks**. : "[…]*incredibly powerful tool for interactively developing and presenting science projects. A notebook integrates code and its output into a single*

document that combines visualisations, narrative text, mathematical equations, and other rich media[...]" link

- **Anaconda Prompt:** a normal prompt (e.g. windows prompt , Linux terminal,...) that simply makes sure you can use conda in its proper environment

After installing anaconda, open an **Anaconda prompt** from the Windows Start menu, then type or copy-paste these lines, one at a time:

```
conda config --set proxy_servers.http http://proxy-eu.shell.com:8080
conda config --set proxy_servers.https https://proxy-eu.shell.com:8080
conda config --set ssl_verify false
```

note the proxy server address various with location. The above one is for europe. Ask for the correct address from a local IT person or colleagues who has been using Anaconda.

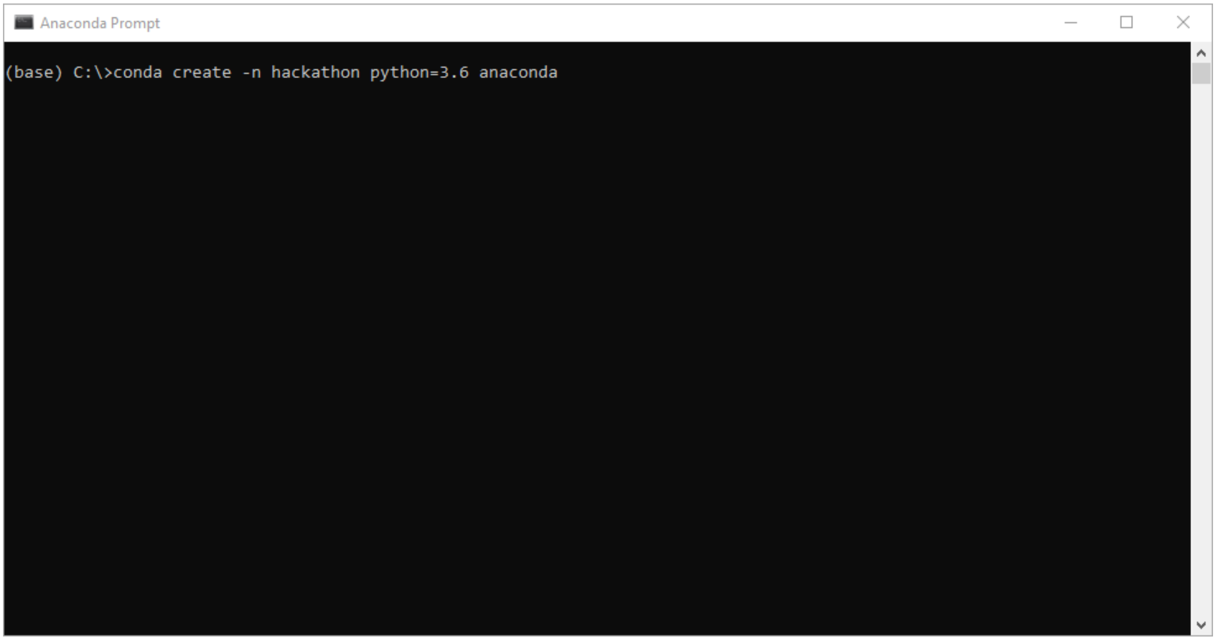
1-Setting up environments

We're almost there! Technically you could start now writing Python code. However, let's do something else first. One of the main power of Python distributions (like Anaconda) is the ability to simply create separated and independent working environments. Why use a sperate environment?

- Short version: best practice that will save you from a **lot** of frustration in the long run
- Long version: read this

Now that you're convinced that it's a good idea to do so let's create our first dedicated environment!

1- Start the Anaconda Prompt from the start menu
2- Type the following line (you can copy and paste) :
conda create –n hackathon python=3.6 anaconda



```
(base) C:\>conda create -n hackathon python=3.6 anaconda
```

3- Press enter and voila! A whole lot of things will appear in the terminal and after a few minutes your environment should be ready.

While it's running let's unpack what we've done in the command line from point 2:

- **Conda**: we're calling Conda the packet management tool that is installed with Anaconda
- **Create**: we're telling Conda that we want to create a new environment
- **-n**: stands for "name", the name of our future environment
- Hackathon: the name of our environment. Change this variable the way you want
- **Python=3.6**: we're telling conda which version of Python we want to use in this environment. Those of you that followed are now wondering why we're using 3.6 when we downloaded Python 3.7? Well spotted! In order to easily read-in .vt/.int files we'll be using a Shell Python library name GeoIO which for now (March 19) does not work on 3.7.

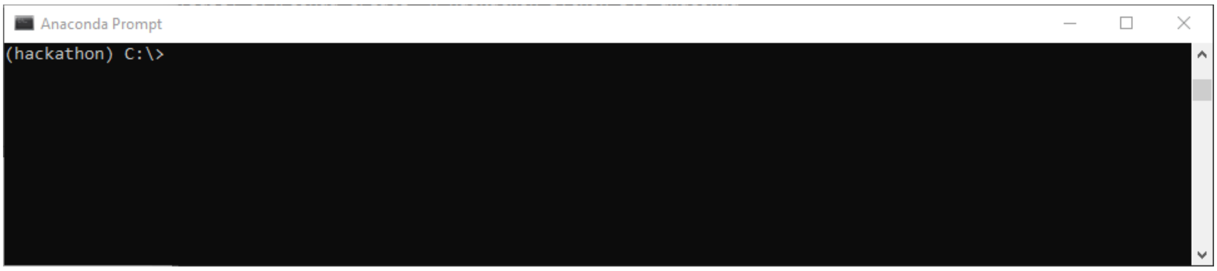
Anaconda prompt is extremely powerful, and it is way easier to use than you think. Here's a very extensive Cheat sheet that you can print and have look to if need be.

Installing new packages in your environment

Now it's time to add some domain specific packages to our environment. Before we do that, we need to activate our environment by typing the following command in the Anaconda prompt:

conda activate hackathon

Your prompt should now look like that



```
(hackathon) C:\>
```

*If you want to move to another environment. Simply type **conda deactivate hackathon** Then activate the other environment*

What it means is that every package you will now install will be confined to this environment. Let's install 2 packages that you might find useful for the GeoComputing course and/or the Hackathon:

- **Equinor's Segy handling package:** simply type the following line in the prompt (make sure you're in the correct environment)
  pip install segyio
  In case it doesn't run, it's most likely a proxy issue so try typing the following:
  pip --proxy proxy-asia-pac.shell.com:8080 install segyio

- **GeoIO:** the latest installation of GeoIO can be found at http://ndidocs.shell.com/GeoIO/install.html

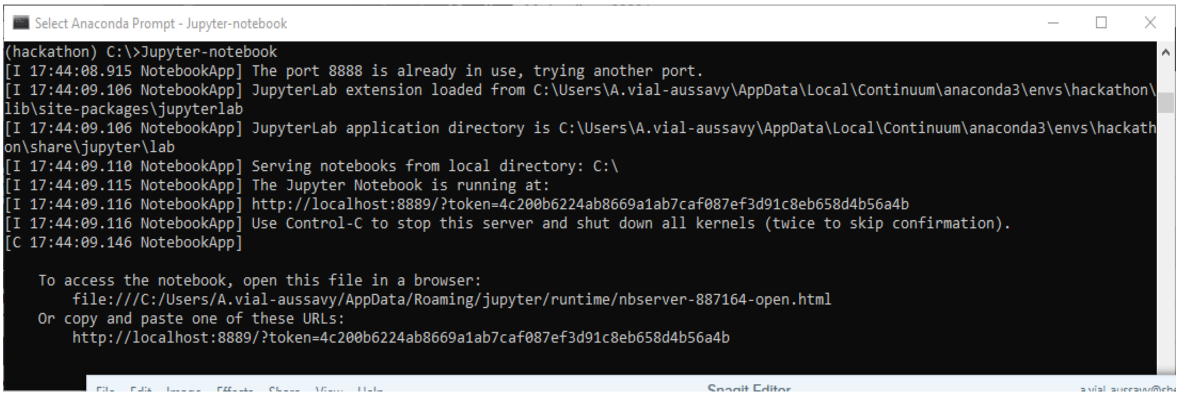
You might want to install other packages to your environment but in a nutshell the line

conda install package-name should work most of the time. Have a look here for more details

Time to start coding

Now that we have a working environment and that we have all the useful packages it's time to open the coding environment. There's plenty of editors available on the market but for simplicity we'll start using Jupyter-notebook. Simply type the following line:

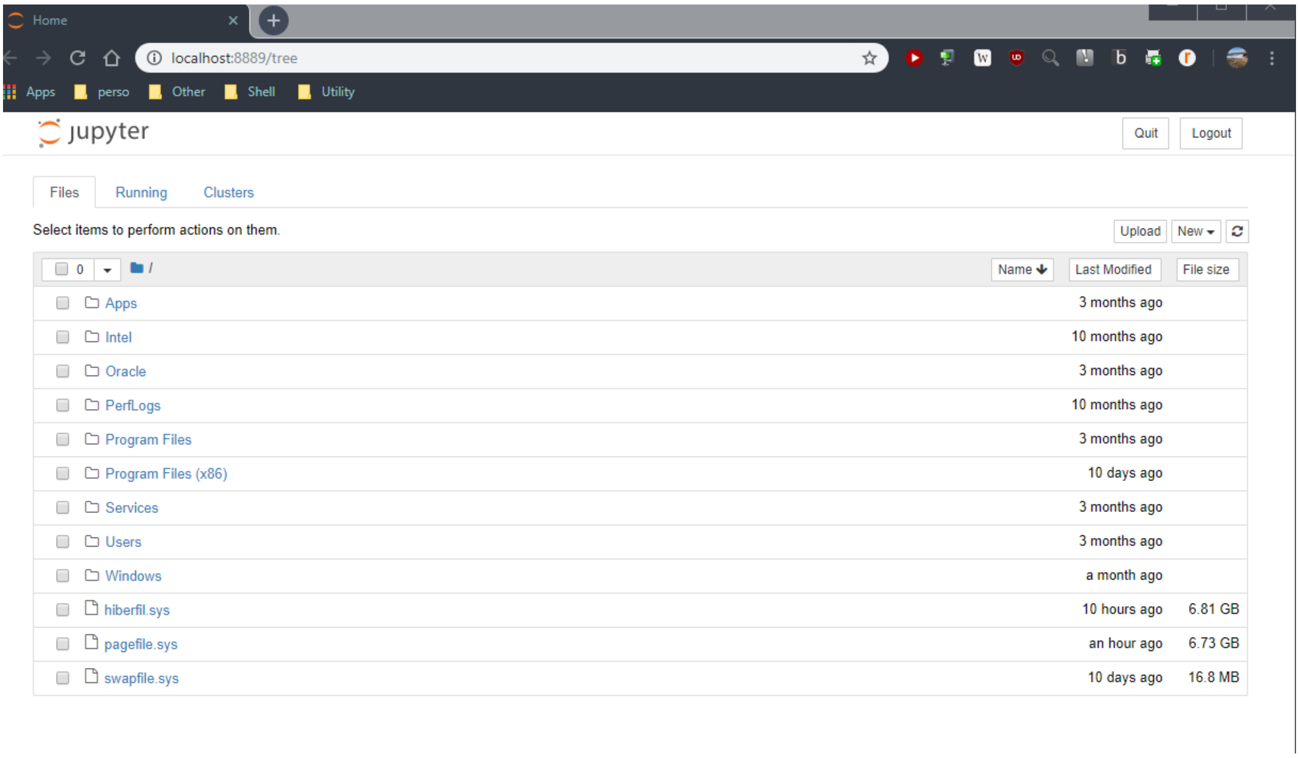
Jupyter-notebook



```
(hackathon) C:\>Jupyter-notebook
[I 17:44:08.915 NotebookApp] The port 8888 is already in use, trying another port.
[I 17:44:09.106 NotebookApp] JupyterLab extension loaded from C:\Users\A.vial-aussavy\AppData\Local\Continuum\anaconda3\envs\hackathon\
lib\site-packages\jupyterlab
[I 17:44:09.106 NotebookApp] JupyterLab application directory is C:\Users\A.vial-aussavy\AppData\Local\Continuum\anaconda3\envs\hackath
on\share\jupyter\lab
[I 17:44:09.110 NotebookApp] Serving notebooks from local directory: C:\
[I 17:44:09.115 NotebookApp] The Jupyter Notebook is running at:
[I 17:44:09.116 NotebookApp] http://localhost:8889/?token=4c200b6224ab8669a1ab7caf087ef3d91c8eb658d4b56a4b
[I 17:44:09.116 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 17:44:09.146 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/A.vial-aussavy/AppData/Roaming/jupyter/runtime/nbserver-887164-open.html
    Or copy and paste one of these URLs:
        http://localhost:8889/?token=4c200b6224ab8669a1ab7caf087ef3d91c8eb658d4b56a4b
```

Your favorite internet browser should open and look like the picture below. Note that the URL reads localhost which means that the environment is running on your machine and not on a cloud or a remote instance.



Select items to perform actions on them.

Name	Last Modified	File size
Apps	3 months ago	
Intel	10 months ago	
Oracle	3 months ago	
PerfLogs	10 months ago	
Program Files	3 months ago	
Program Files (x86)	10 days ago	
Services	3 months ago	
Users	3 months ago	
Windows	a month ago	
hiberfil.sys	10 hours ago	6.81 GB
pagefile.sys	an hour ago	6.73 GB
swapfile.sys	10 days ago	16.8 MB

OPTIONAL

By default, your browser is IE but if you want to use Chrome simply do the following in the Anaconda prompt:

jupyter notebook --generate-config

The jupyter_notebook_config.py file generated is situated in "**C:\Users\YourName\.jupyter**" folder.

Open it using a text editor and change:

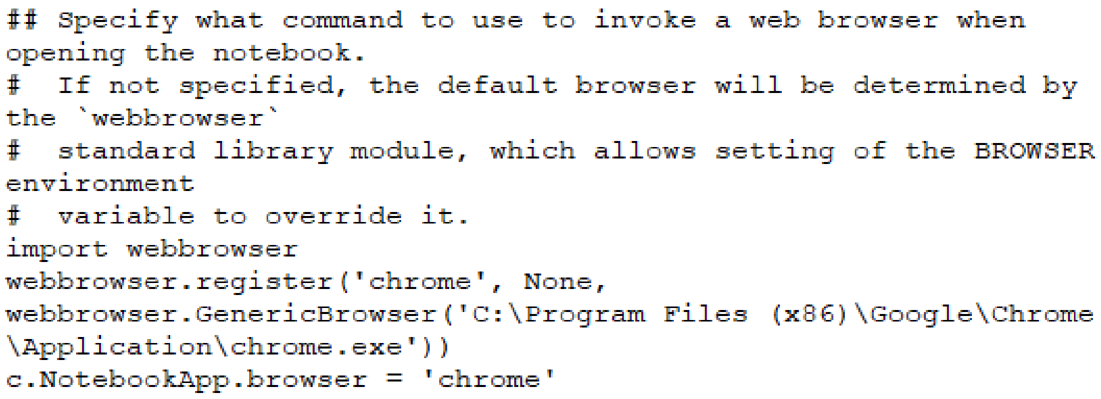
#c.NotebookApp.browser = ''

To (don't forget the #)

import webbrowser

webbrowser.register('chrome', None, webbrowser.GenericBrowser(u'C:\\Program Files (x86)\\Google\\Chrome\\Application\\chrome.exe'))

c.NotebookApp.browser = 'chrome'



```
## Specify what command to use to invoke a web browser when
opening the notebook.
#  If not specified, the default browser will be determined by
the `webbrowser`
#  standard library module, which allows setting of the BROWSER
environment
#  variable to override it.
import webbrowser
webbrowser.register('chrome', None,
webbrowser.GenericBrowser('C:\Program Files (x86)\Google\Chrome
\Application\chrome.exe'))
c.NotebookApp.browser = 'chrome'
```

and save the file.

Now execute the jupyter-notebook command and the set browser will be used

Get a GitHub account (Optional but strongly recommended)

You want a common space to store your code? You want a place where all the team members can work together? Search no more and just follow these simple steps.

1. Register a GitHub account at github.com with your shell email.
2. After that, send Yuanzhong Fan (yuanzhong.fan@shell.com) the handle (username) and he will add you the Shell enterprise access.
3. Then you will receive an invitation email from shell account @sdu-rds and follow the instruction to setup the 2-factor authentication

Note: There were cases some people's outlook block emails from @github.com. To be sure, you can add @github.com in your outlook safe sender list.

4. Watch those youtube videos on git and github. On how to create a repo in Shell account, watch the video https://xdigi.shell.com/x-digi/how-to-create-a-repo-in-shell-enterprise-account-and-set-access-permission/