

# Python Short-course Study Guide

Last updated: Franco Pretorius (2019-07-29)

## 1 Introduction

The aim of this course is to teach the very basics of Python programming for engineering purposes. This will make it easier for you to navigate the online documentation and tutorials in future. These instructions assume you have a Windows operating system. The most important instructions in every section below are given in a blue box.

## Contents

1	Introduction.....	1
2	Installing Python.....	2
3	Opening Jupyter .....	2
4	Version control.....	3
4.1	GitHub Desktop .....	4
4.1.1	Cloning a repository.....	4
4.1.2	Updating a repository .....	4
4.2	Git Bash.....	4
4.2.1	Cloning a repository.....	5
4.2.2	Updating a repository .....	5
5	Customisation .....	5
6	Course outline.....	7

## 2 Installing Python

We will use the Anaconda distribution of Python. Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing that aims to simplify package management and deployment.

- Download Anaconda3 [here](#) (Windows 64-bit)

Further information is available on the [UP Chem Eng wiki](#). Note (if you see old code on the internet) that Python 3 came out in 2008 with some new syntax. Python 2 will have no support past 2020.

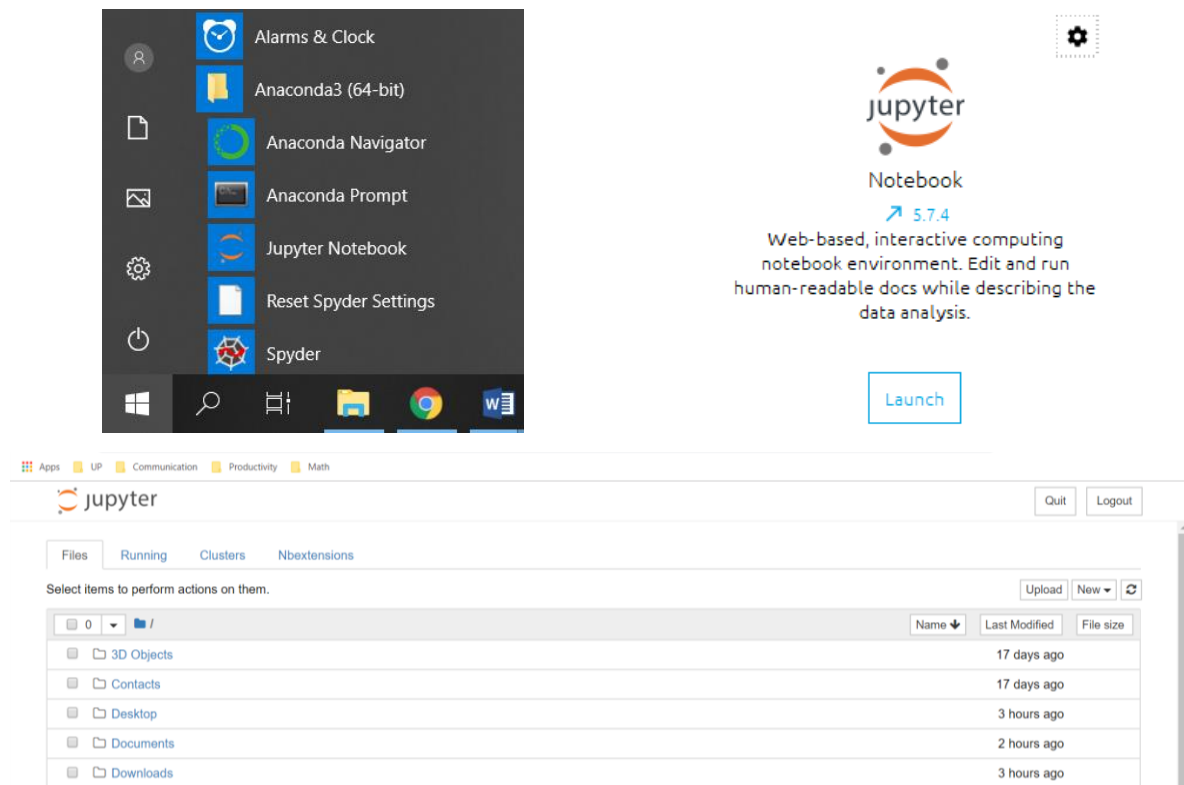
After installation, the [getting started](#) guide should open. You may skip the section on Spyder and read *Run Python in a Jupyter Notebook*, or just close the website and carry on with the instructions here.

## 3 Opening Jupyter

- In your start menu, under the Anaconda3 folder, open the Anaconda Navigator
- In the next menu that opens, click the Jupyter Notebook button (alternatively you can click the Jupyter Notebook button in your start menu as a shortcut)
- This will open the Jupyter Dashboard in your internet browser (note that an internet connection isn't required)

The Dashboard will have opened the files in C:\Users\Your Name. If you wish to change the start-up folder, read [this](#) for help. Note that it will only work if you start from the Jupyter Notebook shortcut, not Anaconda. You can either:

- Click on the Start menu > Anaconda3 > right-click Jupyter Notebook > More > Open file location. Right-click Jupyter Notebook shortcut > Properties > Paste the file location in *Start in* and over "%USERPROFILE%" in *Target*.
- Open the Anaconda Prompt (under Anaconda3), execute the following commands (copy and paste and press Enter):  
**cd /d C:\Users\Your Name\Downloads** (note this must be your chosen filepath)  
**jupyter notebook**



To create a new notebook (to check if everything is working), on the Jupyter Dashboard, on the right side of the menu is a “New” button. Click New and select Python 3. The new notebook displays its name at the top as *Untitled*. Click on the name and type a new name for your notebook. Back in the Dashboard menu, your notebook should appear with a green book showing that the kernel is active. Now select the box to the left of the green book and delete it by pressing the red trash can that appears at the top.

## 4 Version control

Code is often developed simultaneously, thus a version control system (VCM) is necessary. One example of a VCM is Git which works both online and offline. It shows who made changes at which times and allows you to revert back to old versions. GitHub is a website that hosts open source coding projects using Git.

- The Python Tutorial notes are available on GitHub [here](https://github.com/Franco-Pretorius/Python-Tutorial.git).
- Click the green Clone or Download button and then the clipboard next to it (this will copy the URL of the repository). You’ll use this in the next step.
- *E.g.* <https://github.com/Franco-Pretorius/Python-Tutorial.git>

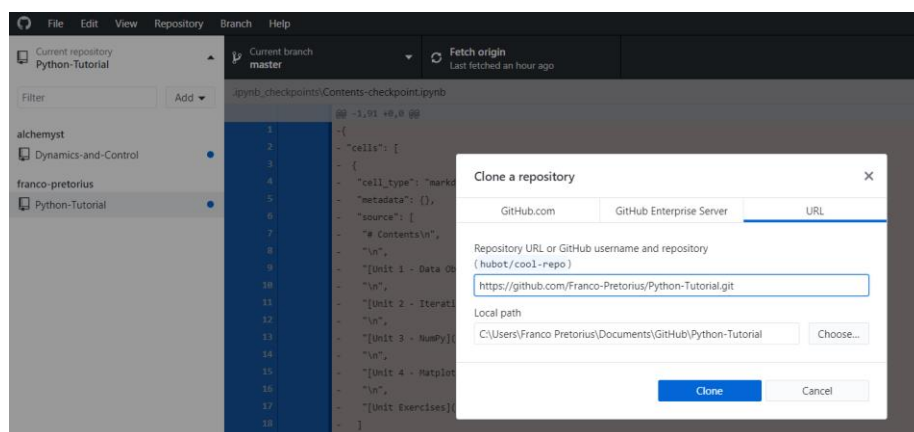
There are many ways of interacting with GitHub, such as Git Bash (which uses the command line interface) and the GitHub Desktop app (with a GUI which may be easier to use for nonprogrammers). Pick one of the methods described below. If you are planning to push code to GitHub (instead of just pulling code), you should create a [GitHub account](#).

## 4.1 GitHub Desktop

Download the application [here](#).

### 4.1.1 Cloning a repository

Once you have it opened, click: Current repository at the top left corner, then Add, Clone repository after which the following menu should open. Paste the URL from Section 4 under the URL tab. Change the local path to where you'd like the repository. Press **clone**. This will clone a local repository on your computer. You can now navigate to your notes in the Jupyter Dashboard and open them.



### 4.1.2 Updating a repository

If the owner of the repository updates the code, you can **pull** the new version by clicking the Repository tab, then Pull.

If you want to create your own repository, read [this](#).

## 4.2 Git Bash

If you are more familiar with the command line, use this method.

### 4.2.1 Cloning a repository

To clone the files (*i.e.* to create a local repository for the first time) do this:

1. Download Git Bash [here](#) (click the picture of a computer screen with a blue button inside it).
2. Complete the default installation.
3. Find the folder on your computer where you'd like to clone the Python-Tutorial notes and right-click to select Git Bash (a black terminal should open). Alternatively, you can open the Start menu and click on Git Bash, then change the working directory by following the advice [here](#).
4. Type **pwd** and press enter to print the working directory (to make sure you're in the right folder where you'd like to clone notes).
5. Copy the repository URL as mentioned in Section 4.
6. Go back to the black terminal and type **git clone** *followed by the URL* you copied, and press Enter.
7. After it has been cloned, you may close the terminal window.

If you get stuck, look at the example [here](#).

Note if you simply use the green Download ZIP button, you'll have to redownload the entire repository every time and won't be able to keep track of changes.

### 4.2.2 Updating a repository

To get the [latest version of a repository](#) from GitHub (after the author has made updates):

1. Open the local repository (Python-Tutorial file) you have cloned on your computer, right-click and select Git Bash here.
2. Type **git reset --hard** and press Enter. This restores the local repository to where it was when you had it originally by undoing all changes you have made (like running, adding or removing cells).
3. Type **git pull** and press Enter. This downloads the new files and the ones which have changed.

More information on making a repository is available [here](#).

## 5 Customisation

Extra packages can be installed from the Python Package Index (PyPI) using pip, which is included with Anaconda. pip is a recursive acronym for "Pip Installs Packages". Note that

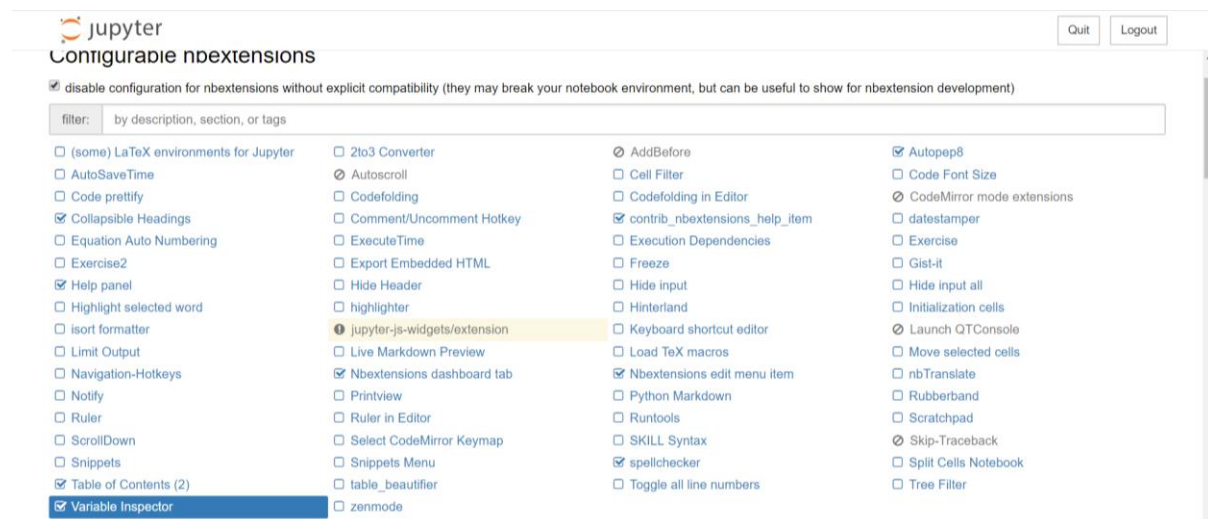
another package management system, **conda**, is also often used to install packages which may contain software written in any language.

*Currently, there are some technical difficulties with these options, and you may have to restart your computer for it to work.*

Using pip, you can add a menu of extensions to your Jupyter Dashboard.

- Open the Anaconda Prompt from the Start menu, and execute the following:  
**`pip install jupyter_contrib_nbextensions`**

The extensions include a spell-checker, variable inspector, table of contents, and even a button to format code according to PEP8 automatically.



You can also change the appearance of notebooks (e.g. use dark mode)

- Open the Anaconda Prompt from the Start menu, and execute the following:  
**`pip install jupyterthemes`**

Within a notebook, run the following in a cell to see the list of themes: **`!jt -l`**

They include:

- onedork
- chesterish
- grade3
- oceans16
- monokai
- solarizedl

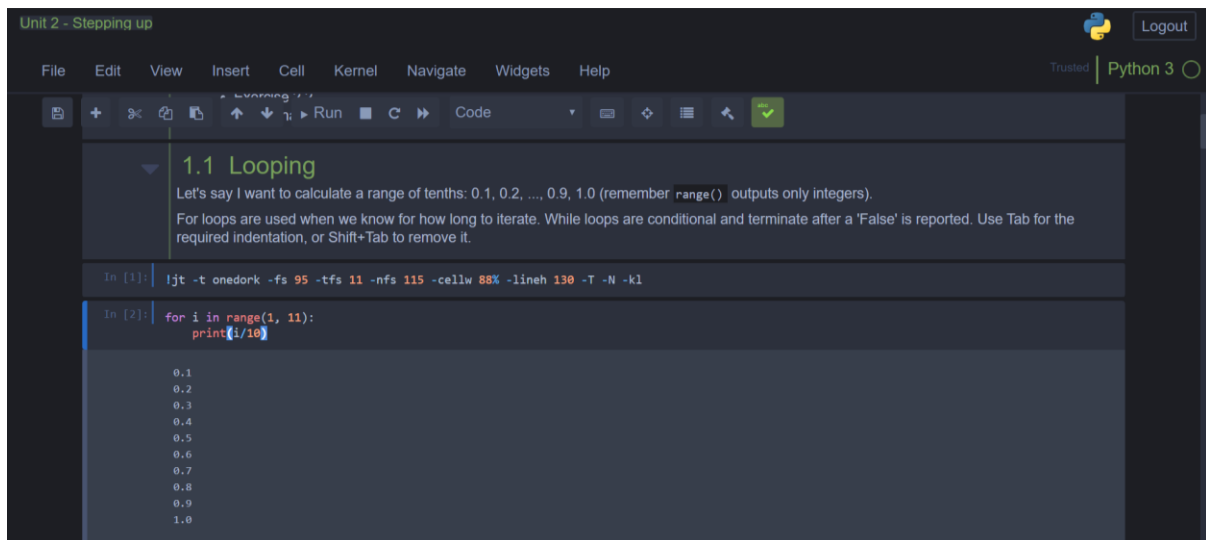
- solarizedd

To change the theme run the following (`!jt -t <name of the theme>`) and refresh the page.

The following is the creator's [preferred style](#):

`!jt -t onedork -fs 95 -tfs 11 -nfs 115 -cellw 88% -lineh 130 -T -N -kl`

To reset the theme, run: `!jt -r`



The screenshot shows a Jupyter Notebook interface with a dark theme. The title bar says "Unit 2 - Stepping up". The menu bar includes File, Edit, View, Insert, Cell, Kernel, Navigate, Widgets, and Help. The toolbar has icons for file operations, running, and other notebook functions. The code area shows a section titled "1.1 Looping" with explanatory text: "Let's say I want to calculate a range of tenths: 0.1, 0.2, ..., 0.9, 1.0 (remember `range()` outputs only integers). For loops are used when we know for how long to iterate. While loops are conditional and terminate after a 'False' is reported. Use Tab for the required indentation, or Shift+Tab to remove it." Below this, there are two input cells. The first cell contains the command `!jt -t onedork -fs 95 -tfs 11 -nfs 115 -cellw 88% -lineh 130 -T -N -kl`. The second cell contains a Python loop: `for i in range(1, 11):` followed by an indented `print(1./10)`. The output of the second cell is a list of values from 0.1 to 1.0 in increments of 0.1.

## 6 Course outline

The course notes introduce a new module per unit. It is recommended to do the Unit exercises before continuing with the next unit.

Extra help and additional examples:

- [Python Tutor](#) (to visualise code, especially looping)
- [NumPy for MATLAB users](#)
- [Markdown syntax](#)
- [Interesting Jupyter Notebooks](#)
- [Python for Chemical Engineers](#) by CACHEM
- [Chemical Engineering Analysis and Control](#) by JC Kantor
- [Dynamics and Control](#) by C Sandrock