First Steps with Python

Introductory Lab to the Bootcamp

Ana Matran-Fernandez & Viola Fanfani

26 November 2018

Table of contents

1. Getting started

2. Git

3. How do I run a Python program?

4. Python basics practice

Getting started

Collaborative tools

> No man is an island
>
> John Donne

Research is a collaborative activity: you are going to write your paper with your supervisor/s, to collaborate with other colleagues, to release code online...

The best way to get started is to know where to look for help and use some useful tools from the beginning.

We will use some collaborative platforms during this bootcamp, but there are many others that you might use in the future.

Set up your accounts

- GitHub “A code hosting platform for version control and collaboration.”
- Slack `https://pgrbootcamp.slack.com/`
- Trello Collaborative tool for project management `https://trello.com/invite/bootcamp90/d0c3fbf3c6d096e3fb0522eff84ced75`

Other collaborative tools

Do you know/use other resources?

Go to the Slack channel and share them with the other attendees!!

Git

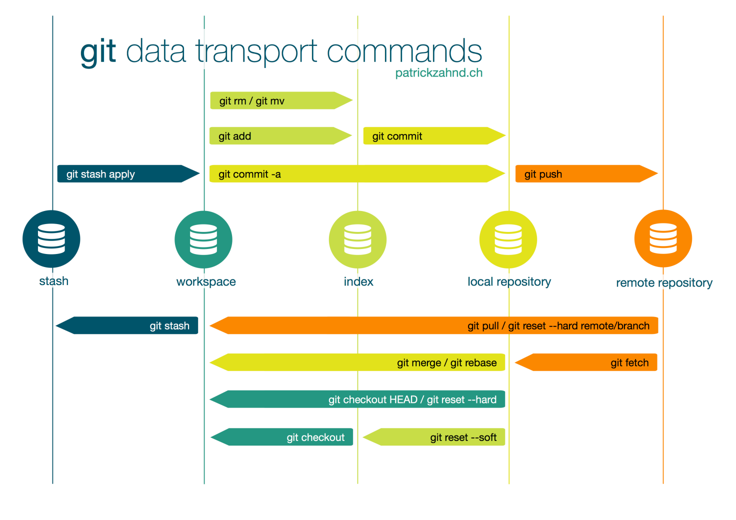
Getting started with GitHub

What is Git?

“Git is an example of a Distributed Version Control System (DVCS) commonly used for open source and commercial software development. DVCS’s allow full access to every file, branch, and iteration of a project, and allow every user access to a full and self-contained history of all changes. Unlike once popular centralized version control systems, DVCS’s like Git don’t need a constant connection to a central repository. Developers can work anywhere and collaborate asynchronously from any time zone.”

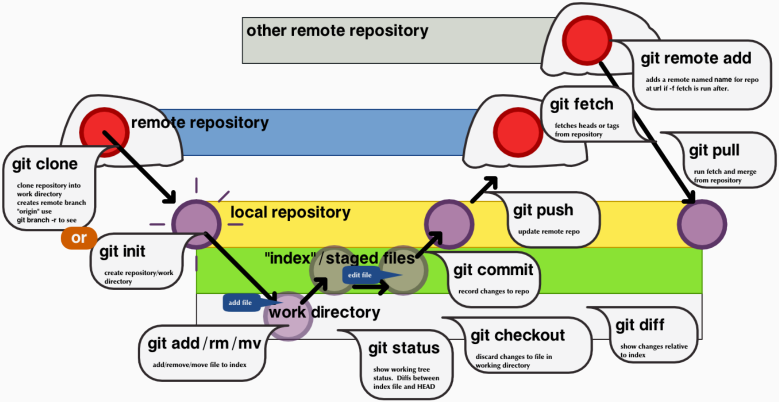
For a full overview on git, refer to the Git Handbook.

Getting started with Git



`https://www.patrickzahnd.ch/uploads/git-transport-v1.png`

Getting started with GitHub



other remote repository

remote repository

local repository

"index" / staged files

work directory

add file

edit file

git clone

clone repository into work directory creates remote branch "origin" use git branch -r to see

or

git init

create repository/work directory

git add / rm / mv

add/remove/move file to index

git status

show working tree status. Diffs between index file and HEAD

git checkout

discard changes to file in working directory

git diff

show changes relative to index

git commit

record changes to repo

git push

update remote repo

git fetch

fetches heads or tags from repository

git pull

run fetch and merge from repository

git remote add

adds a remote named name for repo at url if -f fetch is run after.

Getting started with GitHub

Now, let’s set up our own repository and copy the material for this lab in it.

□ clone the bootcamp repository with

```
git clone <repo_link>
https://github.com/albanda/bootcamp2018
```

□ create your own (empty) repository on GitHub

□ clone your new repo on your computer (use M: folder)

Getting started with GitHub

☐ copy the `intro_to_python` folder from the bootcamp repo inside your personal repo

☐ now add, commit and push your changes

```
git add .
git commit -m "message"
git push
```

☐ go to the online repository and check what happened

How do I run a Python program?

Python Shell and Scripts

The basic way of running some Python code is using the Python command line.

Now open IDLE and run:

```
print("Hello, welcome to the 2018 Bootcamp")
```

It works and it’s easy.

Python Shell and Scripts

Now try to run this code on the console:

```python
def longer_function(float_list, ptr, number):
    try:
        starting_value = float_list[ptr]
    except IndexError:
        print("ptr is too big, the list only has
                %d elements" % (len(float_list)))
        ptr = 0
    larger = []
    for i in range(ptr, len(float_list)):
        if number >= float_list[i]:
            larger.append(i)
    return larger

l = longer_function([1, 10, 15, 30, 40, 60, 22, 4, 56], 4, 7)
print(l)
```

Python Shell and Scripts

Having to write multiple rows (or full functions!) on the command line is not straightforward.

For this purpose we can use scripts — text files with the extension ".py" — that can be run from the command line or imported in a longer script.

Python IDEs and debugging

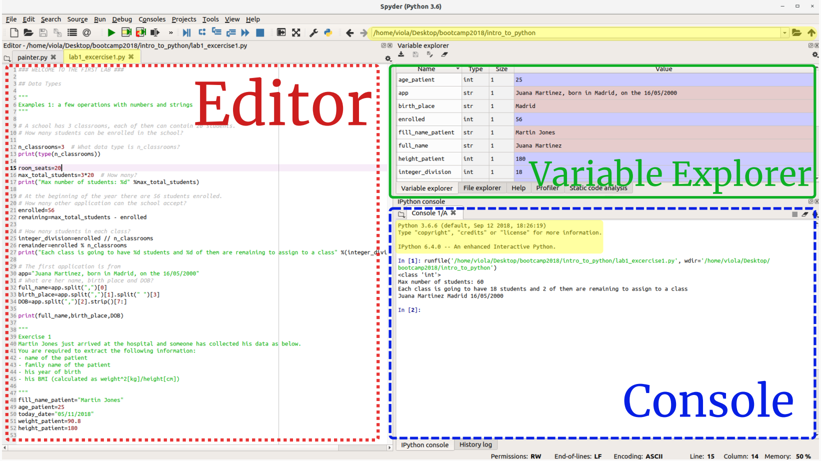
When the length and complexity of the program increase, we may need some more advanced tools that allow us to debug and keep trace of the work we are doing.

PyCharm and Spyder are two examples of such IDEs.

We have chosen Spyder as IDE for this course because :

- It provides a ready-to-go environment for scientific development
- It's free, multi-platform and built for Python
- It has a "MATLAB-like" interface: more data analysis oriented rather than hardcore debugging oriented

Spyder



```python
### WELCOME TO THE FIRST LAB ###

## Data Types

"""
Examples 1: a few operations with numbers and strings
"""

# A school has 3 classrooms, each of them can contain 20 students.
# How many students can be enrolled in the school?

n_classrooms=3  # What data type is n_classrooms?
print(type(n_classrooms))

room_seats=20
max_total_students=3*20  # How many?
print("Max number of students: %d" %max_total_students)

# At the beginning of the year there are 56 students enrolled.
# How many other application can the school accept?
enrolled=56
remaining=max_total_students - enrolled

# How many students in each class?
integer_division=enrolled // n_classrooms
remainder=enrolled % n_classrooms
print("Each class is going to have %d students and %d of them are remaining to assign to a class" %(integer_divi

# The first application is from
app="Juana Martinez, born in Madrid, on the 16/05/2000"
# What are her name, birth place and DOB?
full_name=app.split(",")[0]
birth_place=app.split(",")[1].split(" ")[3]
DOB=app.split(",")[2].strip()[7:]

print(full_name,birth_place,DOB)

"""
Exercise 1
Martin Jones just arrived at the hospital and someone has collected his data as below.
You are required to extract the following information:
- name of the patient
- family name of the patient
- his year of birth
- his BMI (calculated as weight^2[kg]/height[cm])

"""
fill_name_patient="Martin Jones"
age_patient=25
today_date="05/11/2018"
weight_patient=90.8
height_patient=180

```

Name	Type	Size	Value
age_patient	int	1	25
app	str	1	Juana Martinez, born in Madrid, on the 16/05/2000
birth_place	str	1	Madrid
enrolled	int	1	56
fill_name_patient	str	1	Martin Jones
full_name	str	1	Juana Martinez
height_patient	int	1	180
integer_division	int	1	18

```
Python 3.6.6 (default, Sep 12 2018, 18:26:19)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

In [1]: runfile('/home/viola/Desktop/bootcamp2018/intro_to_python/lab1_excercise1.py', wdir='/home/viola/Desktop/
bootcamp2018/intro_to_python')
<class 'int'>
Max number of students: 60
Each class is going to have 18 students and 2 of them are remaining to assign to a class
Juana Martinez Madrid 16/05/2000

In [2]:
```

Permissions: RW   End-of-lines: LF   Encoding: ASCII   Line: 15   Column: 14   Memory: 50 %

Before getting started

For today we have prepared a few exercises on Python basics. But before you dig into them, here is some advice.

The following examples are intended to give you a **flavour** of how data types, data structures and functions are managed in Python.

Showing everything is impossible, not only in a crash course, even in a one-year-long course. Hence, don't be shy! Ask us, each other and Google: most likely there have been others like you looking for help.

Examples:
`https://stackoverflow.com/questions/509211/understanding-pythons-slice-notation`
`https://stackoverflow.com/questions/1024847/add-new-keys-to-a-dictionary`

Python basics practice

Content of the lab

During this first lab you are going to practice some Python basics.

The material is organised in scripts. Each of them contains some examples of Python code to show you how to use the features you have learnt during the lecture. After that, there are a few exercises that you are asked to complete.

The scripts are organised as follows:

- `lab1_exercise1.py`: basic data formats, small operations with variables
- `lab1_exercise2.py`: lists, tuples and dictionaries
- `lab1_exercise3.py`: loops and conditional statements
- `lab1_exercise4.py`: functions

Exercise 1

- □ Have you copied the `intro_to_python` folder into your repository?
- □ Open the `lab1_exercise1.py` script with Spyder (or your favourite IDE)
- □ Make some changes to the script, for example start commenting the code. Remember: keeping the code updated and commented is really good practice!
- □ Commit and push your changes, you'll see your repository updated.
- □ Now go back to the exercise and try to solve all the questions.
- □ **Commit and push every time it's needed**