Lesson 1 - First Steps

$ whoami

> Lorenzo Lancia

- ➔ BSc in Physics
- ➔ MSc in Data Science

Student Representative in Course Council
Contact me if you have trouble! lorentz90@gmail.com

Didn't knew python before MSc

Amazon Italy intern Fall 2016

Interviewer: Is studying computer science the best way to prepare to be a programmer?

Bill Gates: No. the best way to prepare is to write programs, and to study great programs that other people have written.

Do you agree?

Outline

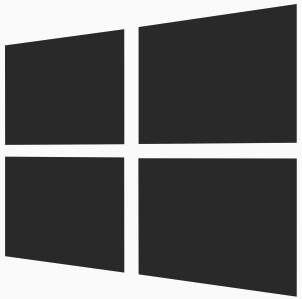
- Interacting with the system in a powerful way -> the **command line interface**
  - Different systems, different cli
  - Bash and Gnu Core Utilities
- Installing your **python interpreter** and **Anaconda** package manager
- Choosing your editor
  - Text Editors
  - IDEs
  - The **Jupyter Notebook**
- Version Control (**Git**)
- Looking for help offline and online
- Slack - collaboration with your friends and colleagues

Q&A
Session

The command line interface

https://xkcd.com/1084/

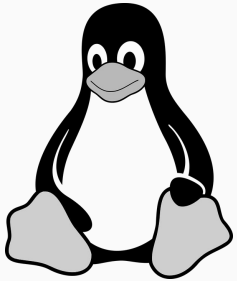
Open it UP!



Windows+X -> Powershell or cmd

OS X

Terminal App



Maybe you should go and grab a coffee

Setting up bash

Windows

Install Linux Subsystem

https://goo.gl/ZajLVs

OSX

Everything is ok, but let's install a package manager

https://brew.sh/

Meanwhile ...

Why Data Science @ Command Line?

The command line is:

- Agile
- Augmenting
- Scalable
- Extensible
- Ubiquitous

Bash bourne again shell

It’s the common shell for unix like system.

A maybe useful example

http://www.dis.uniroma1.it/~rosati/dmds/lectures.htm

Let's download all the pdf

```
$ lynx --dump http://www.dis.uniroma1.it/~rosati/dmds/lectures.htm | grep pdf
| awk '{print $2}' | xargs wget
```

Warning you need to install lynx

```
$ sudo apt-get install lynx
$ brew install lynx
```

Basic commands

`$ whoami`	Print current username
`$ pwd`	Print current path
`$ ls`	List file in current directory
`$ cd`	Change directory
`$ mkdir`	Create directory
`$ head`	Print first lines of input
`$ tail`	Print last lines of input

Basic commands

$ man <nameofcommand>
$ help <command> HELP
$ <command> -h
$ <command> --help

The Environment

Top Down approach

- Command-line tools
- Terminal
- Shell
- Operating system

Five Types of Command-Line Tools

- A binary executable
- A shell builtin
- An interpreted script
- A shell function
- An alias

Redirecting IO

Writing the output of a command to a file

```
Seq 10 > ten-number.txt
```

Append to a file

```
echo " World" >> hello-world.txt
```

Passing output of a command to input of another program

```
cat hello-world.txt | wc -w
```

```
< hello-world.txt wc -w
```

```
wc -w hello-world.txt
```

Working with files

```
$ mkdir data
```

Moving files

```
$ mv hello-world.txt data
```

can also rename files with mv :

```
$ cd data
$ mv hello-world old-file
```

Removing files

```
$ rm old-file
```

Remove directory

```
$ rm -r ~/book/ch02/data/old
```

Copy files

```
$ cp server.log server.log.bak
```

Integrate python scripts

I've created a python script to check top words in a document. And saved in topwords.py

```
#!/usr/bin/env python
import re
import sys
from collections import Counter
num_words = int(sys.argv[1])
text = sys.stdin.read().lower()
words = re.split('\W+', text)
cnt = Counter(words)
for word, count in cnt.most_common(num_words):
    print "%7d %s" % (count, word)
```

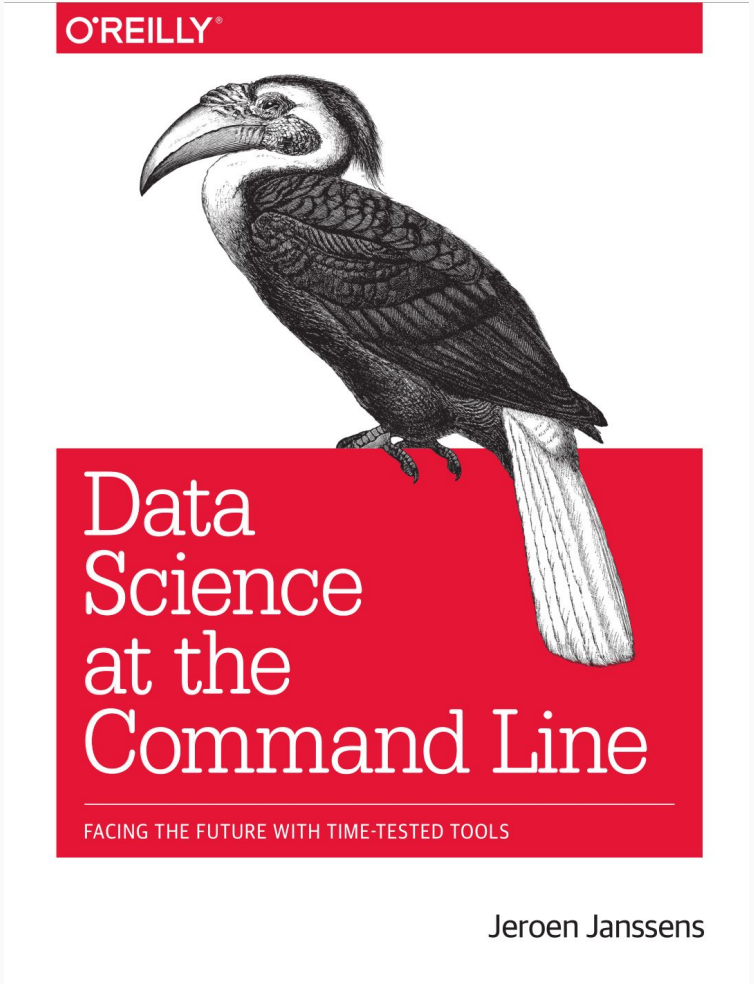
```
$ curl www.gutenberg.org/files/1342/1342-0.txt | ./topwords.py 5
```

Resources and further topics

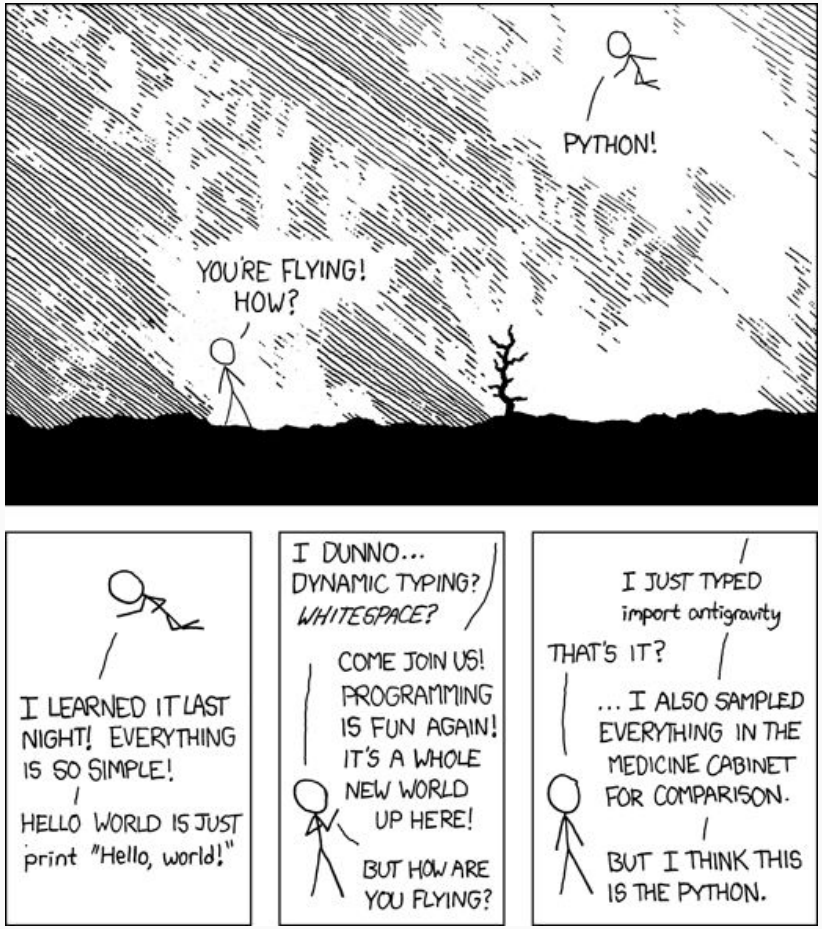
- Useful commands for data:
  - grep, curl, cut ...
- Bash scripting
- Other shells (i love zsh)
- Complex commands
  - awk, sed

The Linux community is full of great resources.

Q&A Session



Setting up your python system



https://xkcd.com/353/

Anaconda

Leading Open Data Science Platform
Powered by Python

https://www.anaconda.com/

PYTHON INSTALL PARTY

- User level install of the version of python you want
- Able to install/update packages completely independent of system libraries or admin privileges
- conda tool installs binary packages, rather than requiring compile resources like pip -
- More or less eliminates the headaches of trying to figure out which version/release of package X is compatible with which version/release of package Y, both of which are required for the install of package Z
- Comes either in full-meal-deal version, with numpy, scipy, PyQt, spyder IDE, etc. or in minimal / alacarte version (miniconda) where you can install what you want, when you need it
- No risk of messing up required system libraries

Using conda

One of the power of python stand in its libraries. The **conda** package manager will help you installing the one you need.

```
$ conda install seaborn
```

You can manage virtual env with conda to have different version of python and/or libraries and use the env you need

Installing a package with pip

For packages that are not available from conda

The python interpreter

```
$ python
>>
```

Or better:

```
$ ipython
In[1]:
```

Read–Eval–Print Loop (REPL)

```
help(object)
```

Warning spaces are important in **python**

indentation is meaningful

No { --- } (curly brackets)

And No ; at the end of line

Working with .py files

REPL is good. But you want re-usable scripts

```
$ nano my-first-python-script.py
```

Write:

```
print(“hello world”)
```

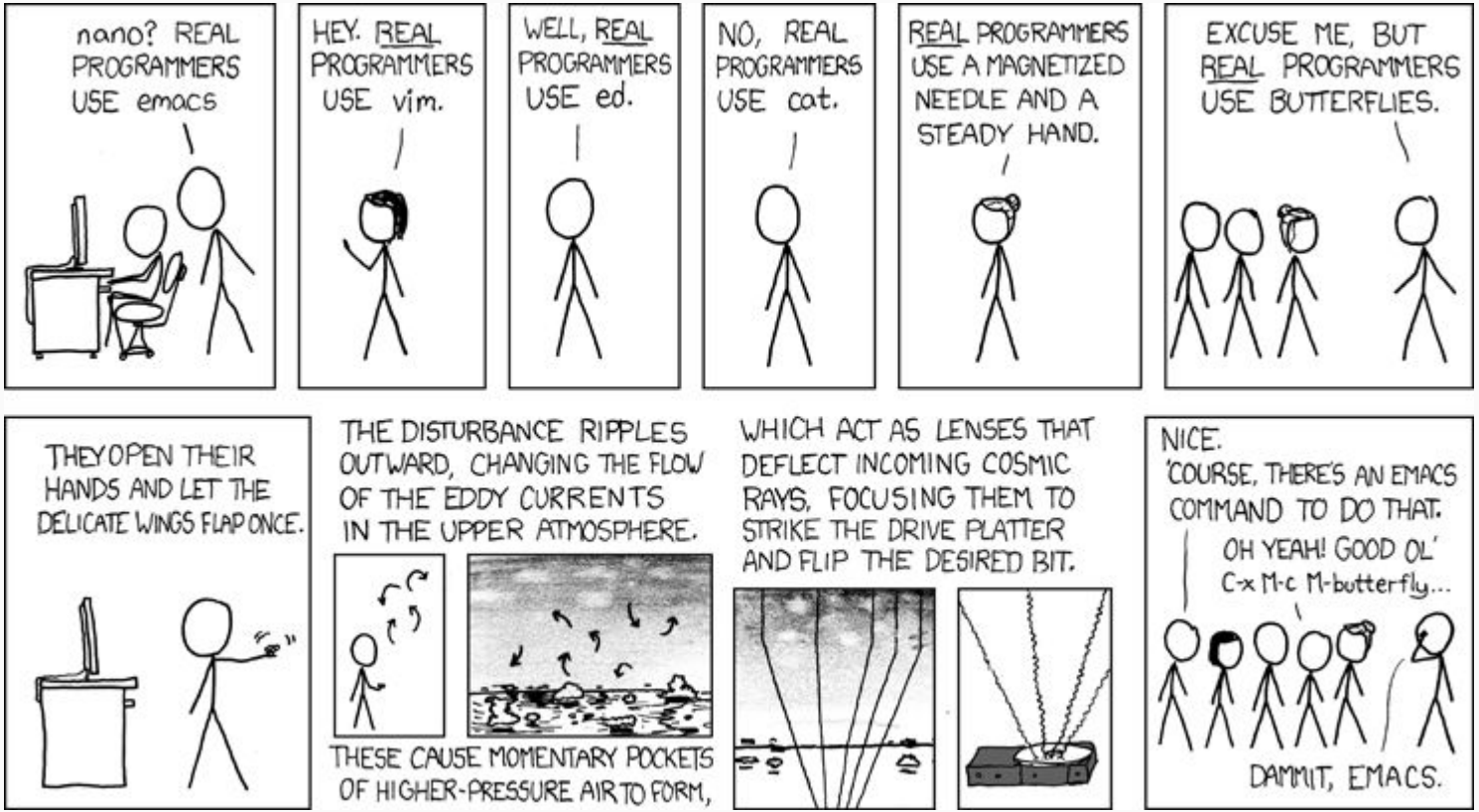
Write out file ctrl+o, exit editor ctrl+x

```
$ python  my-first-python-script.py
```

Your development environment

- Minimal
  - Just your terminal
- Simple
  - A Text Editor
- Advanced
  - An IDE
- The outsider
  - Jupyter Notebook

Rodeo



https://xkcd.com/378/

Jupyter Notebook

A **must have** tool in the belt of every Data Scientist.

```
$ jupyter notebook
```

A webserver will start listening on port 8888 and a webpage will automatically appear. Create a new Notebook and see the %Magic%.

The Notebook is REPL, but can export to a .py script.

The Nb can display tables, images, graphs inline.

You can write formulas, text and export the result as a complete report!

Further topics

Virtual Environments with conda

Choose your own editor or IDE

Play Around with Jupyter Notebook

Conda & Jupyter Notebook can be used also with one other important languages you'll use in this master: R

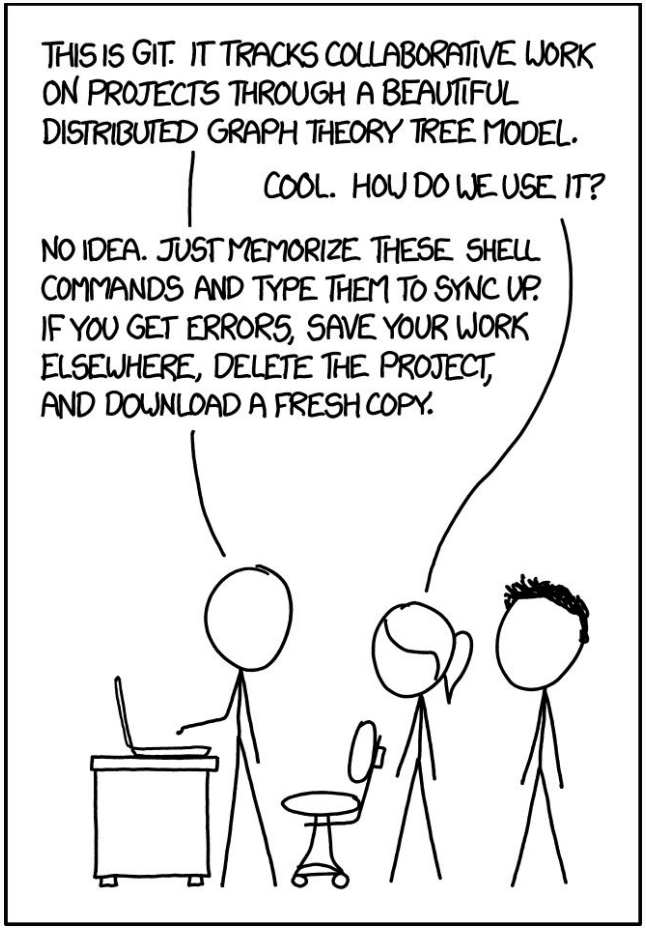
Q&A Session

In the next days …

- Python Data Structures
  - List
  - Tuples
  - Dictionaries
- Control Flow Statements
  - Loops
  - If
- Functions
- List comprehension & generators
- Python OOP
  - Classes & Methods

Git - The version control system

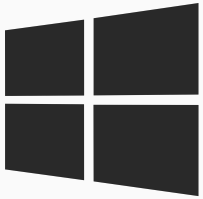
Git is beautiful



Git can be a pain

https://xkcd.com/1597/

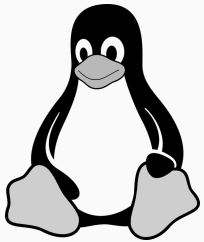
Installing Git



https://git-scm.com/download/win

OS X

Developers Tool
Or
$ brew install git



$ sudo apt-get install git

Git is not Github

https://guides.github.com/activities/hello-world/

Git is a distributed version control system

Github is a online service that offer git repository storage and related services

Create a github account if you don't have one already. **We are going to use git & github** for **assignment** and **projects** in the next few days.

Starting up with git

`$ git help` can be a starting point!

Some basic config

$ git config --global user.name "Lorenzo Lancia"

$ git config --global user.email lorentz90@gmail.com

$ git config --global color.ui true

`$ git config --global core.editor "nano"`

Starting up with git

Create a Repository

$ mkdir testrepo

$ cd testrepo

$ git init

Git Workflow

make some changes (create, edit or delete files)

add your changes to the staging area

Commit your changes

```
$ nano README.md                          write something in it

$ git status

$ git diff

$ git add README.md

$ git commit -m “add something to readme file”
```

Going back in time

Discard changes to a file

```
$ git checkout -- README.md
```

Undoing a commit

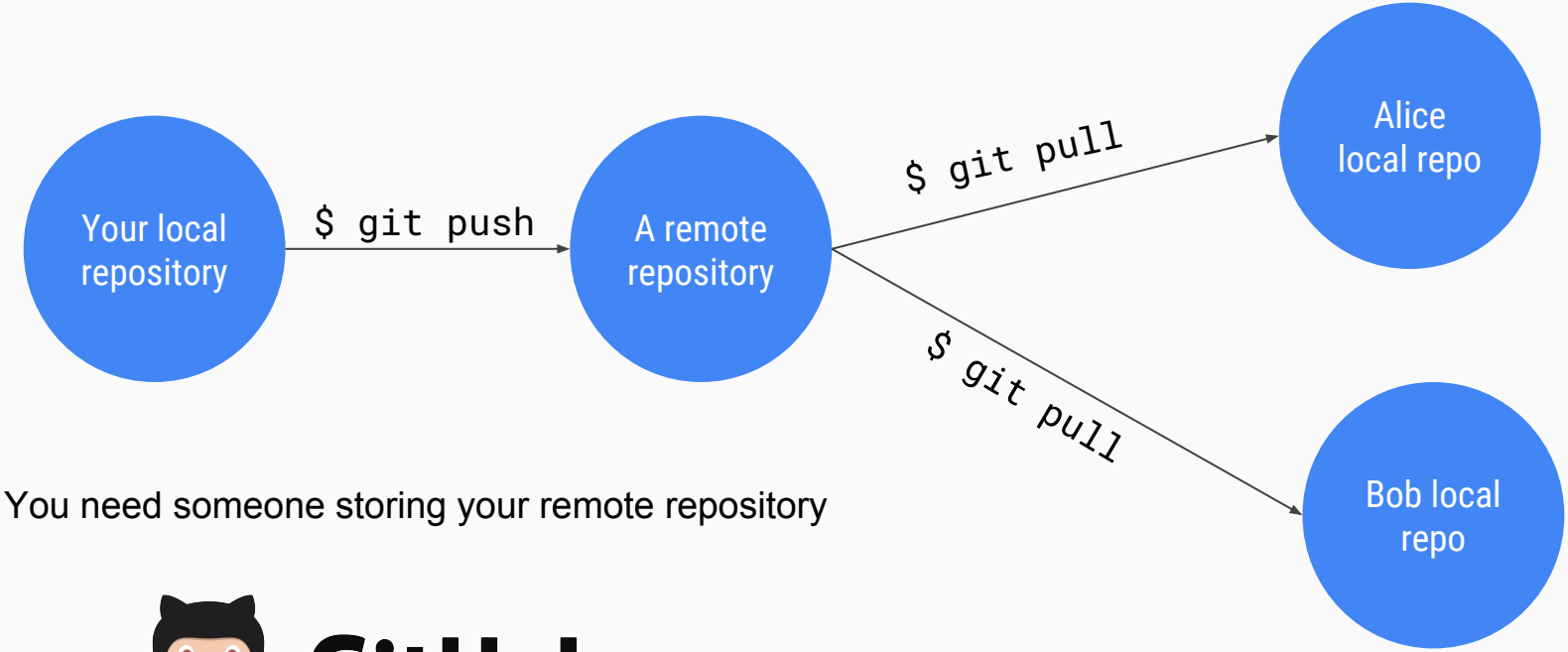
```
$ git reset --soft HEAD^
```

undo but put changes into staging

```
$ git reset --hard HEAD^
```

Undo last commit and all changes

Remote Repositories



You need someone storing your remote repository

Managing remote repo

Create an empty repo on github

Add to local repo a remote source

```
$ git remote add origin https://github.com/llancia/py101-gitintro.git
```

Push local commits to remote

```
$ git push
```

Pull remote commits to local

```
$ git pull
```

Collaborate!

Different people (if authorized) can **push** to a remote repository and collaborate to the same project.

Start by cloning someone else repo

```
$ git clone <remote repo address> <local path>
```

Automatically adds the origin remote

```
$ git remote -v
```

Branching & Merging

```
$ git branch <branchname>
```

```
$ git checkout <branchname>
```

You can work on your branch

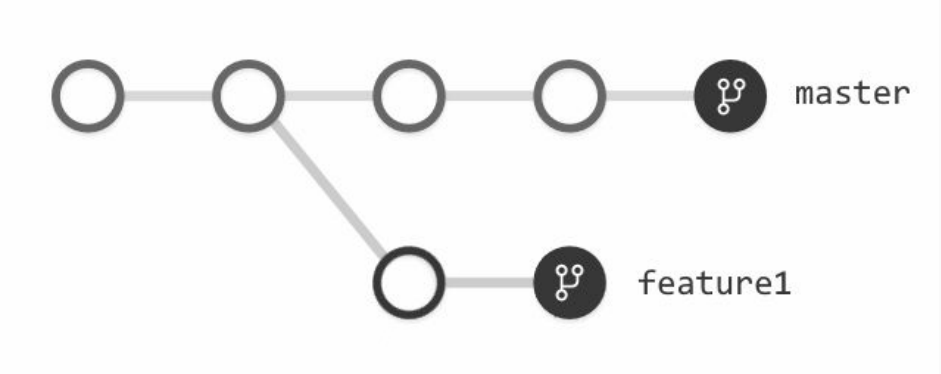
Then switch back to master

```
$ git checkout master
```

And merge back

```
$ git merge <branchname>
```

If needed git will create a merge commit



Troubleshooting ---> Don’t Panic!

Zero Theorem of Troubleshooting:

Your problem is always someone else's problem

Corollary 1

Google is your friend!

Corollary 2

If Google can’t help you. You are misexpressing your problem

Resources and further topics

Git is not the only version control but open source projects and many private industries uses it.

Github can be your code portfolio to show to recruiters.

This was just an brief introduction to git.

Online interactive course

https://www.codecademy.com/learn/learn-git

Learn by doing.

Q&A Session

Your very first assignment
Look on slack!

Thanks!

101

python crash course