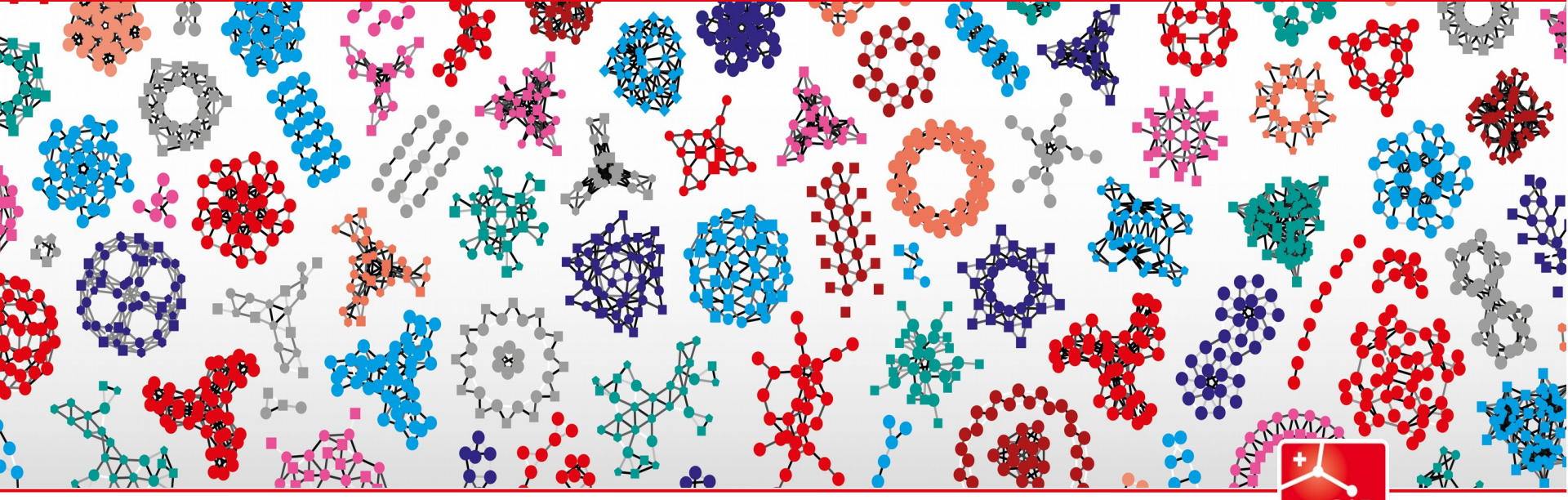


SIB Swiss Institute of Bioinformatics

First steps with Python in life sciences

Wandrille Duchemin, Robin Engler, and Bulak Arpat

General information

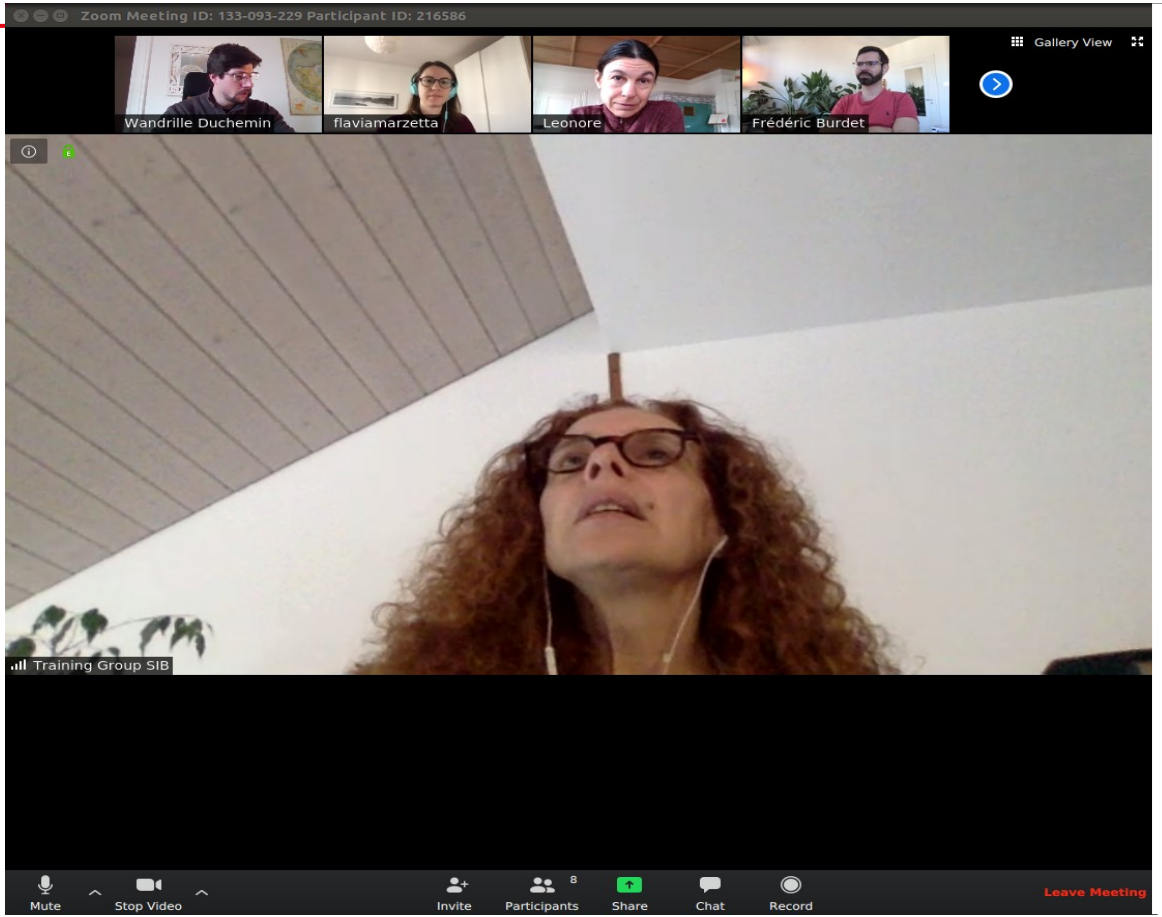
Course page : https://edu.sib.swiss/course/view.php?id=455

Student account : pls20

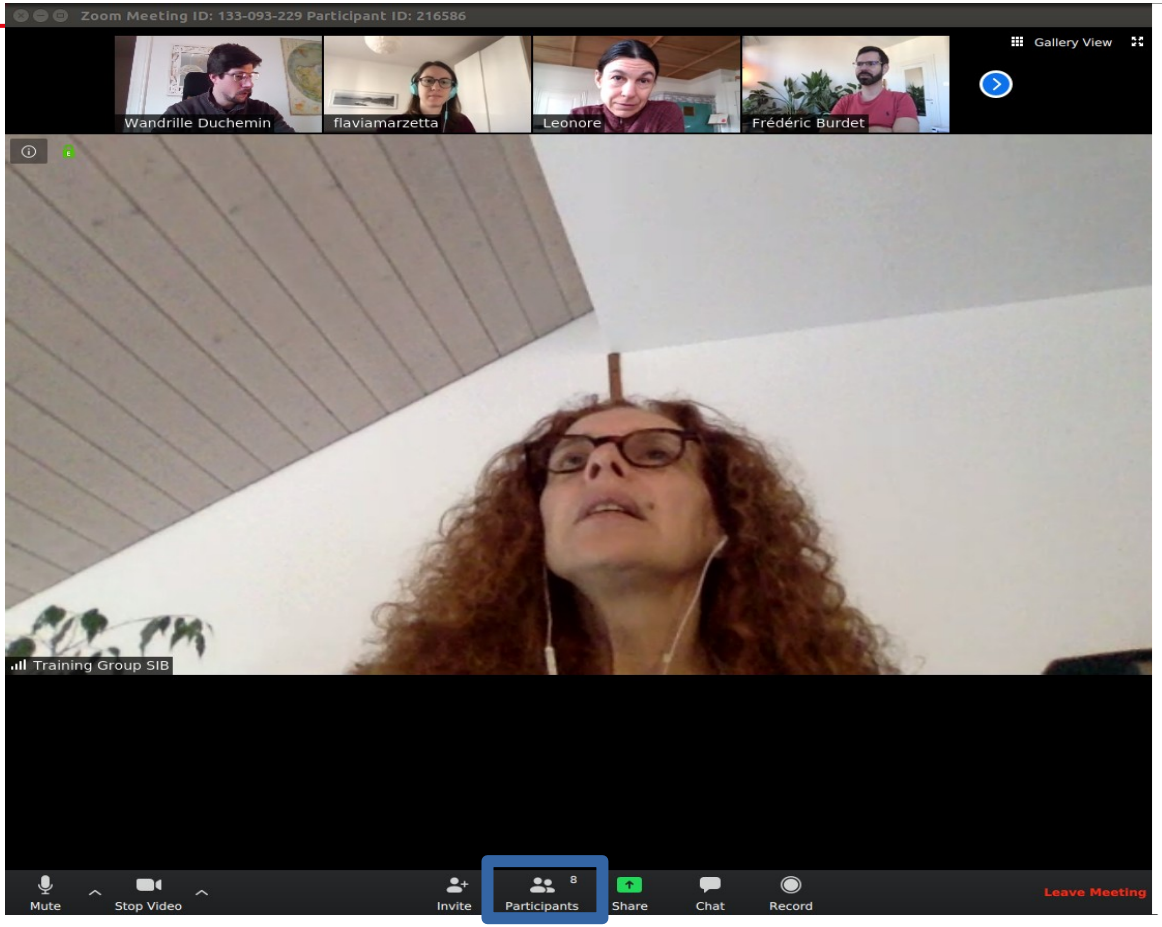
password: SIB-pls20

Etherpad : https://public.etherpad-mozilla.org/p/python-ge2018

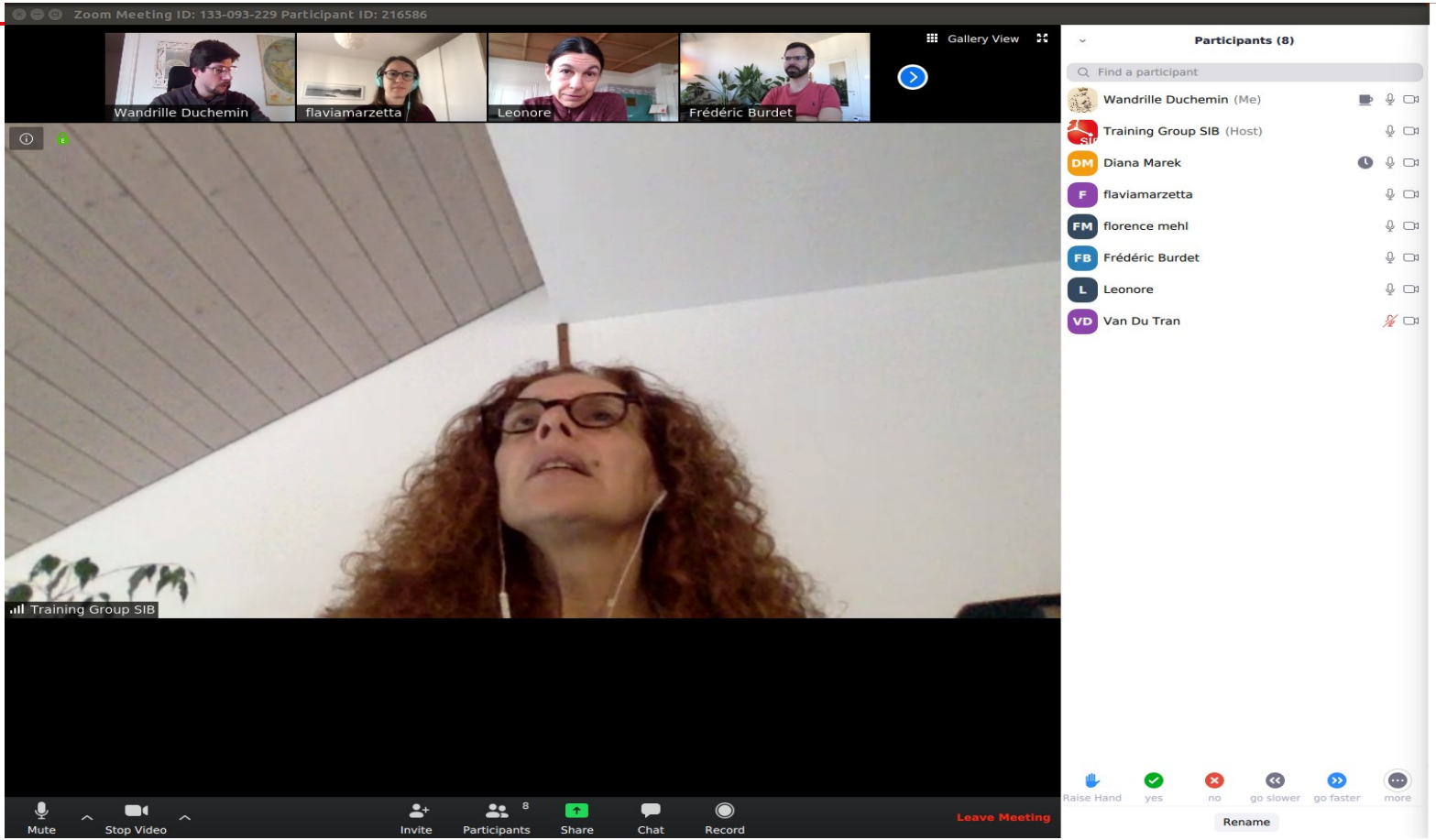
Online meeting client



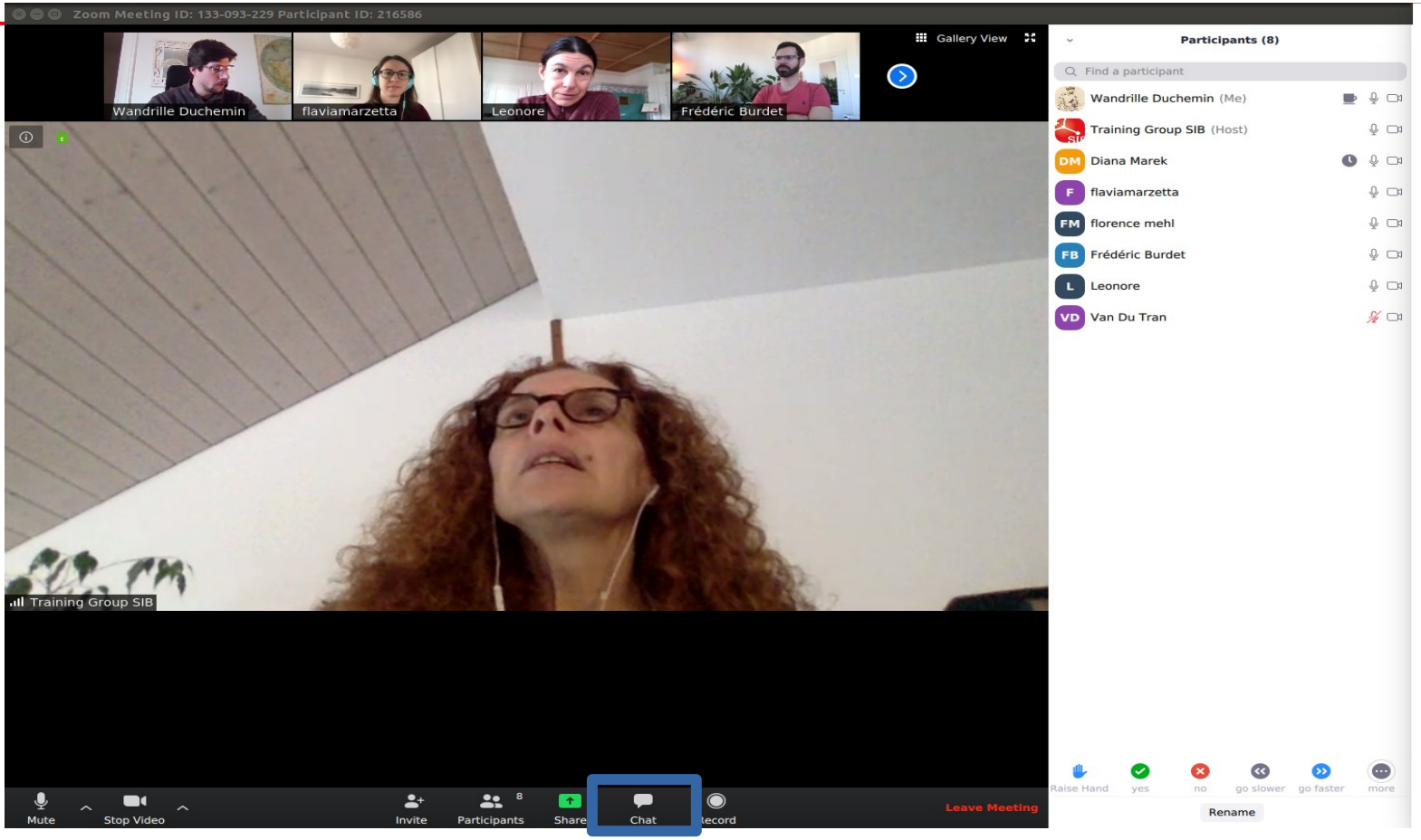
Online meeting client



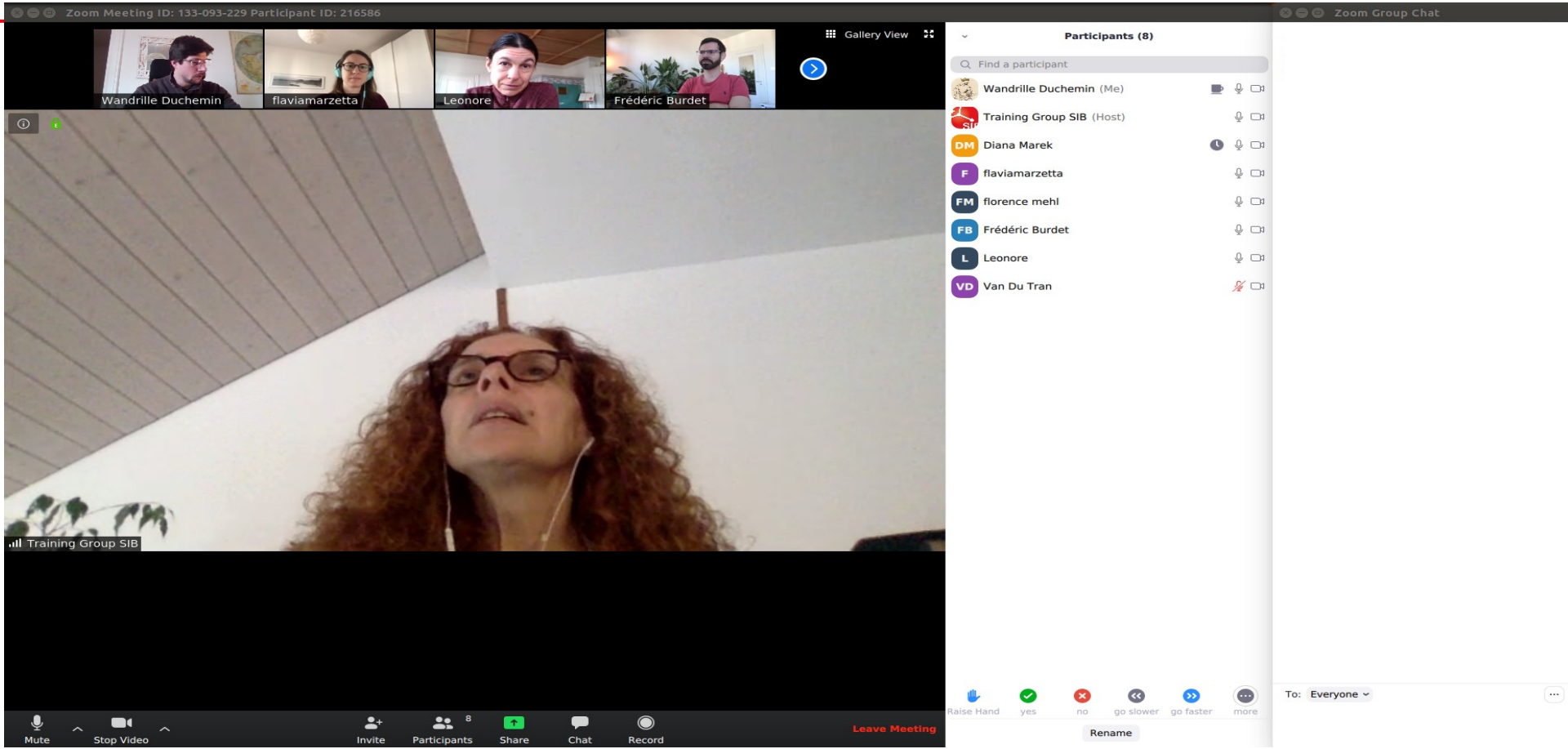
Online meeting client



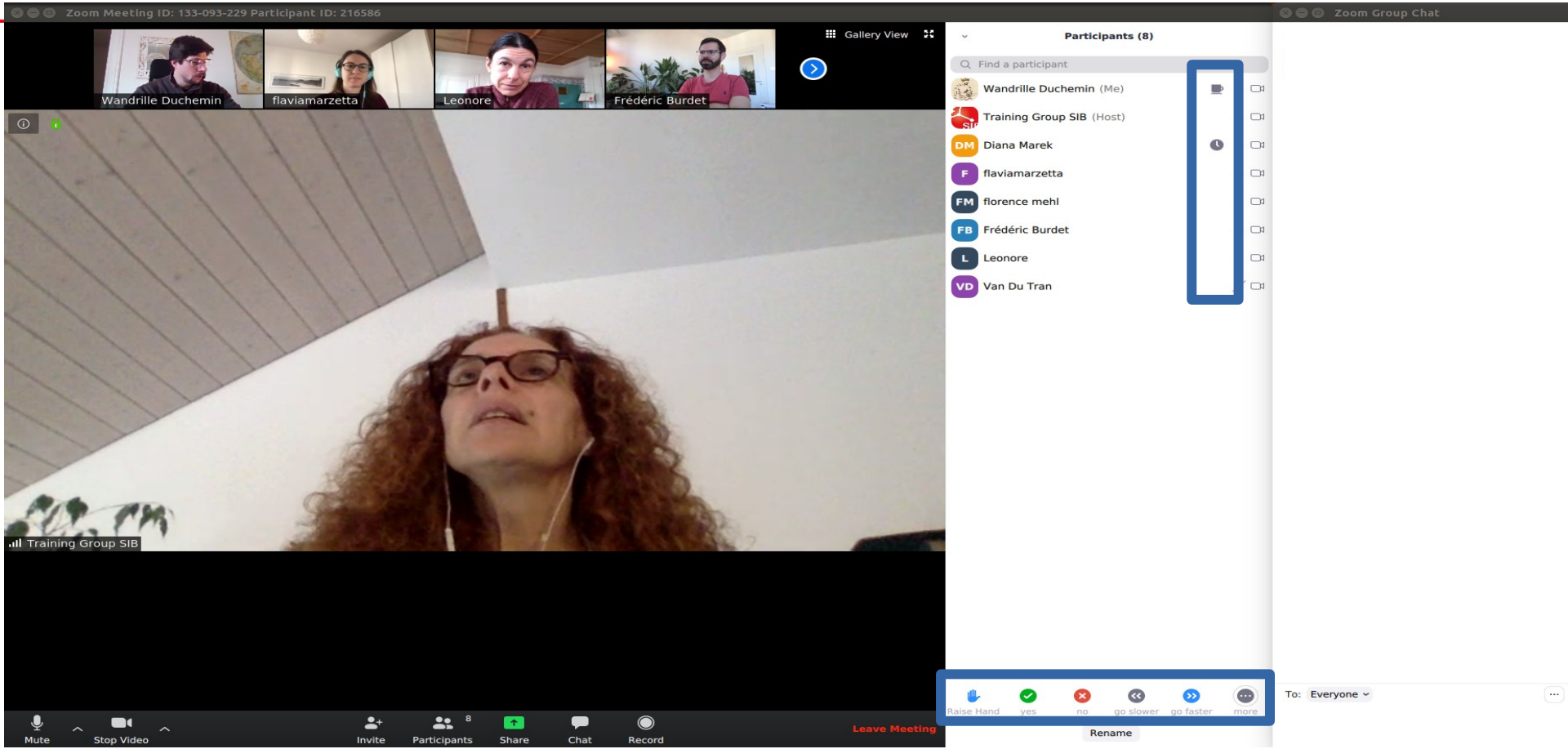
Online meeting client



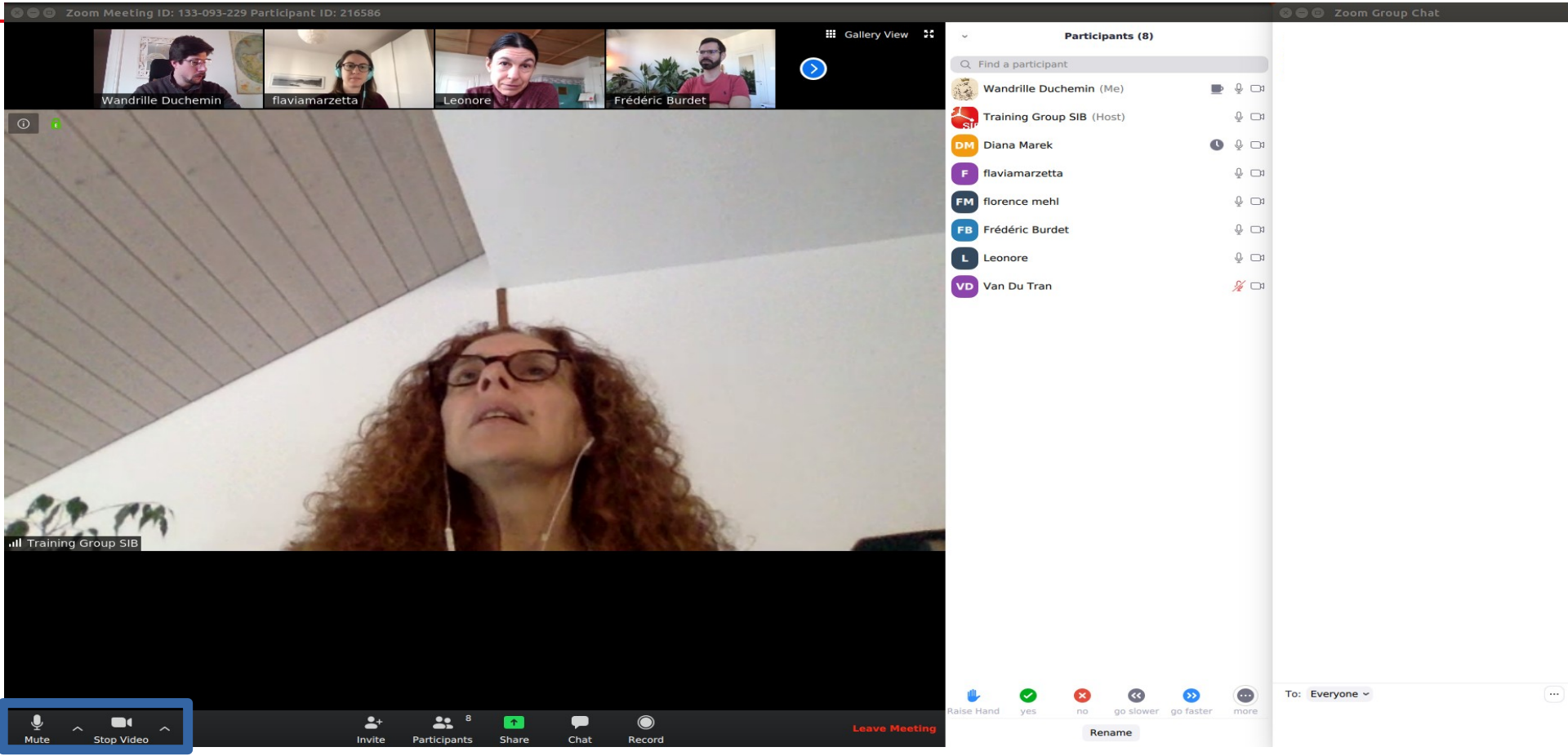
Online meeting client



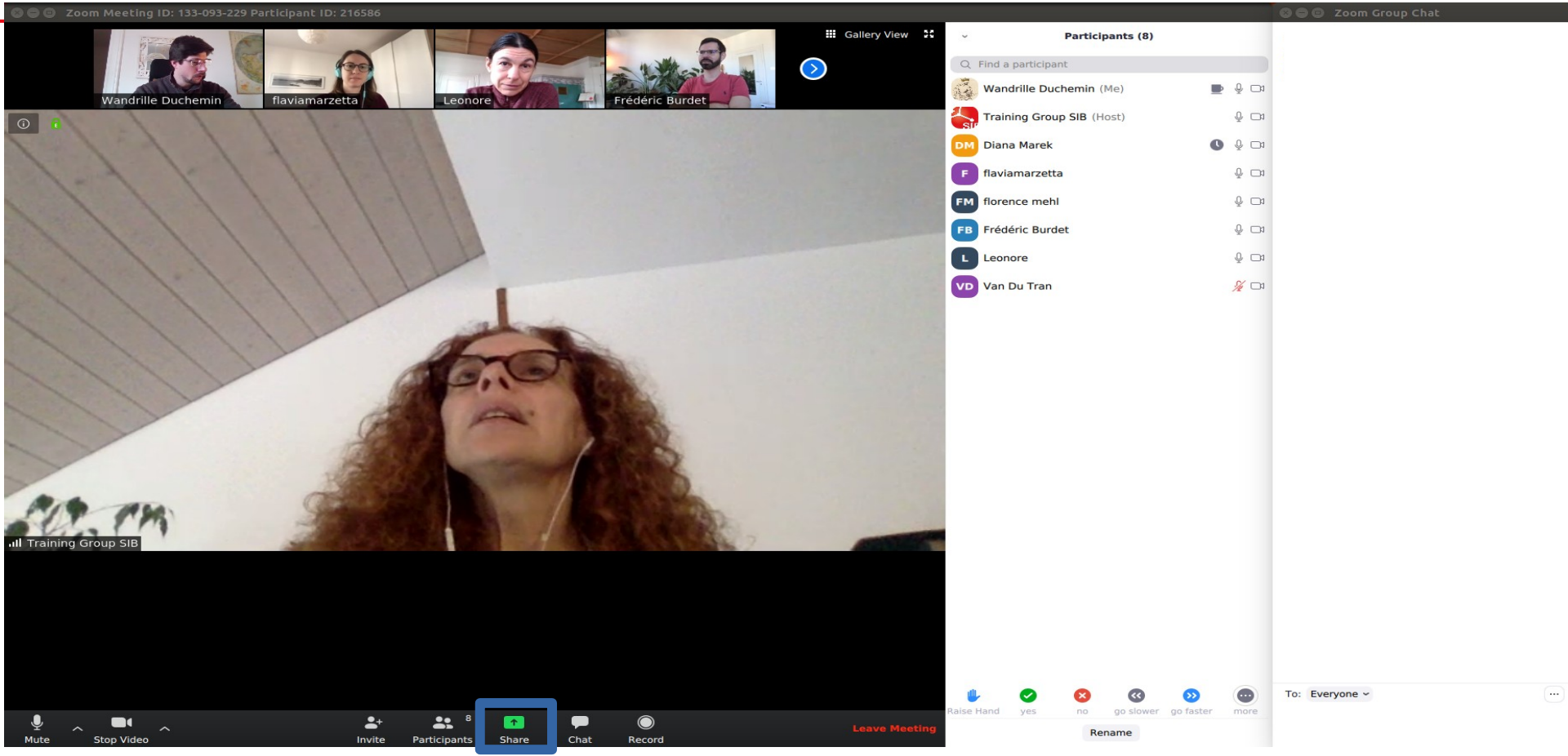
Online meeting client



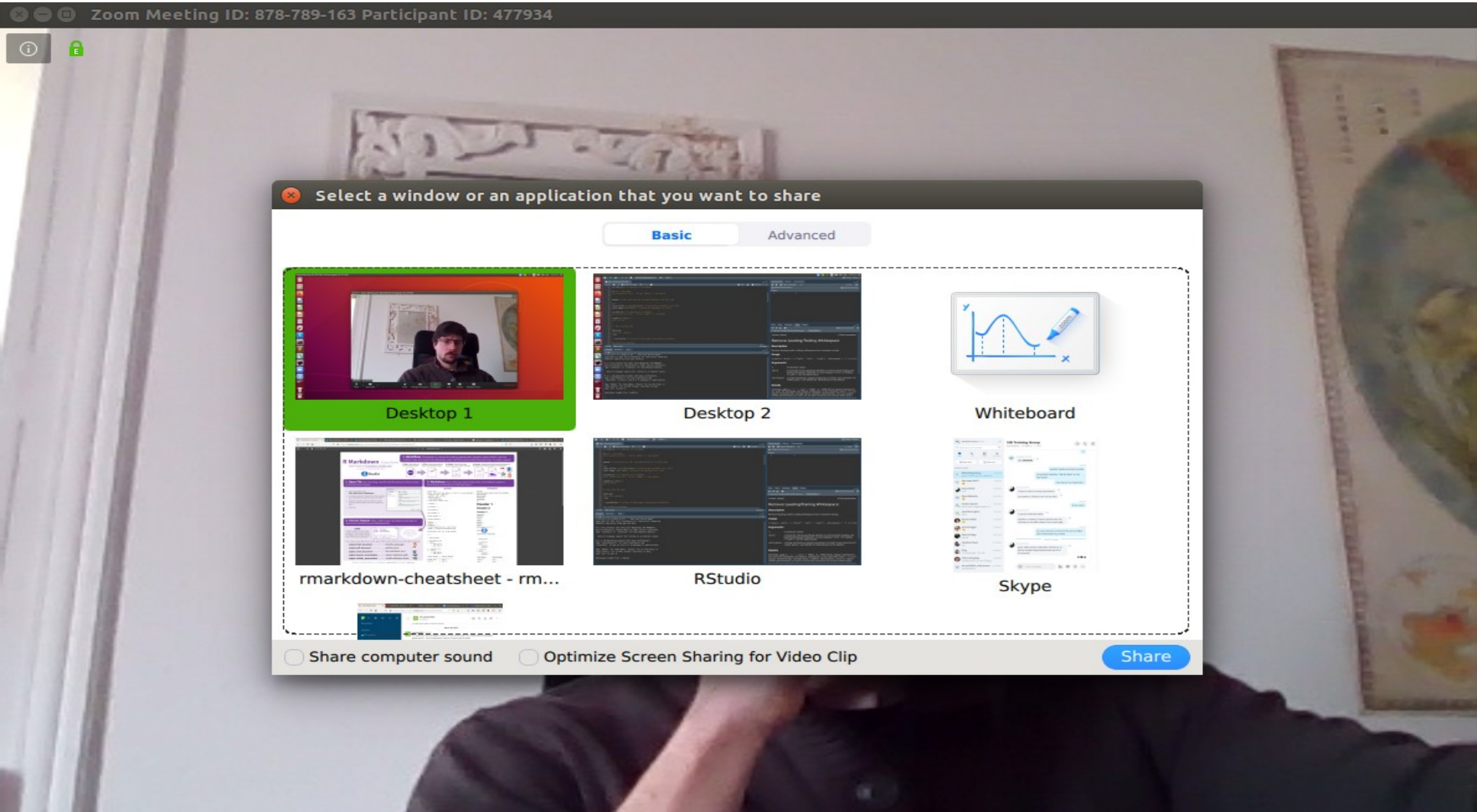
During practicals - breakrooms



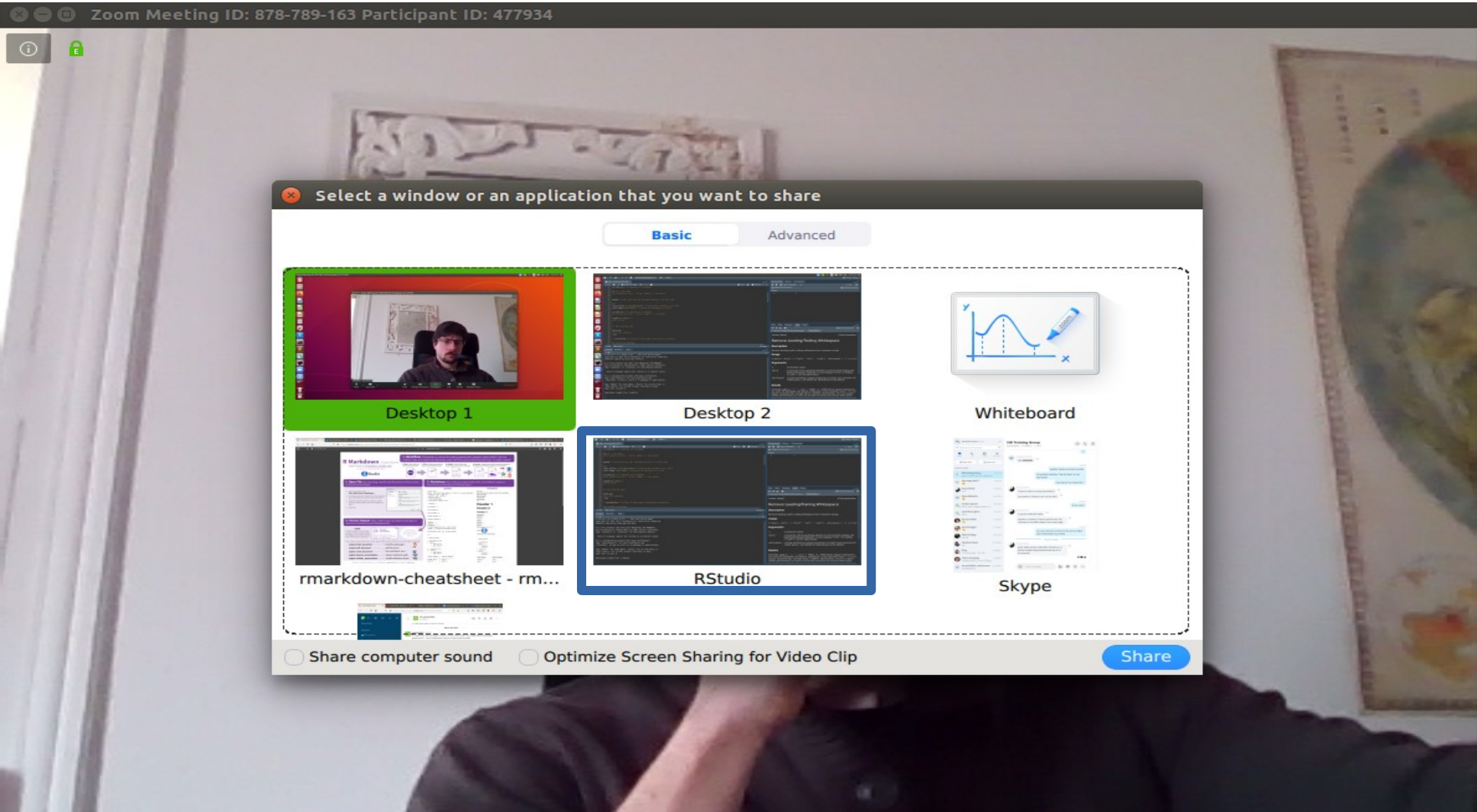
During practicals - breakrooms



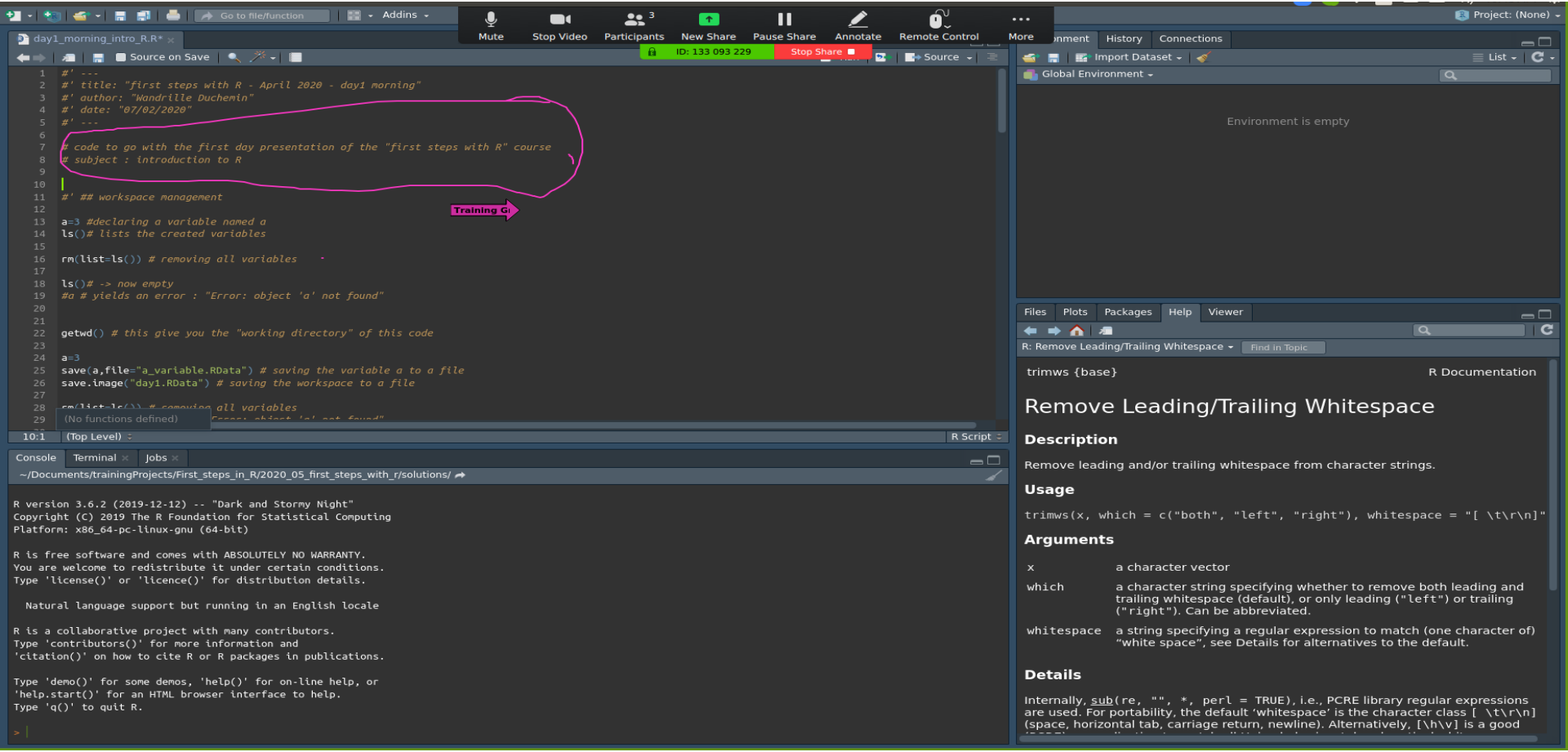
During practicals - breakrooms



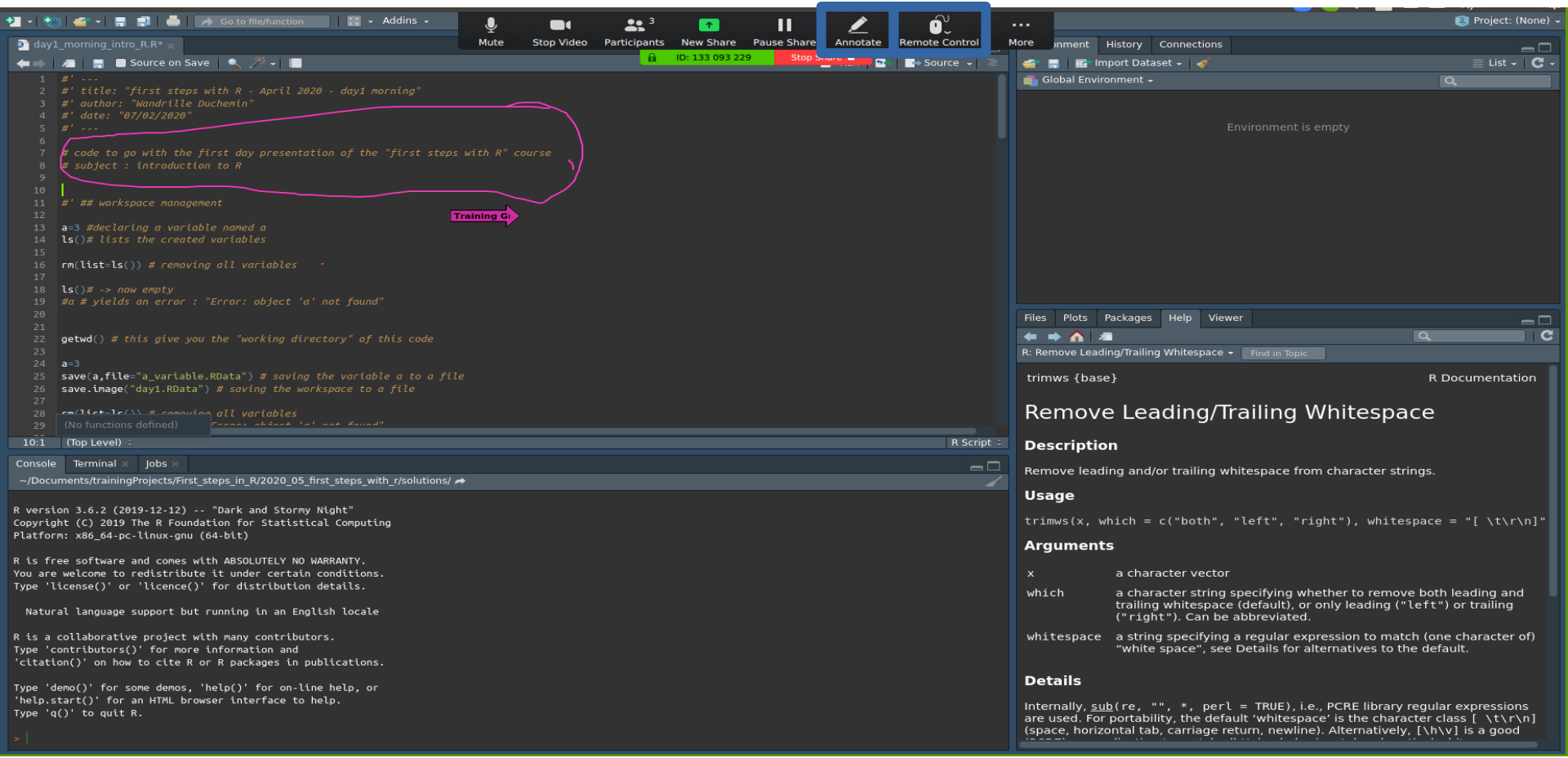
During practicals - breakrooms



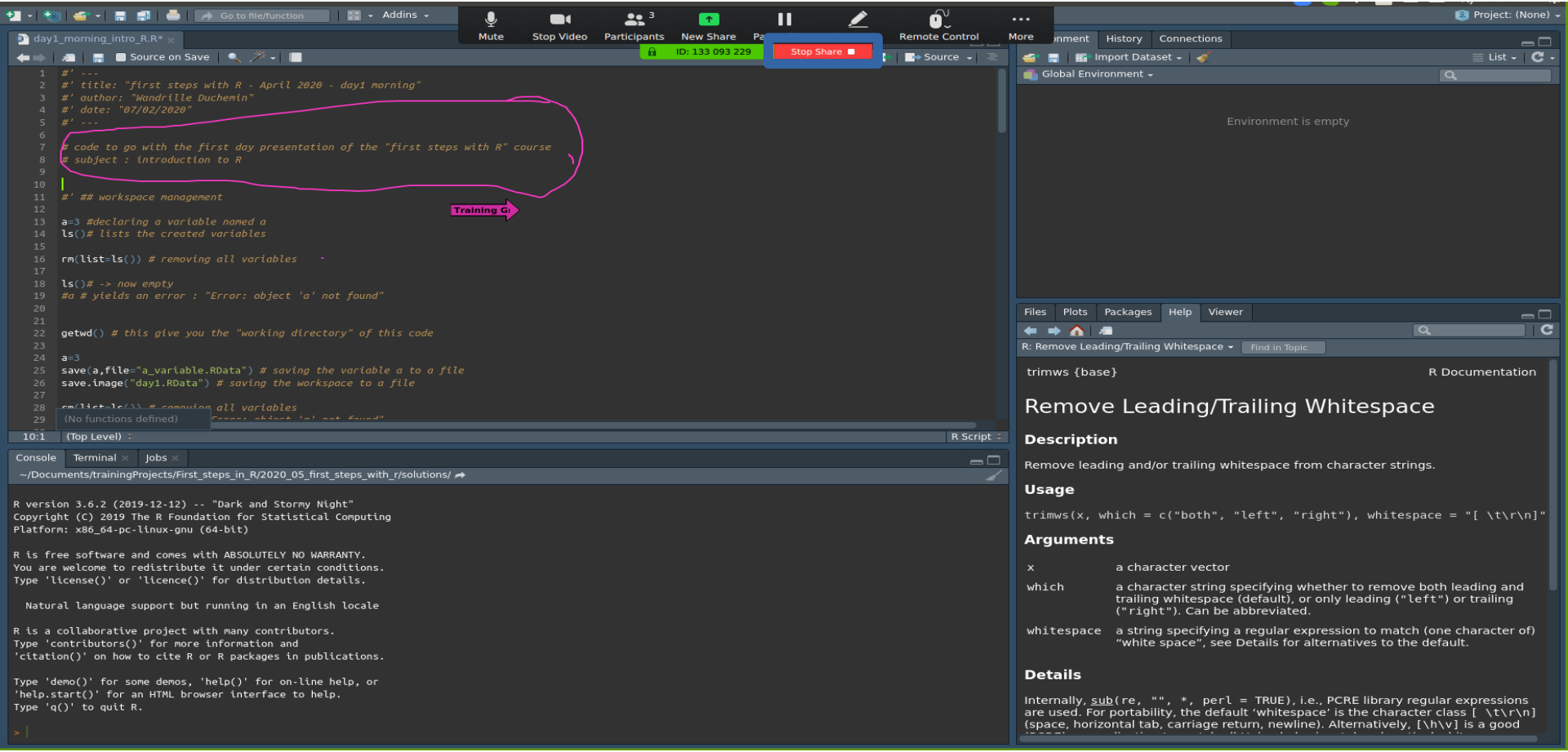
During practicals - breakrooms



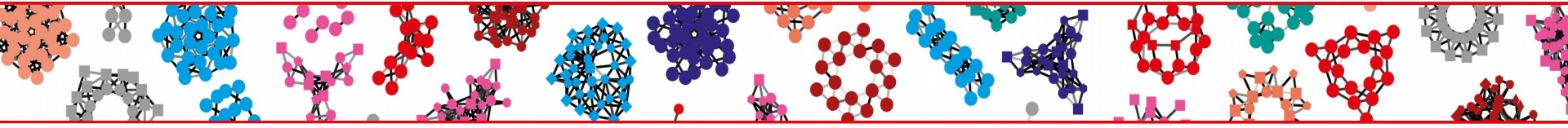
During practicals - breakrooms



During practicals - breakrooms



Overview



01 What is python ? Why use it ?

02 Computer resources and how (not) to use them

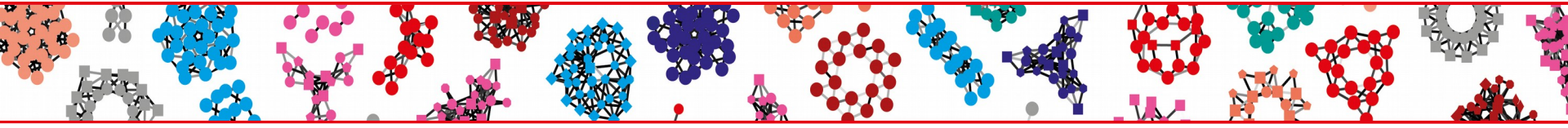
03 FAIR practices in coding

04 Practice, practice and more practice

But first - Getting to know you

- Have you programmed before ? Which language ?
- Any experience with command line ?
- Why do you want to program :
    - to analyze data ?
    - to create scripts that serve as *glue* in my pipeline ?
    - to implement my cool new model ?
    - to become one of the cool kids ?
    - to have something to do on my sundays ?

Overview



01 What is python ? Why use it ?

02 Computer resources and how (not) to use them

03 FAIR practices in coding

04 Practice, practice and more practice

What is Python ?

“Python is an interpreted, high-level, general-purpose programming language.” wikipedia

What is Python ?

“Python is an **interpreted**, **high-level**, **general-purpose** programming language.” wikipedia

What is Python ?

"Python is an **interpreted**, **high-level**, **general-purpose** programming language." wikipedia

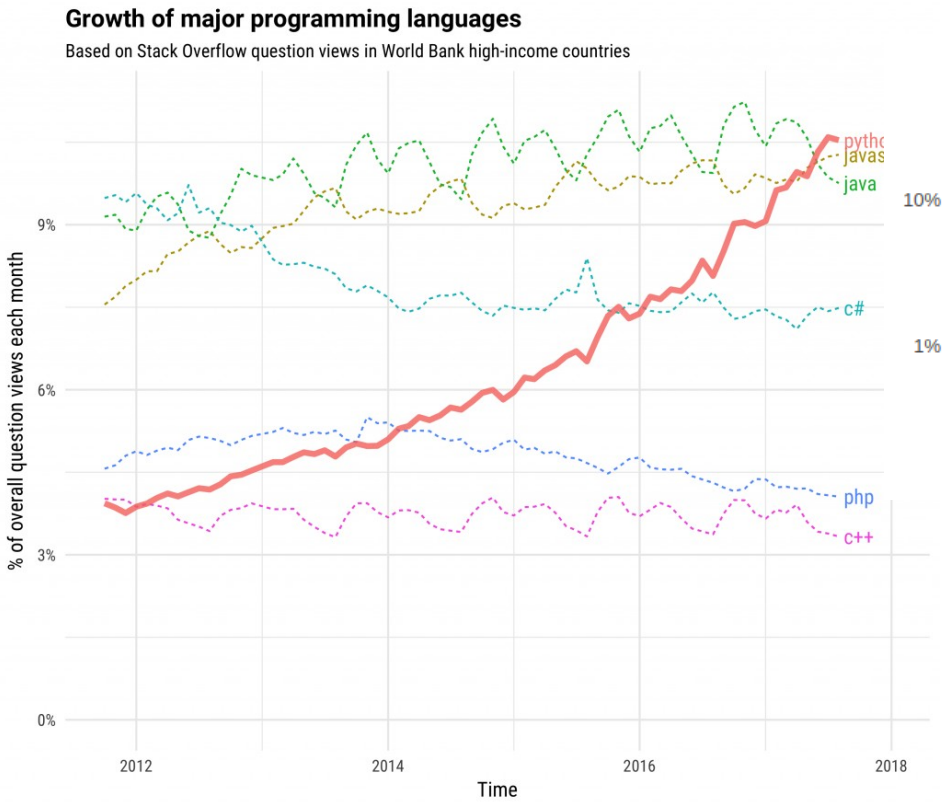
- **Interpreted** : no compilation of the program prior to execution
  - + platform independence (portability), dynamic typing
  - - usually slower, buggy program may still run
- **High level** : abstracted from details of the machine
  - + focus on the application itself
  - - possibly counter-intuitive behaviours
- **General-purpose** : not domain specific, can be used in a broad range of software application
  - + wide user base, usable for any purpose
  - - core language fairly simple → modules for domain specific uses

Python - a brief history

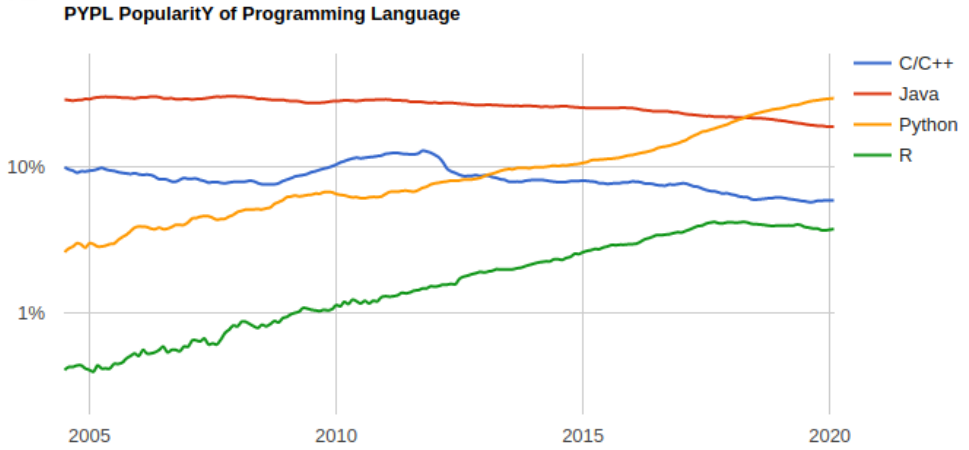
- **1990** : Python begins
- **2000** : Python 2.0
- **2008** : Python 3.0
- ~~**2015**~~ **2020** : end of Python 2.7 support

- **Current version** : 3.8 (as of february 2020)

Python - is it used ?



Source: https://stackoverflow.blog/2017/09/06/incredible-growth-python/



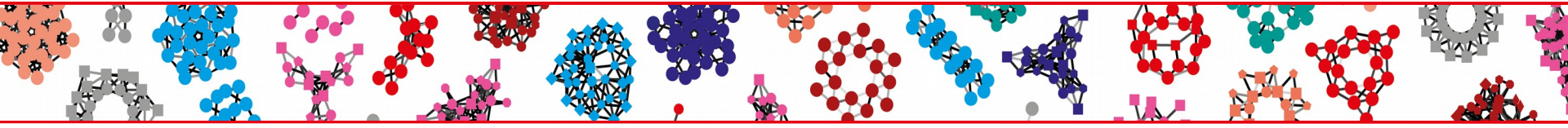
Source: http://pypl.github.io/PYPL.html

Python – sum up

“Python is an interpreted, high-level, general-purpose programming language.” wikipedia

- Easy to learn (you will experience that first hand !)
- Portable : coding for Windows, MacOS or Linux is (almost) the same
- Broad range of applications : can be used for anything !
- Widely used, including in science application
    - Huge community to get help/tutorials
    - Huge number of modules for domain specific applications

Overview



01 What is python ? Why use it ?

02 Computer resources and how (not) to use them

03 FAIR practices in coding

04 Practice, practice and more practice

A computer’s ressources

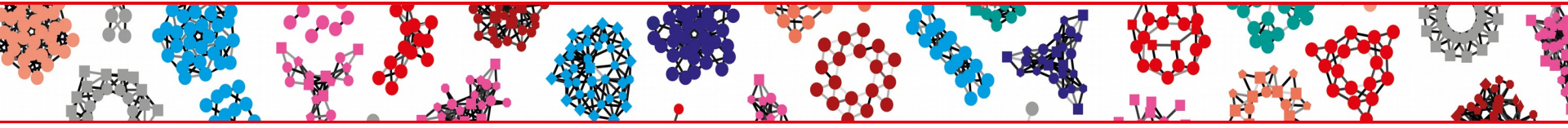
- **CPU** : main computing unit. Nowadays **multi-core** and **multi-threaded**
- **GPU** : graphical processor : like CPU but with **lots of slower cores**
- **RAM** : ‘short term memory’ of the computer. fast
- **Hard disk** : ‘long term memory’ of the computer. Slow
    - HDD : older, ‘cheap’, slow
    - SSD : newer, ‘expensive’, faster
- … external ? file/computation accessed through the network ?

A computer’s ressources

Event	Latency	Scaled
1 CPU cycle	0.3 ns	1 s
Level 1 cache access	0.9 ns	3 s
Level 2 cache access	2.8 ns	9 s
Level 3 cache access	12.9 ns	43 s
Main memory access (DRAM, from CPU)	120 ns	6 min
Solid-state disk I/O (flash memory)	50–150 µs	2–6 days
Rotational disk I/O	1–10 ms	1–12 months
Internet: San Francisco to New York	40 ms	4 years
Internet: San Francisco to United Kingdom	81 ms	8 years
Internet: San Francisco to Australia	183 ms	19 years
TCP packet retransmit	1–3 s	105–317 years
OS virtualization system reboot	4 s	423 years
SCSI command time-out	30 s	3 millennia
Hardware (HW) virtualization system reboot	40 s	4 millennia
Physical system reboot	5 m	32 millennia

Credit:
R.Cabezon,
M.Jacquot

Overview



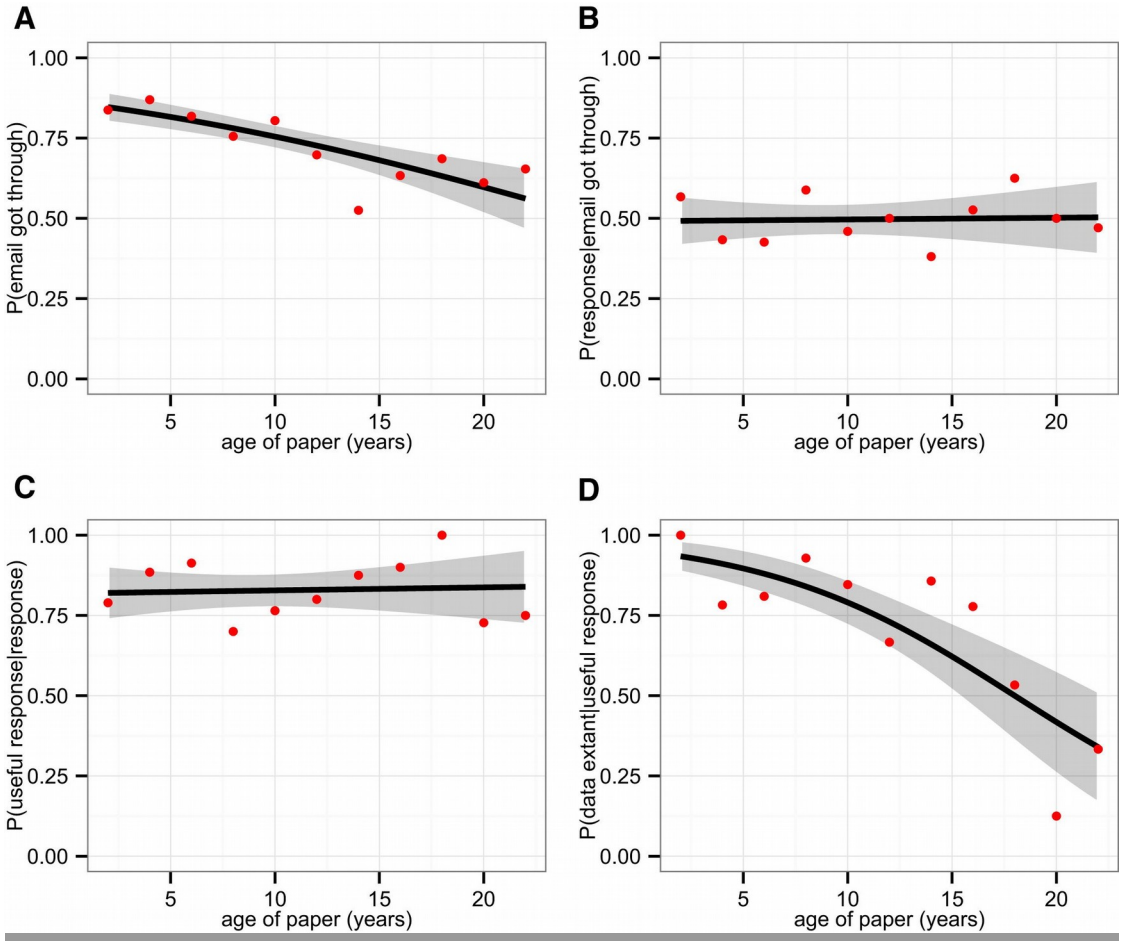
01 What is python ? Why use it ?

02 Computer ressources and how (not) to use them

03 FAIR practices in coding

04 Practice, practice and more practice

The reproducibility crisis



- Vines et al. (2014) Curr. Biol. doi.org/10.1016/j.cub.2013.11.014

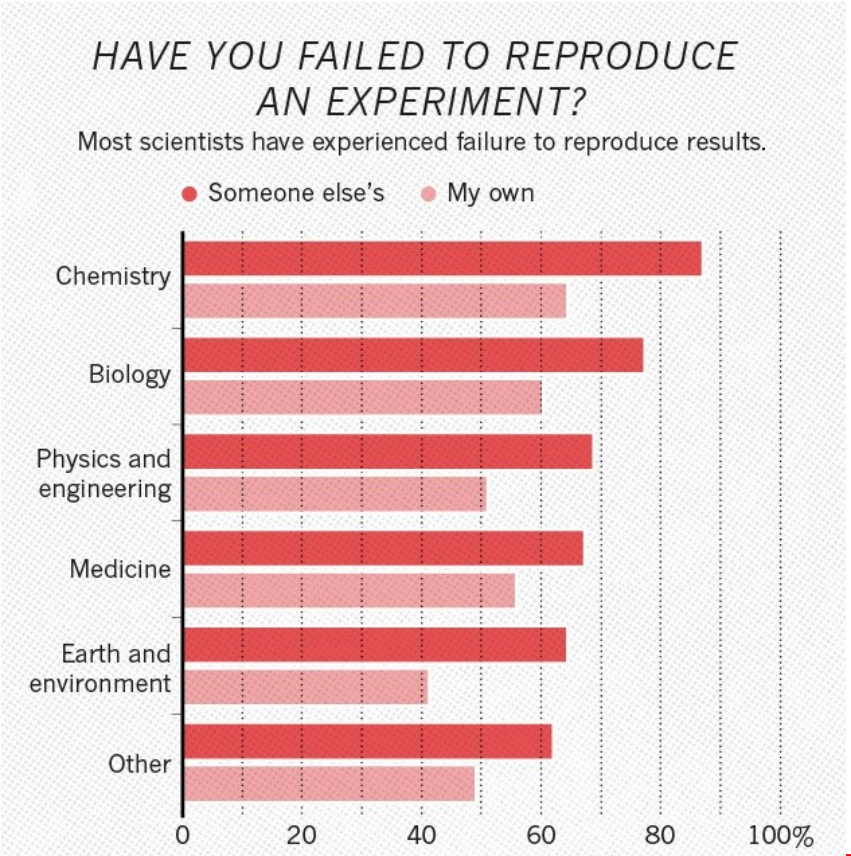
Survey of 516 studies :

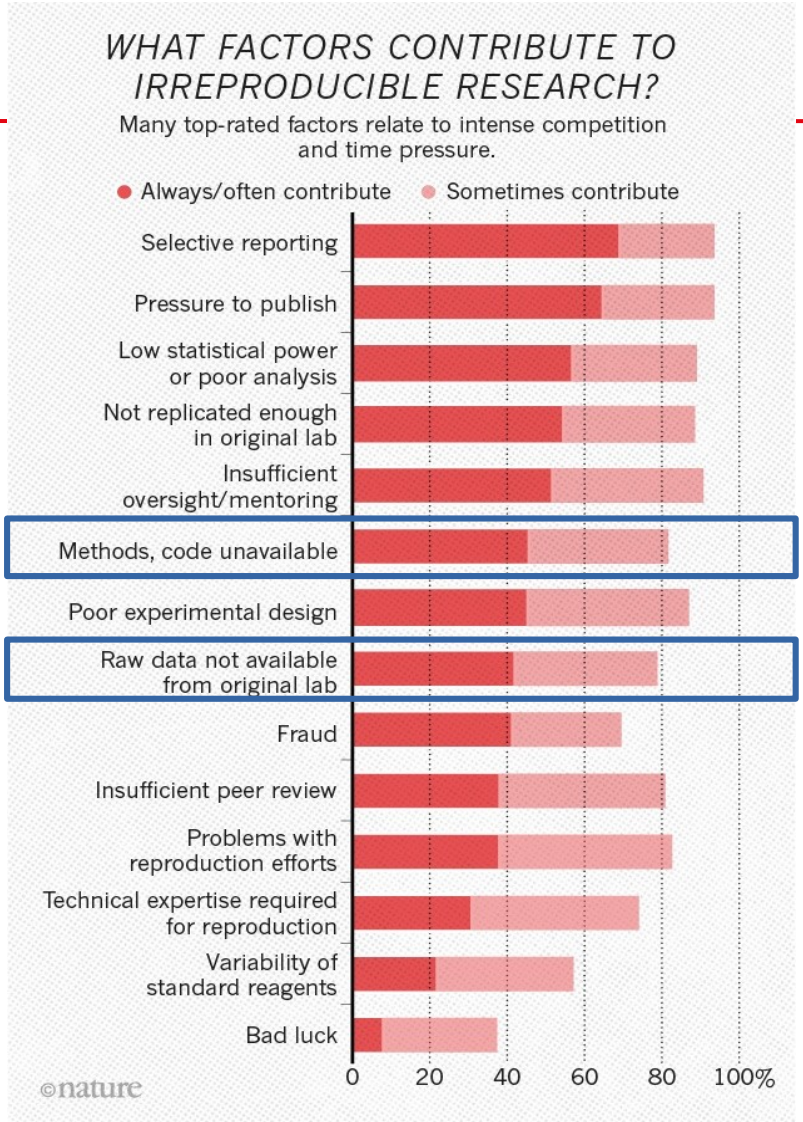
- - 17% data availability per year
- Only 19% retrieval rate after 10 year...

The reproducibility crisis

‘1,500 scientists lift the lid on reproducibility’

M. Baker 2016, nature news feature





The **FAIR** Guiding Principles

Findable : Metadata and data should be easy to find for both humans and computers

Accessible : The exact conditions under which the data is accessible should be provided in such a way that humans and machines can understand them

Interoperable : the (meta)data should be based on standardized vocabularies, ontologies, thesauri etc. so that it integrates with existing applications or workflows

Reusable : Metadat and data should be well described so that they can be replicated and/or combined in different research settings

M.D.Wilkinson, et al. The FAIR Guiding Principles for scientific data management and stewardship. Sci Data. 2016;3:160018. doi:10.1038/sdata.2016.18
https://www.go-fair.org

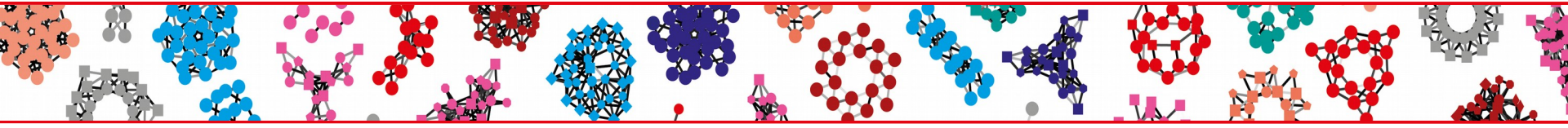
FAIR applied to code

- Write code that also acts as documentation, and clearly communicates the analysis
- Apply the standard you would expect of a ‘wet-lab’ protocol
- Will a reasonably competent colleague understand your code ?
- Will you understand your code in 6 month ??

For this :

- Comment everything
- Use explicit names when naming things
- Jupyter-lab can help you
- Also, you can look at : Ten quick tips for getting the most scientific value out of numerical data

Overview



01 What is python ? Why use it ?

02 Computer ressources and how (not) to use them

03 FAIR practices in coding

04 **Practice, practice and more practice**

Step 1 : how do I use python ?

- 3 main modes of interaction :
- Interactive console
- Python code file (.py)
- Jupyter notebook

Live demo time

The following slides are here for posterity

Python – using the console

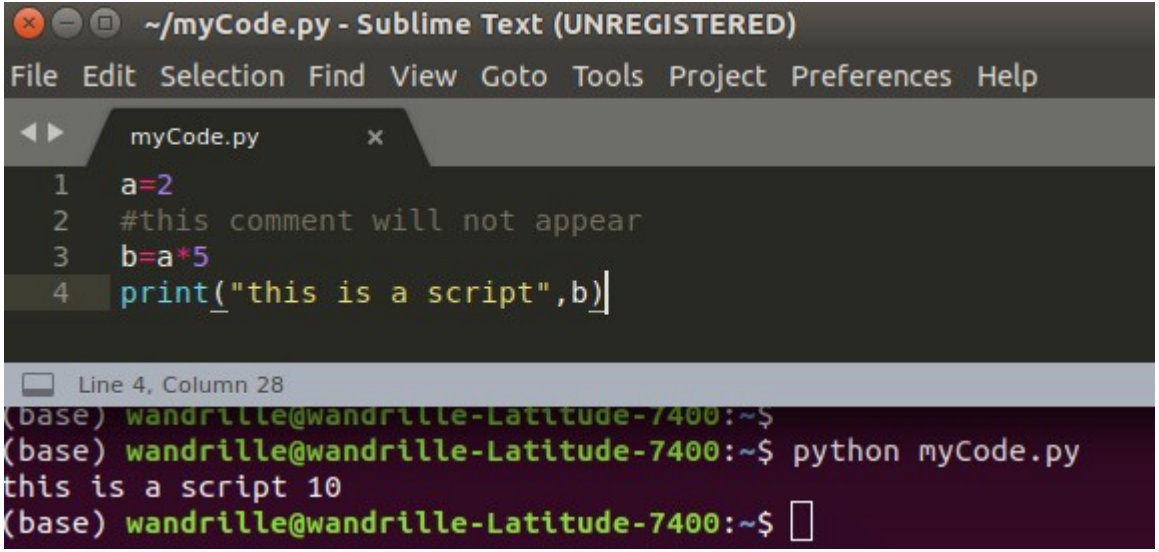
```
(base) wandrille@wandrille-Latitude-7400:~$ python
Python 3.7.4 (default, Aug 13 2019, 20:35:49)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> a=3
>>> b=4
>>> a*b + 7
19
>>> print("this is an interactive console")
this is an interactive console
>>>
```

- **Interactive** : the code is executed as you press ‘Enter’
- Great for **testing** things out
- But, you keep **no trace** of your workflow/environment

Only use it to test little bits of code

Python – writing a code in a .py file

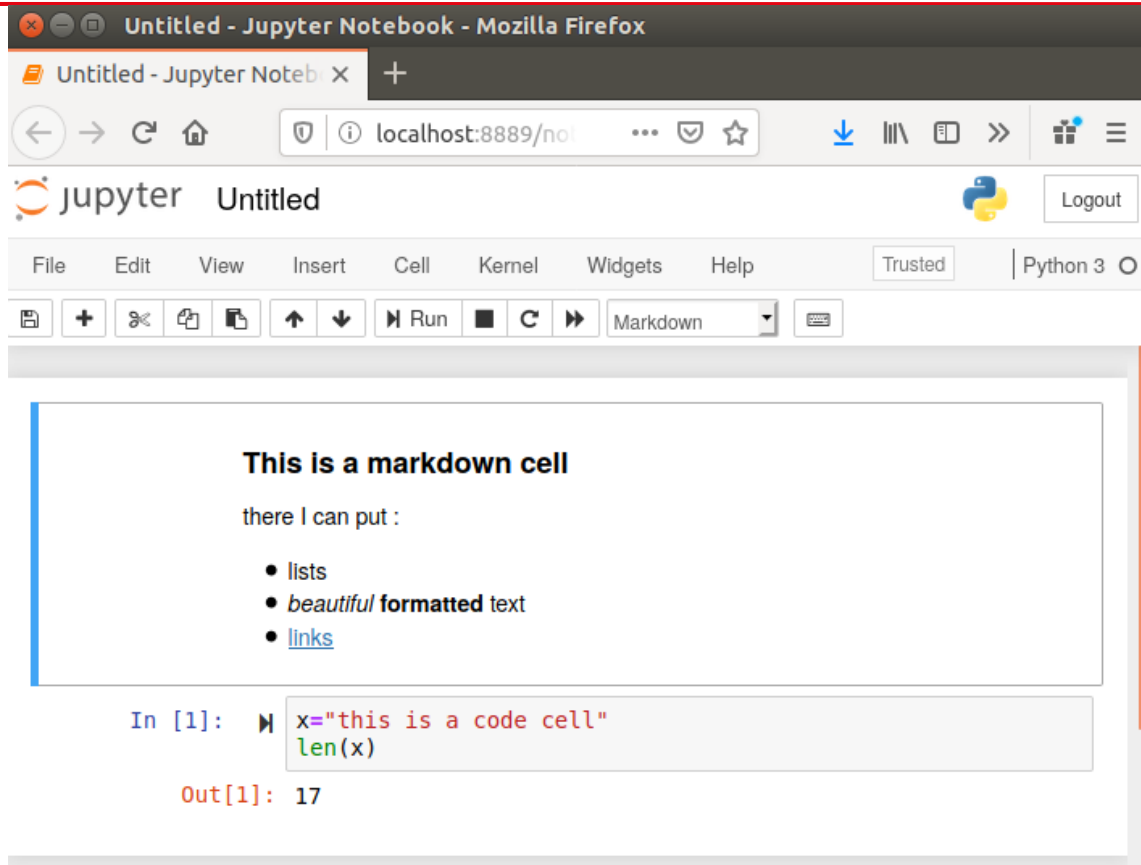
- Write a script, then execute it
- Main way python code is
- shared
- Ideal for standalone programs
- and modules
- Code and results are kept separate (may be a good or a bad thing)



→ General purpose coding, “operational script”

Python – jupyter

- Browser based interface
- Interlace markdown and python ‘cells’
- Execute code cell by cell
- Commentary, code, and results together in the same file
- Visually pleasant and fairly ergonomic



Ideal “analysis scripts” : helps reproducibility of results