PYTHON 3

FOR PROGRAMMERS

SCHEDULE

- Motivation for learning and using Python
- Skim through the Python basics
- Overview of more advanced Python features
- Debugging Python code
- Installing 3rd party Python packages
- Collaborate on a Python project

SO WHY SHOULD I USE PYTHON?

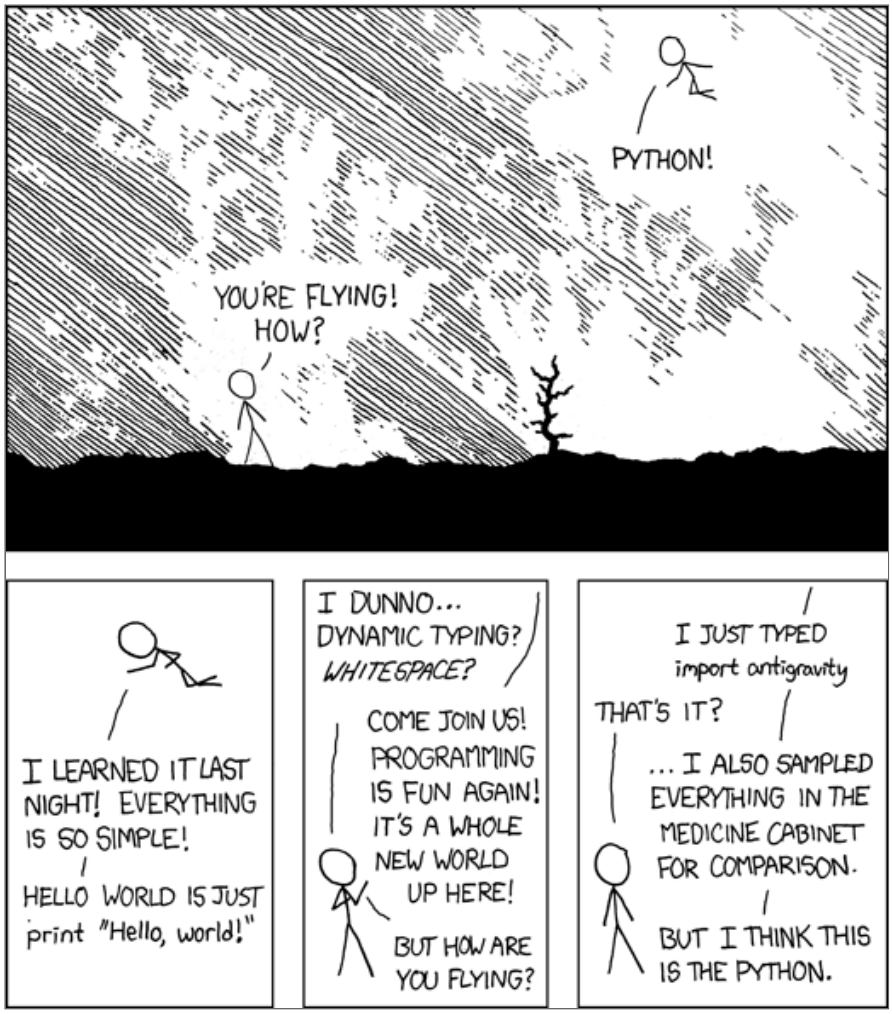
- free
- rising
- mature
- readable
- high-level
- documented
- general-purpose

Stackoverflow trends

PYTHON HAS A HUGE SET OF

WELL-MAINTAINED LIBRARIES

WHICH SIMPLIFY COMPLEX PROJECTS MASSIVELY



THE ZEN OF PYTHON

```
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one -- and preferably only one -- obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

BOOT YOUR NOTEBOOKS

SLIDES

https://escodebar.github.io/trainings/python/

WHERE DO I START?

There is an official **tutorial**!
The **Python documentation** is awesome!

...but let's take a look at it together!

INTERACTIVE INTERPRETER

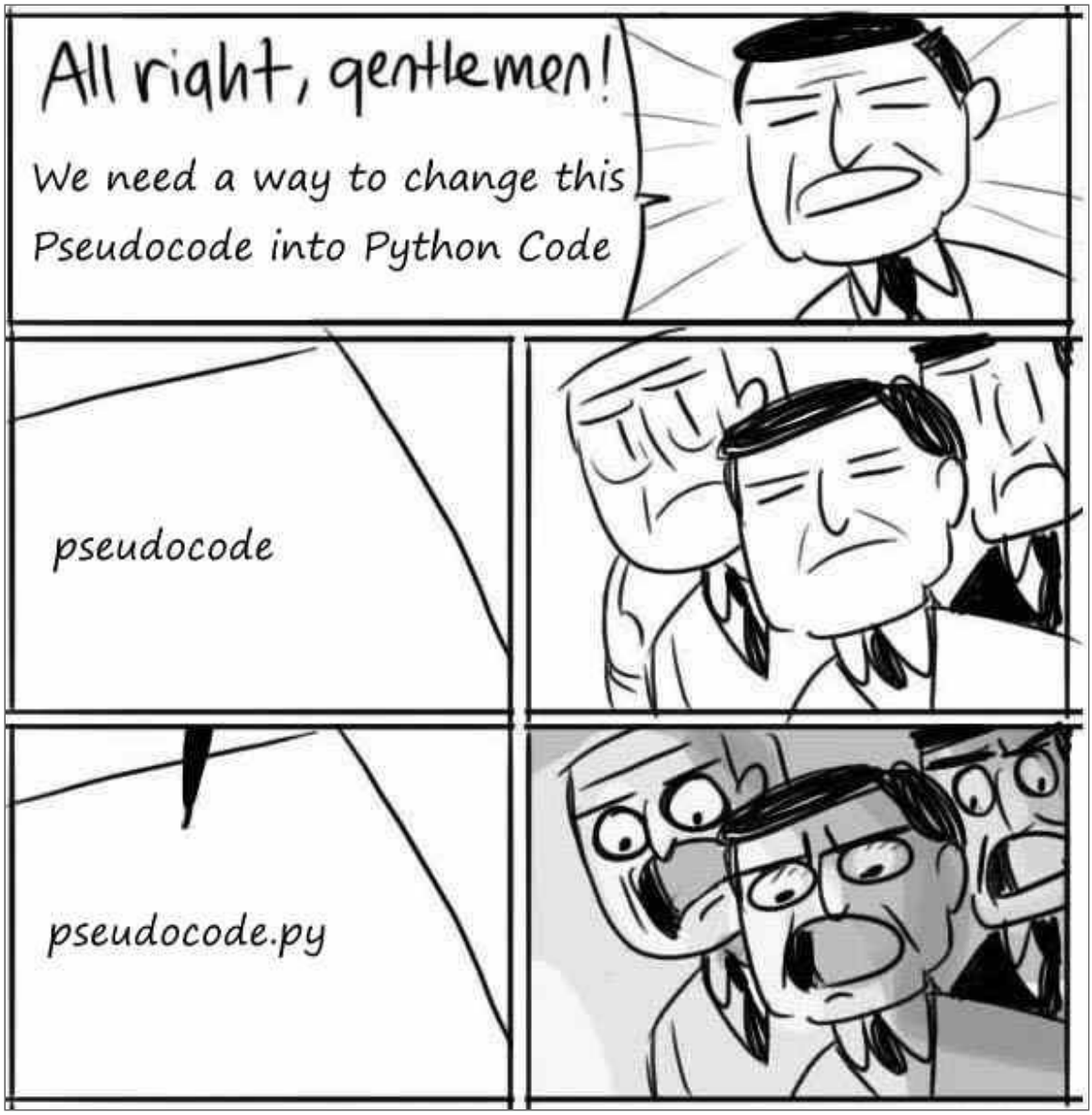
Python is an interpreted language.
The interpreter comes with an **interative mode**.

...we need to discuss

THE PYTHON SYNTAX

Let's start with the **informal introduction**

...then take a closer look at **more flow control tools**.



SLIGHTLY

MORE ADVANCED

PYTHON FEATURES

GENERATORS

Build complex iterables using **generators**.

Python comes with

COMPREHENSIONS

for commonly used data structures like **lists**, **dicts** and **sets**.

CLASSES

Describe your own object types using **classes**.

CONTEXT MANAGER

Setup and tear down a context using **context managers**.

DEBUGGING

PYTHON CODE

PYTHON DEBUGGER

Debug your code using **the Python debugger**.

INTROSPECTION

Inspect live objects using the **inspect module**.

THERE IS A

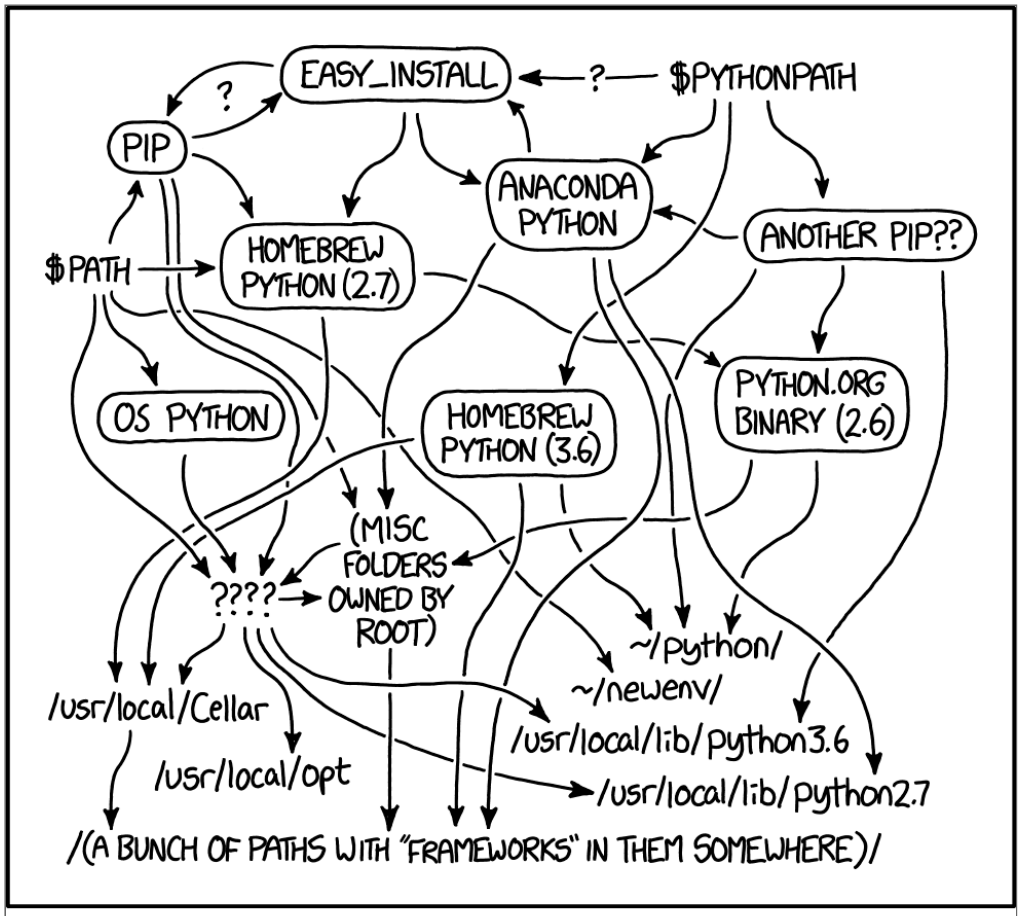
PYTHON PACKAGE

FOR (ALMOST) EVERYTHING

INSTALL 3RD PARTY PACKAGES

Install all packages available on **pypi.org** using **pip**.

CREATING ENVIRONMENTS FOR ISOLATING APPLICATIONS



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED
THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

File failed to load: file:///home/marie/Documents/CAS_data_scien

VIRTUAL ENVIRONMENTS

Run your applications in clean **Python virtual environments**.

DEVELOPING A PYTHON PACKAGE

PYTHON PACKAGES

Create your own **Python packages** like a pro.

SCRIPT ENTRY POINT

Define your application's entry point with **`__main__`**.

LET'S COLLABORATE!