Four years of Python

Tales of an (un)experienced Pythonhista

PyCon Italia 03/06/2022

Duarte O.Carmo

duarteocarmo.com - @duarteocarmo

Who the *hell* am I?

- */du-art/*
- From Lisbon, based in Copenhagen
- I like running a lot
- Ancora imparando l'italiano (ragazza di Ancona)

- *Past:* Strategy, Product Management, New Ventures, Management Consulting
- *Now*: ML Engineer @ Amplemarket / Contractor
- *Always*: Python (well, Computers in general)

Big thanks to Amplemarket

amplemarket.com/careers

This is not a technical talk

`if experience_years <= 4:`

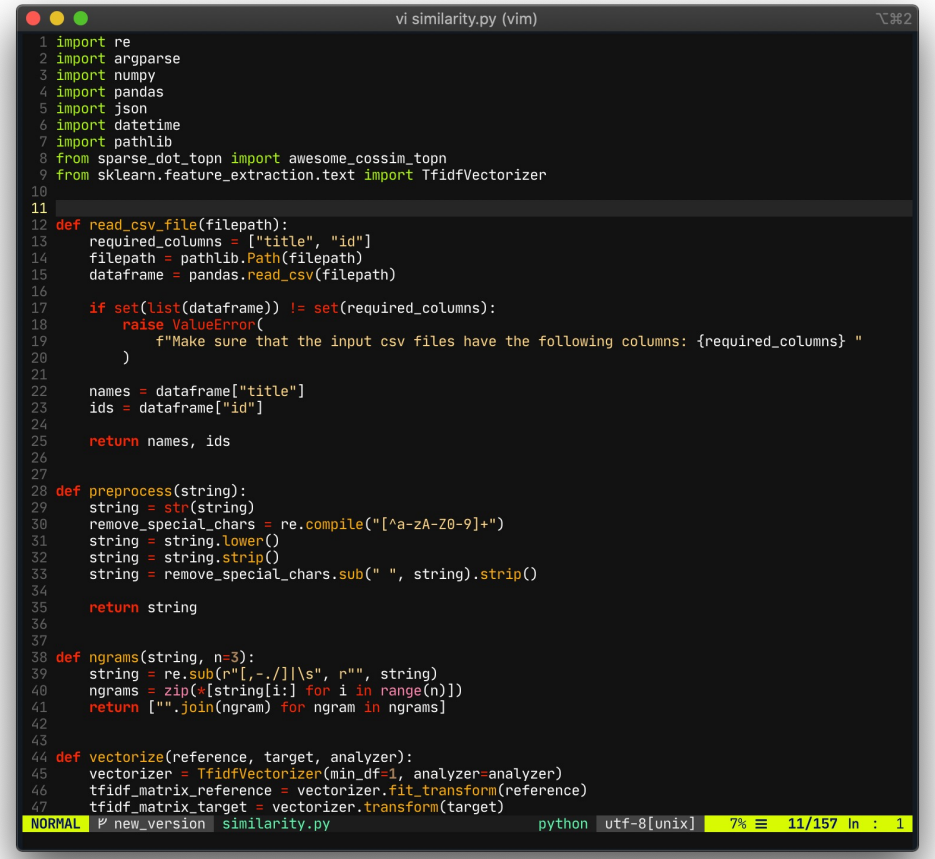
- How to be a better dev
- Tips and tricks
- Opinionated advice

`else:`

- Reminisce
- Celebrate
- Stay intentional

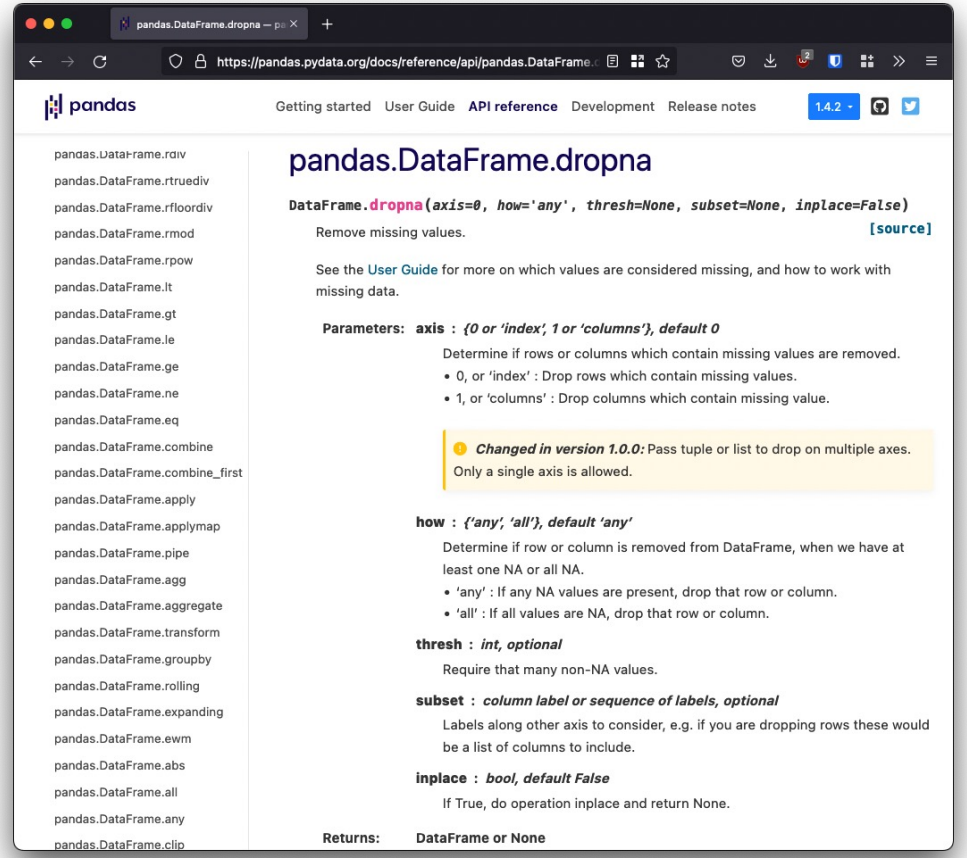
1 | Reading is better than googling

When we start, we have *superpowers*



- Autocomplete
- Google
- Stack overflow
- Nails everywhere
- Pip install the world
- But.. We forget quickly

But there’s quite nothing like reading



- What does it do?
- Options?
- Default behaviors
- Maybe I can re-use this
- **It *actually* sticks**

2 | Explicit is better than implicit

“I can make this program shorter”

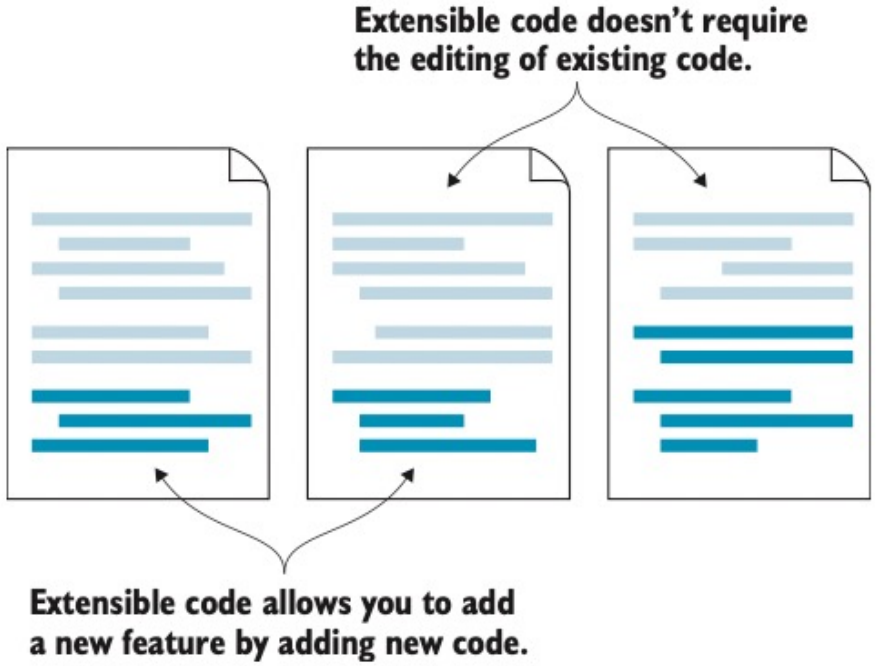
```
print len((lambda lookandsay: (lambda func1, in1, current_depth1, target_depth1:
func1(func1, in1, current_depth1, target_depth1))(lambda self, _in, current_depth,
target_depth: _in if current_depth == target_depth else self(self, ''.join('%d%s'
% (len(seq), seq[0]) for seq,trash in re.findall(r'((\d)\2*)', _in)),
current_depth + 1, target_depth), lookandsay, 0, 40))(_input))
```

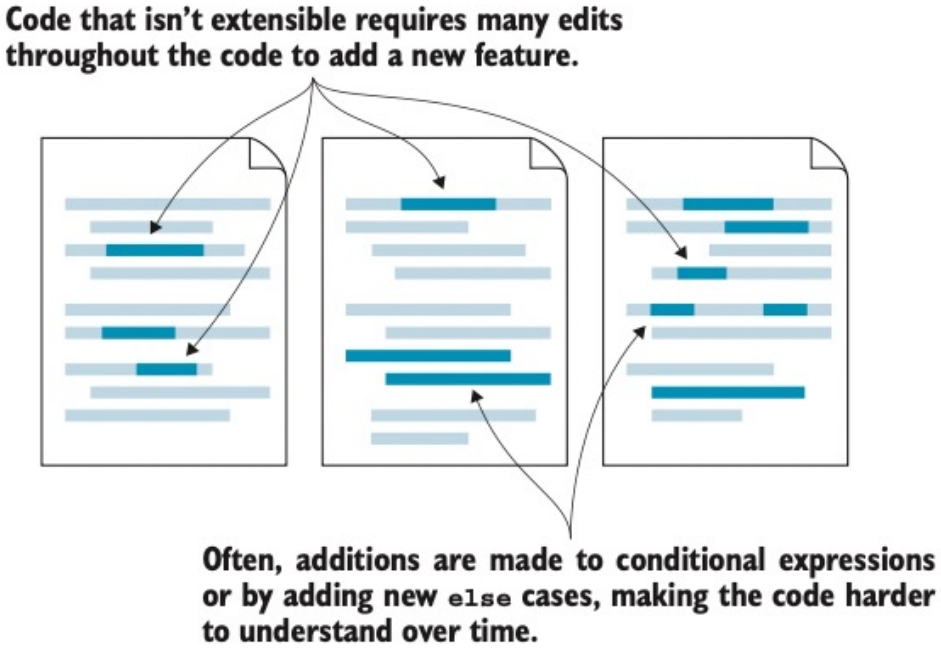

```
print len((lambda lookandsay: (lambda func1, in1, current_depth1, target_depth1:
func1(func1, in1, current_depth1, target_depth1))(lambda self, _in, current_depth,
target_depth: _in if current_depth == target_depth else self(self, ''.join('%d%s'
% (len(seq), seq[0]) for seq,trash in re.findall(r'((\d)\2*)', _in)),
current_depth + 1, target_depth), lookandsay, 0, 40))(_input))
```

Less lines = better code

```
strings = [x if isinstance(x, str) else pass for index, x in enumerate(list_)]
```

```
Scheduler, State Machine, Abstract, Controller, Operator…
```

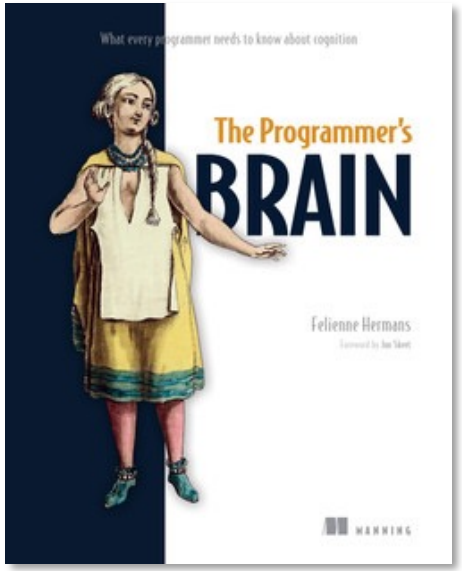





[Reference: Practices of the Python Pro]

Less lines != better code

- Readable
- Understandable
- Changeable
- Shareable
- Enjoyable



3 | First make it work, then make it pretty

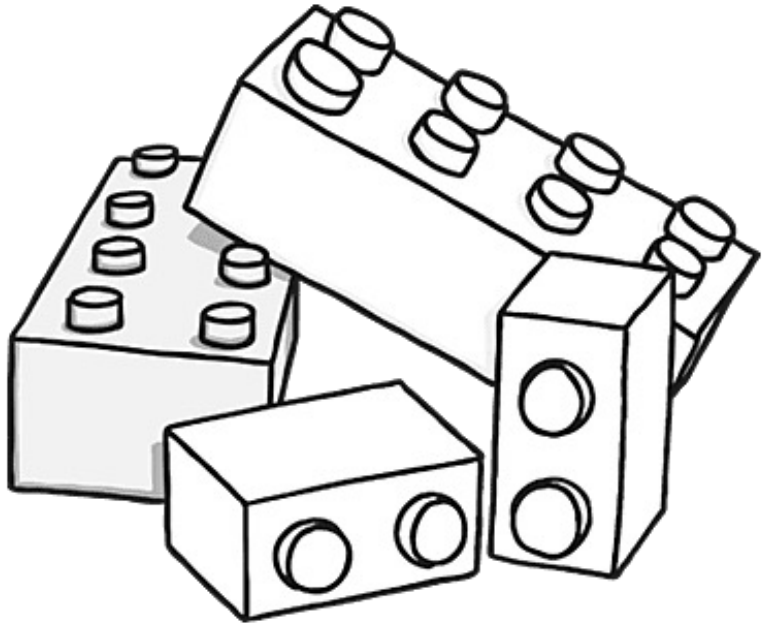

DUNKIN' DONUTS
SHY BIR

“The Data Management Framework”

(a.k.a. Reinventing the wheel)

- **Me**: Let’s not do it
- **Vendor**: Don’t do it
- **Partners**: Don’t do it
- We **still** did it.
- I hope it went well (*kinda*)

First make it work, then make it pretty



- The bare minimum
- Catching all exceptions
- 100% code coverage
- That weird edge case
- Do users care?
- What NOT to write

More ranting: duarteocarmo.com/blog/simple-software

4 | Test early, test often

Tests are a mirrage....

- 80% projects don’t
- Value is not obvious
- Users don’t see them
- Bugs can still happen

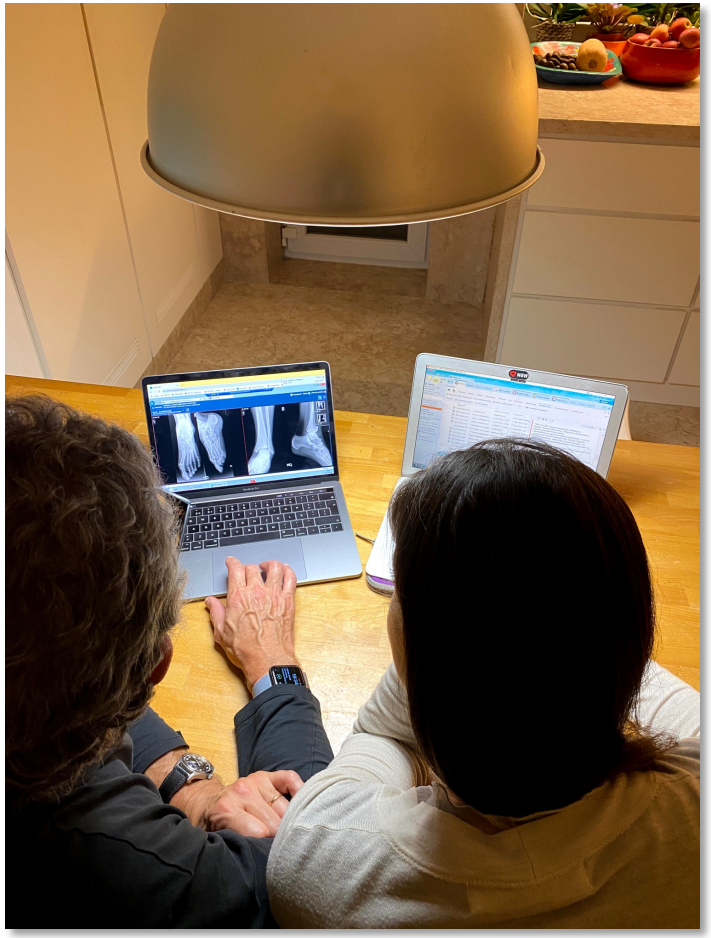
Tests are a mirrage....But they matter!

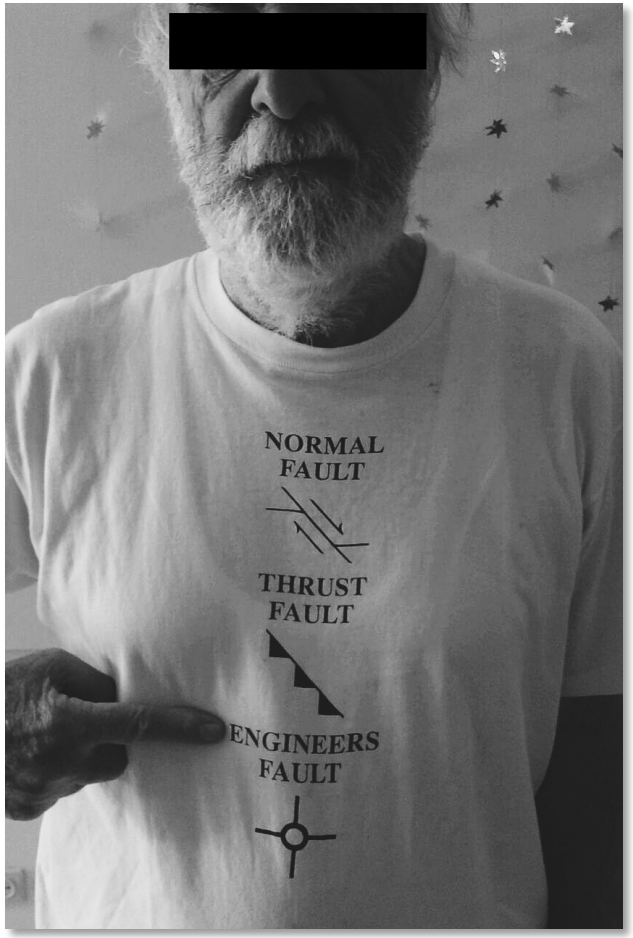
- 80% projects don’t
- Value is not obvious
- Users don’t see them
- Bugs can still happen

- Yours should
- Deploy with confidence
- What if the app goes down?
- Minimizing vs. Eliminating

5 | Continuously learn

Python is our craft

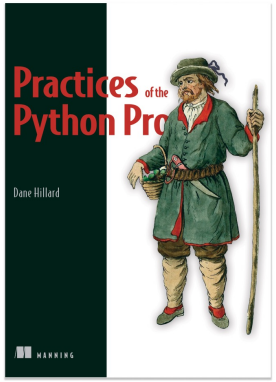


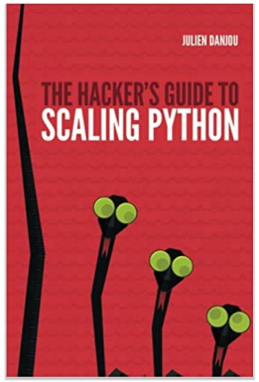

NORMAL
FAULT
THRUST
FAULT
ENGINEERS
FAULT

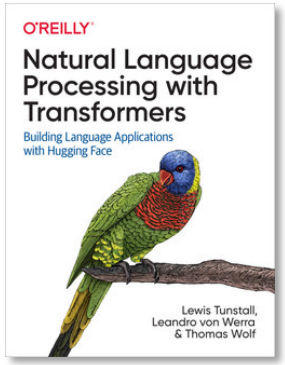

EINSTEIN
TINHA
RAZÃO?
Testando
a Teoria
da Relatividade
Geral
Clifford
M.Will
Portugal
Madeira
NOVO
ATLAS
GEOGRÁFICO

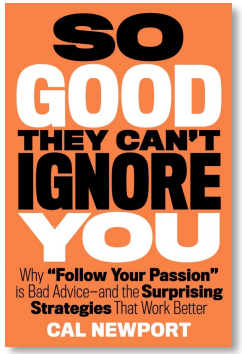
We should be masters of our craft

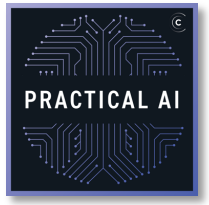
- Study
- Stay up-to-date
- Lean regularly
- Build things
- Give back and write











An OCD list of resources

Books

Practices of the Python Pro

Hacker's guide to scaling Python

Designing Data-Intensive Applications

Serious Python

Tutorials

Flask Mega-tutorial

RealPython

Stack Abuse

Kaggle + GitHub

YouTube

CodingTech

Sentdex

Abhishek Thakur

MLOPs Community

Podcasts

Talk Python to Me

Python Bytes

Podcast.__init__

Practical AI

News

PyCoder's Weekly

Medium

Awesome Python Weekly

Reddit RSS

...

Thank you, questions?