

Getting to Intermediate

John Ouellette / jouell@gmail.com

- Not a Software Engineer
- Automater / Python practitioner

PyCon South Africa

<https://github.com/jouellnyc/PyConZA>

Where is my Python Alpaca Book?



Why / Why Now?

Now is a great time!

- Lots of leveling up resources!
- ML/DevOps/Cloud/Data Science

Who is this for?

Beginning Pythonistas who want:

To be effective and deliver projects

- * You don't need to "know everything"
- * You can have a big impact!

Outline

- Mindset and Getting Started
- Work product - working with a goal
- Initial concepts to fast track
- My project - how my project evolved
- Resources - books / videos / free info!

Outline

- **Mindset and Getting Started**
- Work product - working with a goal
- Initial concepts to fast track
- My project - how my project evolved
- Resources - books / videos / free info!

Mindset – Imperfect def():s:

- **Beginner**

for loops | lists, dicts, strings| basic file ops

- **Intermediate**

Networking/Iterators/Decorators/Exceptions/GenExp/
pdb/Well Documented code/(ADD/REMOVE MORE HERE)

- **Advanced**

Framework creators/ ABCs / OOP “Patterns” / High
Performance / Meta Programming/G.V.R on speed dial

Mindset – A Proposal

Work towards a Work Product

- ~Fairly important
- Affects more than just YOU
- Put it “out there”

It drives you and guides you.

Outline

- Mindset - Why pursue this goal?
- **Work product - working with a goal**
- Initial concepts to fast track
- My project - how my project evolved
- Resources - books / videos / free info!

Work product – Why?

You **need** to have a **burning desire**.

(or else NetFlix will win your attention)

- Career / Future Proof / Cause / Problem
- ~~Be a 'Python Ninja', Impress friends~~

Work Product – What

- **Complete** but minimally viable project
- Don't *just do* coding challenges
- Achievable, yet will **stretch** your skills
- Build **momentum** with small **wins**

Work Product – How

- Put your project on GitHub in 4 months
- Python Website deployed in 5 months
- Deploy Work code to prod in 6 months

Outline

- Mindset and Getting Started
- Work product - working with a goal
- **Initial concepts to fast track**
- My project - how my project evolved
- Resources - books / videos / free info!

Concepts - Write Good Functions

Everyone needs to write functions!

“I get lost trying assemble functions”

Concepts - Good Functions

- Small (Jack Diederich says 5-20 lines)
- Does one thing, not 7
- Boring, don't chase exotic features
- Golden Rule

Concepts - A pretty good function

```
def create_url_map(file_name):  
    """  
    Return a dictionary (URL : city, state)  
    given a file containing "City, State = URL"  
  
    Parameters  
    -----  
    file_name : file  
  
    Johannesburg, South Africa = https://www.joburg.org.za/  
    Durban, South Africa = http://www.durban.gov.za/  
  
    Returns:  
    {dictionary}  
    """  
    with open(file_name) as fh:  
        contents = fh.readlines()  
        city_links = {}  
        for one_line in contents:  
            city_text, link = one_line.split("=")  
            link = link.strip()  
            city_text = city_text.strip()  
            city_links[link] = city_text  
    return city_links
```


Concepts – Not a great function

```
def cmap(f):  
    with open(f) as fh:  
        file = fh.readlines()  
        city_links = {}  
        for x in file:  
            c, l = x.split("=")  
            l = l.strip()  
            c = c.strip()  
            city_links[l] = c  
        return city_links  
  
print(cmap('south_africa.txt'))
```

Learn Logging

<https://docs.python.org/3/howto/logging.html>

```
import logging
logging.basicConfig(
    filename='/tmp/out.txt',
    format='%(asctime)s %(levelname)s %(message)s',
    level=logging.INFO,
    datefmt='%Y-%m-%d %H:%M:%S'
)
```

```
In [4]: cat /tmp/out.txt
2021-09-23 12:08:46 WARNING No data found
2021-09-23 12:09:02 ERROR No File found
2021-09-23 12:13:09 CRITICAL Bad Type
2021-09-23 12:18:10 WARNING No data in file
```

Get Comfortable with Exceptions

- Least amount of code in a try block
- Start with Network and File Operations
- Log failures - capture details
- Decide how to handle them

Get Comfortable with Exceptions

```
try:
    file = 'south_africa.txt.DNE'
    create_url_map(file)
except FileNotFoundError as e:
    print(dir(e),end="\n\n")
    print(e.__class__)
    print("strerr",e.strerror)
    print("file",e.filename)
    logging.warning(e)
    logging.warning(f"YourText {file} ", exc_info=True)
except Exception as e:
    logging.critical(f"{e}", exc_info=True)
else:
    logging.info("Another run w/o exceptions!")
finally:
    logging.info("Always Awesome!")
```

```
['_cause__', '__class__',
 '__context__', '__delattr__',
 '__dict__', '__dir__', '__doc__',
 '__eq__', '__format__', '__ge__',
 '__getattr__', '__gt__',
 '__hash__', '__init__',
 '__init_subclass__', '__le__',
 '__lt__', '__ne__', '__new__',
 '__reduce__', '__reduce_ex__',
 '__repr__', '__setattr__',
 '__setstate__', '__sizeof__',
 '__str__', '__subclasshook__',
 '__suppress_context__',
 '__traceback__', 'args',
 'characters_written', 'errno',
 'filename', 'filename2', 'strerror',
 'with_traceback']

<class 'FileNotFoundError'>
strerr No such file or directory
file south_africa.txt.DNE
```

```
In [2]: cat /tmp/out.txt
2021-10-06 13:03:07 WARNING [Errno 2] No such file or directory: 'south_africa.txt.DNE'
2021-10-06 13:03:07 WARNING YourText south_africa.txt.DNE
Traceback (most recent call last):
  File "/home/john/gitrepos/jouell/pc/sa_exp_out.pgood.raise.py", line 27, in <module>
    create_url_map(file)
  File "/home/john/gitrepos/jouell/pc/sa_exp_out.pgood.raise.py", line 14, in create_url_map
    with open(file_name) as fh:
FileNotFoundError: [Errno 2] No such file or directory: 'south_africa.txt.DNE'
2021-10-06 13:03:07 INFO Always Awesome!
```

Don't Dread Decorators

A decorator is a function that receives a function and enhances it by

- - wrapping the passed function with a modification
- - returning the passed function to the original function name

```
def there(some_func):  
  
    def wrapper():  
        some_func()  
        print('there')  
    return wrapper  
  
@there          #same as hello=there(hello)  
def hello():  
    print('hello')  
  
hello()
```

Don't Dread Decorators

A decorator is a function that receives a function and enhances it by

- - wrapping the passed function with a modification
- - returning the passed function to the original function name

```
def there(some_func):  
  
    def wrapper():  
        some_func()  
        print('there')  
    return wrapper  
  
@there          #same as hello=there(hello)  
def hello():  
    print('hello')  
  
hello()
```

```
In [11]: hello()  
hello  
there
```

Decorator Story

“Surely these are all different, right?”

- `$ cat HealthCheck.py`
`from testService1 import *`
`from testService2 import *`
`from testService3 import *`
`from testService4 import *`
`from testService5 import *`
`from testService6 import *`

6 Modules to 1; 28 defs() to 8!

Outline

- Mindset and Getting Started
- Work product - working with a goal
- Initial concepts to fast track
- **My project - how it evolved**
- Resources - books / videos / free info!

Minimally Viable Python Project

GOAL: Create a Web App that queries stocks and calculates growth rates in 4 months

Enter a Stock to See it's History



Stock:

Image courtesy of Alexander Schimmeck - https://unsplash.com/photos/AoI2E_7E11w

Minimally Viable Python Project

AAPL	Market Cap: 2.4 T	Rev TTM: 347.16	Net INC TTM	ROIC
BookValue : 3.882	PriceToBookRatio : 37.56	PriceToSalesTTM : 6.92	TrailingPE: 28.0	Currency : USD
Revenue and Net Income				
4Years	Revenue: 271.64 B	RevGrowth: 5.94 %	NetIncome: 57.41 B	NetIGrowth: 5.88 %
3Years	Revenue: 256.6 B	RevGrowth: 5.97 %	NetIncome: 55.26 B	NetIGrowth: 6.54 %
2Years	Revenue: 265.6 B	RevGrowth: 10.98 %	NetIncome: 59.53 B	NetIGrowth: 14.15 %
1Years	Revenue: 229.23 B	RevGrowth: 6.3 %	NetIncome: 48.35 B	NetIGrowth: 5.83 %
0Years	Revenue: 215.64 B	RevGrowth: 0.0 %	NetIncome: 45.69 B	NetIGrowth: 0.0 %



Back Data provided on this site is for education purposes only. No offer or guarantee of any financial outcome is being made of any kind. By using this site you agree to using the data as-is with no warranty of any kind.

<https://github.com/jouellnyc/DockerStocksWeb>

Just keep thinking Next Step

- Detour - learn MongoDB / PyMongo
- Detour - learn WSGI / Gunicorn
- Detour - learn CSS/ NginX / Flask / Jinja
- Tried IEX/AV/Other APIs
- Optimized too early - MemCached
- *** **Minimal effective dose!** ***

Outline

- Mindset and Getting Started
- Work product - working with a goal
- Initial concepts to fast track
- My project - how it evolved
- **Resources - books/videos/free info**

Get Feedback

- Try to Get your Code Reviewed
- Isolated journey feels more real
- More difficult to kid yourself :)
- Learn and Listen

Give back - Seek opportunities

“How can I help you?”

Ask a Dev coworker, your manager,
someone at PyCon or a local meetup.

Start small, show value. Rinse. Repeat.

Get Free Lessons

Python Trainers, Core Developers tweet, share videos and send newsletters for free! (Some courses for pay too).

- <https://twitter.com/reuvenmlerner> - Python Trainer
- <https://twitter.com/dabeaz> - Author of Python Books
- <https://twitter.com/dontusethiscode> - Lots of Pycon talks
- <https://twitter.com/raymondh> - Core Developer
- <https://twitter.com/treyhunner> - Python Trainer

- Youtube for “PyCon” – <http://youtube.com>

Resources: Newsletters

- <https://lerner.co.il/newsletter/>
- <https://powerfulpython.com/newsletter/>
- <https://realpython.com/newsletter/>
- <https://pycoders.com/>

Resources: Challenges

- WPE - <https://store.lerner.co.il/wpe> (pay)
- Python Workout: 50 ten-min ex. (book)
- 100 Skills to Better Python (book)
- <https://projecteuler.net/>
- <https://pybit.es/>

Resources: Articles

Exceptions

<https://docs.python.org/3/library/exceptions.html>

Logging

<https://www.loggly.com/blog/exceptional-logging-of-exceptions-in-python/>

Documenting

<https://datacamp.com/community/tutorials/docstrings-python>

Decorators

<https://realpython.com/primer-on-python-decorators>

Resources: Books

- Python Crash Course - Eric Matthes **
- Beyond the Basic Stuff with Python - Al Sweigart
- 90 Specific Ways to Write Better Python- Brett Slatkin
- Fluent Python - Luciano Ramalho
- <https://www.humblebundle.com/> - (various combos)

** True beginners: start here

Resources: Videos

- Journey Over the Intermediate Gap - Sarah Packman
<http://y2u.be/49CIIu1XkIE>
- Goodbye Print, Hello Debugger! - Nina Zakharenko
<http://y2u.be/5AYIe-3cD-s>
- Practical decorators - Reuven M. Lerner
<https://youtu.be/MjHpMClvwsY>
- How to write a function - Jack Diederich
<http://y2u.be/rrBJVMyD-Gs>

Summary

- Find your why/motivation
- Put your Work Product out there
- Get Free Python info from experts
- Offer to help and Get Involved
- Minimal Effective Dose / Next Steps

<https://github.com/jouellnyc/PyConZA>
jouellnyc@gmail.com