lab?

i make.

Welcome to the DDO!

- Historical property – keep it nice
- Fire exit – look for exit sign in hallway
- Washrooms:
  - Men's downstairs
  - Women's on second floor
  - Can be treated as unisex. Call it out.
- Water has tested potable
  - Help yourself to fridge in corner
- All volunteer – so keep it clean!

About ylab

- Maker space – all volunteer
  - We host A.I.North on meetup.com
- Partnership with Richmond Hill
  - We rent this space.
- Open Mon, Tue, Thu evenings for members
- Non-members: classes, open houses
- Laser cutters, 3D printers, electronics, software, radio...

upcoming events

http://ylab.ca

Meetup: ylab maker space

Meetup: A.I. North

agenda

- Housekeeping – 8:30 bio break
- brief intro – python history/background
- hands-on intro
- 8:30 bio break/quick tour if interested
- code away – with our mentors' help
  - challenges – database, web
  - suggest something you want to do

Talk about us

- Twitter: @ylab_maker #pythonconstricted
- Facebook
-
- Feedback most appreciated on meetup group

python

- Released 2000 by Guido van Rossum
- Open source
- Most searched language (more people learning)
  - Now 3rd most popular (Tiobe Index, Sept 2018)
- Interpreted – but used in major applications
  - Tools to get compiled speed when required
- Front end to math/science libraries and A.I. engines
- Major part of Linux/cloud administration tool development
- Breaks a lot of unwritten rules

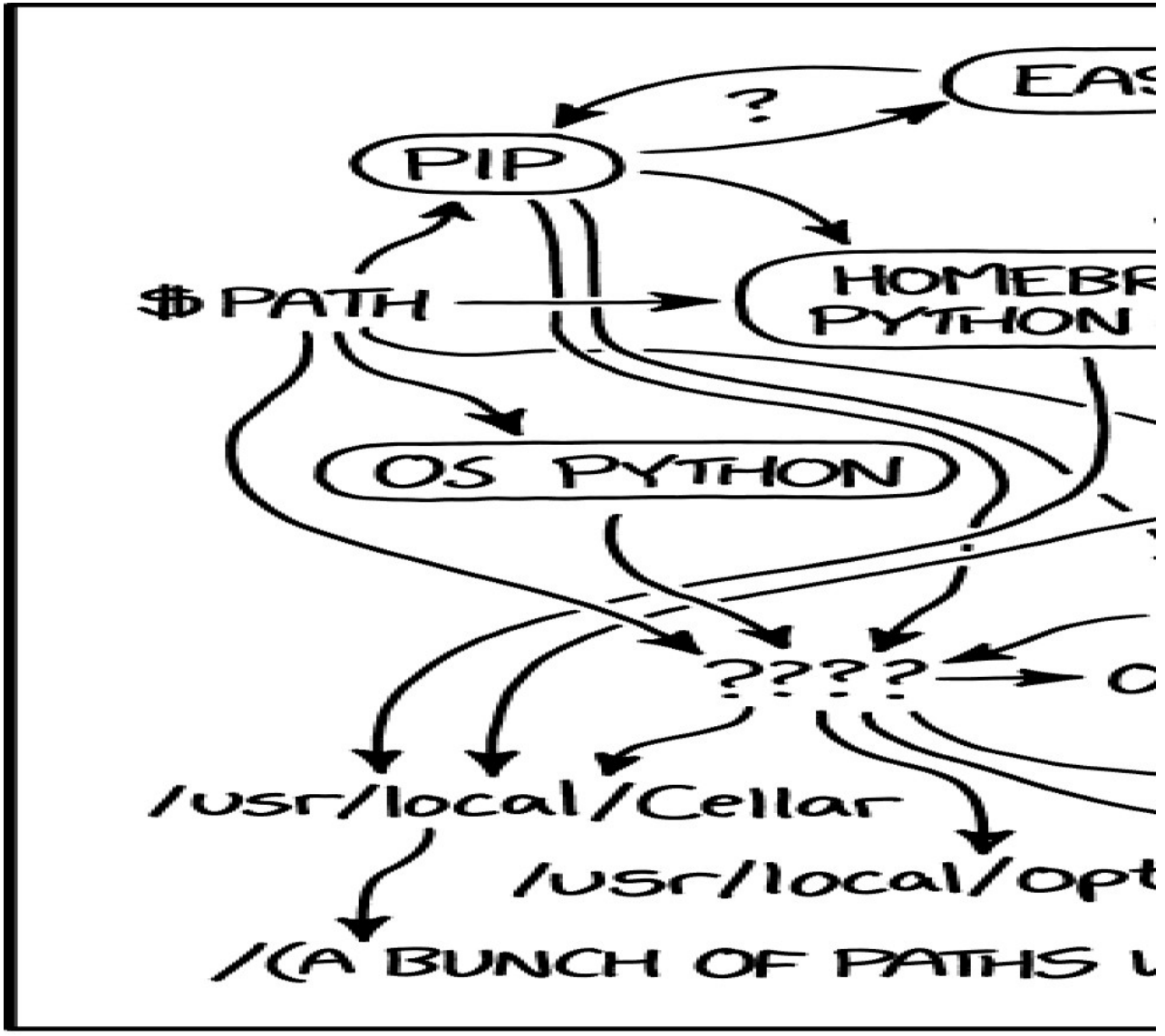
Version 3.X

- Breaks some compatibility with 2.X
  - Lost of complaining
  - But changes make sense – improved consistency
  - Transition not that bad (team doing it at our office)
  - To help with transition, v. 2.7 works with 2.X and 3.X code
  - Multiple versions can live on your systems
- Continual advancements for threads, parallel processing, async I/O

Suggested reading

- Eric S. Raymond (ESR) – Why Python?
  - Why Python? https://www.linuxjournal.com/article/3882
  - 2:1 reduction in code volume from C; comments on GO, C++ and OO
    - http://esr.ibiblio.org/?p=7724
  - And of course: http://www.catb.org/~esr/writings/cathedral-bazaar/
- PJ Plauger: Object oriented – or just encapsulate ?
  - Column 21 in http://seriouscomputerist.atariverse.com/media/pdf/book/Programming%20on%20Purpose.pdf
- Python tutorials and documentation phenomenal. Just search.
- Google what you're trying to do
  - 90% of it you will find on Stack Exchange
  - And please read past the first entry.

Which Python?



Which Python?
Most experimented language

- CPython is the default (written in C). The Reference.
- Jython: written in Java
  - generates JVM code
  - Can use Java libraries
- IronPython: Targets .NET; import C# libraries; written in C#
  - But you can use Python.NET in Cpython
- Cython: generates C code for compiled libraries.
  - Scientific/high performance computing
- PyPy: “If Python is so great, why is it written in C?”
  - So they built a JIT compiler using Rpython (Restricted Python)
- Brython: Python in browser (translates to Javascript)
- And more at https://wiki.python.org/moin/PythonImplementations

Portability and Style...

- Works great between Linux and Windows and Mac and...
  - Be sure to use language features for things like directory separators (/ vs \)
- Style: Just like Unix: not “can I do it?” but “which way will I do it?”
  - Some claim it tries to force you a certain way... I disagree
    - Things tend to be well thought out
    - Things can be re-thought – Python 2.X to Python 3.X
  - Old-guy procedural fine
  - Object oriented fine
- Throws out a lot of old conventions
  - “Interpreted is too slow”. We have ***massive*** speed and memory available
    - Google, AutoCad and others beg to differ
    - Only hard, core stuff needs speed (OS/hardware; OLTP; scientific routines)
    - Flexibility and speed of development more important
  - Nested structures simplified (you’ll see)

GUI Development

Local apps: lots of portable options

- TkInter (Tk/TCL) for simpler stuff
- WxPython for full native (Windows/Unix/Mac)
    - ... and all the complexity that can imply
- Kivy – above, plus IOS, Android, Raspberry Pi

If not for phone app, use a web platform

- Many, many options.
- HTML tools offer greatest flexibility, least pain.

GUI development - Web

If not building IOS/Android apps, probably best bet

- Flexibility, variety, behaviour of web tools
- **Micro frameworks:** All in one including web server
- **Full stack** for Apache, nginx, etc
- **Asynchronous frameworks**
  - Tens of thousands of non-blocking connections
- https://hackernoon.com/top-10-python-web-frameworks-to-learn-in-2018-b2ebab969d1a

IDE for the class

- For today's class: Python default IDLE editor
- After class... You have your choice of IDE for Python
  - Supported in Eclipse, EMACS and others
  - https://www.linuxlinks.com/9-best-free-python-integrated-development-environments/
- **Slack** support for code snippets – structure and execution
- Pull in test code - emailed, or http://ylab.ca/python
- START IDLE NOW
  - Editor
  - Execution window

Basic data structures

NO DECLARATIONS REQUIRED. JUST DO IT.

- Tuple: immutable t=(1, “two”, 3)
- List: fungible l=[1,”two”,3]
- Dictionary: superset of JSON structures
- d={“first”: 1, “second”:2, “third”:3}
- len(t)
- Reference item: t[0] t[1:2] or [1:] or [-2] (from right)
- Tuples: + and * operators OK
- Each can be embedded in the other : t[0][1][3]
- Number: xint=1 xlong=1L xreal=1.0 xcomplex=1.0j
- String uses same reference []
- String split: slist = “one two three”.split(‘ ‘)

Code File Structure

Everything is a .py file – main or library

- No .h, no .lib
- # For comments
- Functions run when called
- Code outside of functions run when library called
- Convention: __main__, main()
  - Standard part of .py programs
  - To only run if this is the program run
  - Awesome place to leave test code for the library

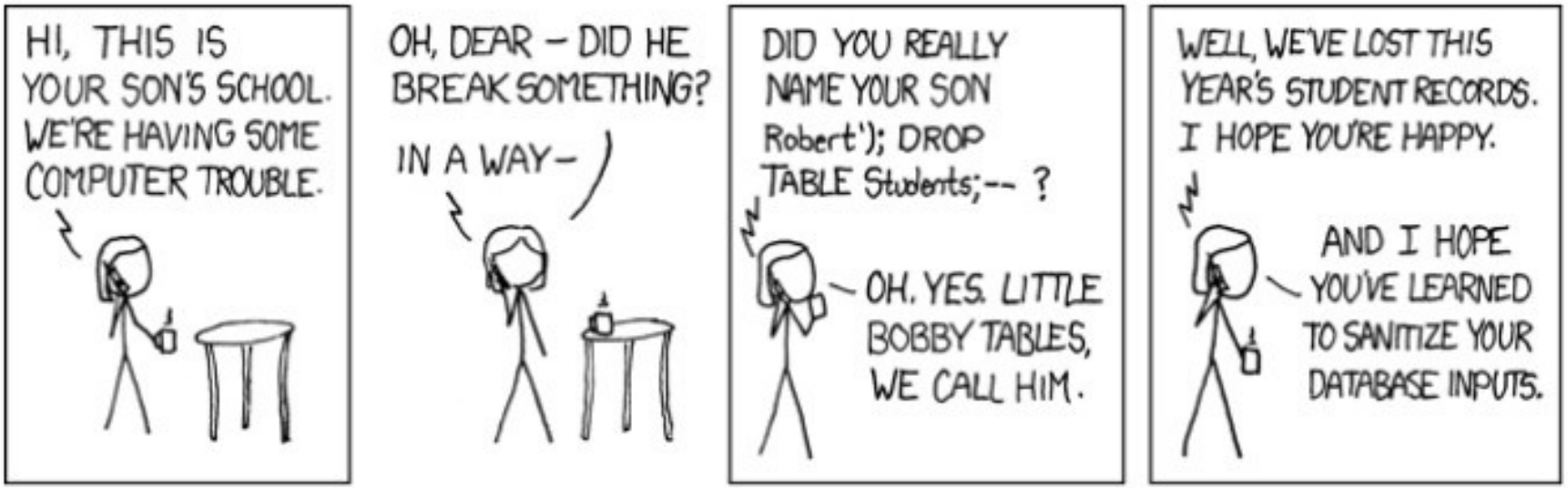
Typical program

```
#!/usr/bin/python
import sys    #these are the libraries
def addtwo(a,b): #This is a function
    return(a+b)      # indent mandatory. When indent
                     # ends, function ends
def main(argv): # By convention only... main function
    x=addtwo(10,20)
    print(“x is:”, x)

if __name__ == “main”:
    print(‘This is the program that was run from command line’)
else:
    print(‘\n\n*** THIS MODULE IS AN IMPORT***\n\n’)
```

eval – the danger

"Exploits of a Mom" (XKCD #327)



Indent defines loops, functions

- No more BEGIN/END, { }, etc.
- Simple indent – you chose number of spaces
  - Also tabs... but don’t. 4 spaces is default
  - Set tab in editor to generate spaces, not \t
- Function:
  -

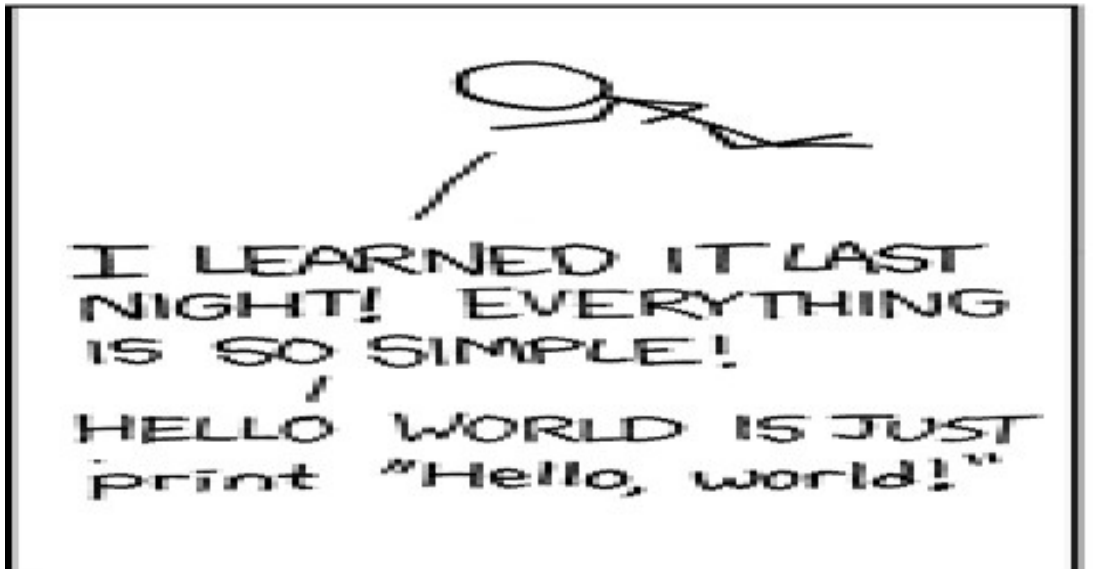
Basic iteration

- for item in list_structure: #iterates through values in tuple, list, dictionary
- for x in range(5): # values 0 to 4
- for x in range (3,6): #values 3,4,5,6
- while <condition>:
-
- Both have **break** and **continue**
- Both have **else** (when loop exhausted or terminated – but not if break or continue executed)
- If <condition>:
- elif <condition>: # **elif** and **else** at same indent level as **if**
- else <condition>:

Basic file operations

- f.open
- f.read -> the whole file, binary or not
- f.readline -> a single line
- f.readlines -> all lines into a list!
- f.seek: re-position the file pointer
- f.close: tidy up after yourself
- f.write: same deal...
- BIG FILE OR SMALL FILE?
  - Efficiency differences... and we have lots of RAM.
  - C really designed before RAM=64K, floppy =140K, hard disk = $$$$

Python Libraries



Library: Web Access

Library: Postgres SQL

PyCon Canada Nov 10-13

https://2018.pycon.ca/

Downtown near Eaton Centre

Conference Nov 10-11

Code Sprints Nov 12-13

Registration opened today!

Ask our volunteers about it