PYTHON, IPYTHON, & JUPYTER

9.4.2018

Synapse

Neuroscience Organization

Speakers · Socials · Free Food

SEA 3.250 every Monday

Socials at 5:30 p.m.

Meetings at 6:00 p.m.

utsynapse.wixsite.com/synapse

facebook.com/groups/synapseUTaustin

Guido van Rossum

* programming language

* named after monty python

* interpreted, not compiled

  * you can type programs directly into the python interpreter

PYTHON DEMO

* `python` - *start the interpreter*

* `print("blah")` - *printing to screen*

* `5 + 5` - *basic math*

* `stuff = 12` - *variables*

* `stuff = [1, 2*3, 14, -5]` - *lists*

PYTHON DEMO

* breakfast = {‘spam’: 2, ‘eggs’: 1, ‘sausage’: 1} *- dictionary*

* breakfast = dict(spam=2, eggs=1, sausage=1) *- also dictionary*

PYTHON DEMO

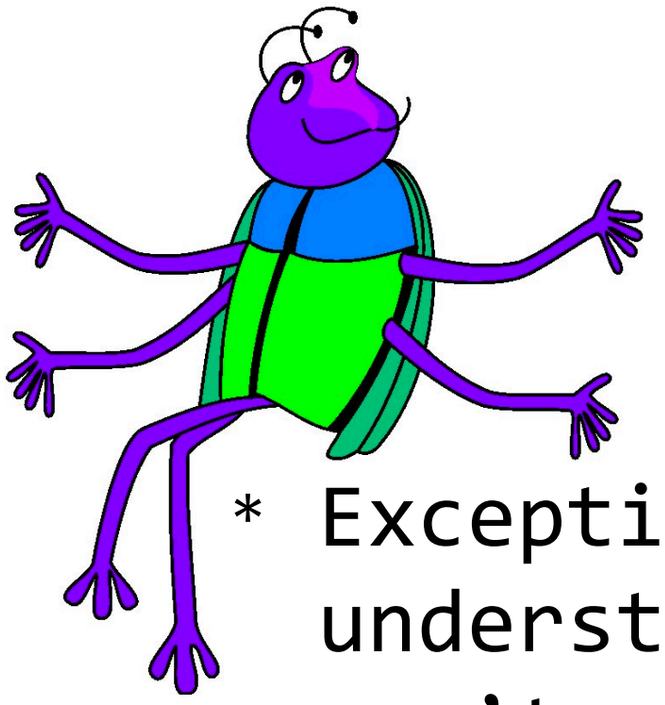
* *write code into* myfile.py

* *run the file using* python myfile.py

* import myfile

PYTHON DEMO

* `def my_function(a, b): …` - *functions*
  * *whitespace delimited!*
* `if: … else: …` *and* `elif: …`
* `for x in y:` - *for loop*



EXCEPTIONS

* Exceptions happen when python can’t understand what you did, or otherwise can’t continue running

* SyntaxError - *you straight messed up*

* NameError - *misspelled variable name?*

* KeyError - *dict doesn’t have that key*

TRACEBACKS

* When an exception occurs, python helpfully tells you exactly where and why in the *traceback*

* The traceback also tells you what function called that one (and where), and what function called that one, and so on

* Learning to read tracebacks is INSANELY USEFUL. YOU WILL DO THIS ALL THE TIME. GET GOOD

IP[y]: IPython
Interactive Computing

Fernando Perez

* Different (much, much better) interpreter for python

* When would you use python over ipython? Pretty much never!

IPYTHON DEMO

* Docstrings
  * (Hopefully) every function contains a docstring that describes what the function does
    * function? - *show docstring in ipython*
* Source code
  * function?? - *show code of function in ipython*
* Tab completion

JUPYTER

Damian Avila

Anaconda, Inc.
@damianavila on GitHub

Matthias Bussonnier

UC Berkeley
@carreau on GitHub

Sylvain Corlay

QuantStack
@sylvaincorlay on GitHub

Brian Granger

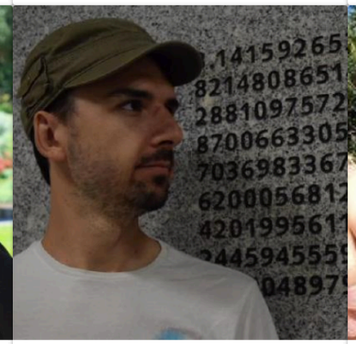
Cal Poly, San Luis Obispo
@ellisonbg on GitHub

Jason Grout

Bloomberg
@jasongrout on GitHub

Jessica Hamrick

DeepMind
@jhamrick on GitHub



Paul Ivanov

Bloomberg
@ivanov on GitHub

Kyle Kelley

Netflix
@rgbkrk on GitHub

Thomas Kluyver

University of Southampton
@takluyver on GitHub

Peter Parente

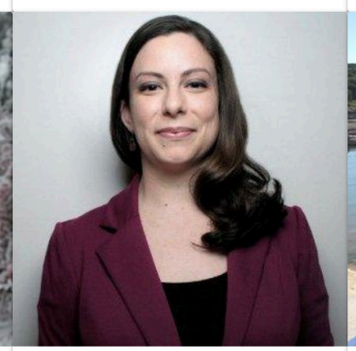
Valassis Digital
@parente on GitHub

Fernando Perez

UC Berkeley
@fperez on GitHub

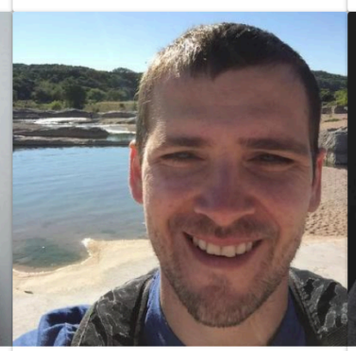
Min Ragan-Kelley

Simula Research Lab
@minrk on GitHub



Ana Ruvalcaba

Cal Poly, San Luis Obispo
@ruv7 on GitHub



Steven Silvester

JPMorgan Chase
@blink1073 on GitHub

Carol Willing

Cal Poly
@willingc on GitHub

JUPYTER

* Run python “notebooks” in a web browser
  * Code lives in *cells*
* Run other stuff too! (Julia, R)

JUPYTER DEMO

READING

* By next Friday (9/14), please read:

* *Inferential Thinking* Chapter 3.1–3.3 (if you read 3.4 I will be disappoint)

  * https://www.inferentialthinking.com/chapters/03/programming-in-python

* *Python Data Science Handbook* Chapter 1

  * https://jakevdp.github.io/PythonDataScienceHandbook/

ASSIGNMENT

* Download and install Anaconda for Python 3.6 (anaconda is a distribution of python that includes a lot of libraries that we will use):

https://www.anaconda.com/download/