INTRODUCTION TO PYTHON

BRIAN HAGUE, WDI INSTRUCTOR
GENERAL ASSEMBLY

Brian Hague

- WDI Instructor for General Assembly
- Background
  - Web Developer at IMS Inc.
  - Software Development Engineer at BudgetMailboxes.com

INTRODUCTIONS

Agenda

- Why programming?
- Why Python?
- Python Overview and Setup
- Python Basics
- Build programs to analyze data

WHY PROGRAMMING?

Why Programming?

- Solve Problems
  - Recommended Products
  - Optimal Road Trip
- Essential Skill of the 21st Century
  - Everyone from health professionals to basketball players can learn to code.

WHY PYTHON?

Why Python?

- Batteries Included
- Simple, clean, easy-to-learn
- Many uses
  - the data science language
  - web applications (Django, Flask)
- Free and open source

```
def sum(a, b):
    return a + b

print sum(1, 2)
```

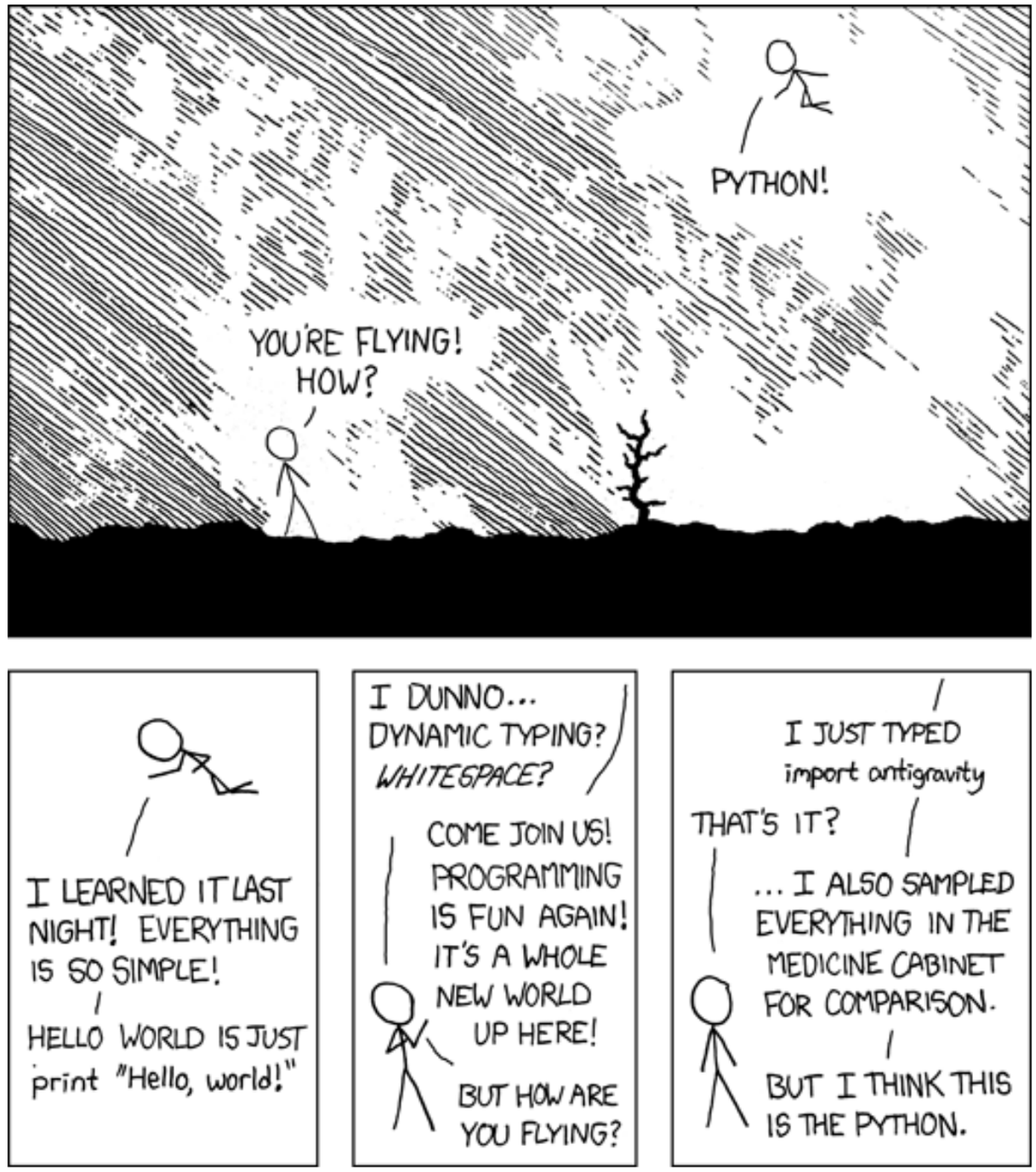
PYTHON OVERVIEW

- Guido van Rossum
  - Benevolent Dictator for Life

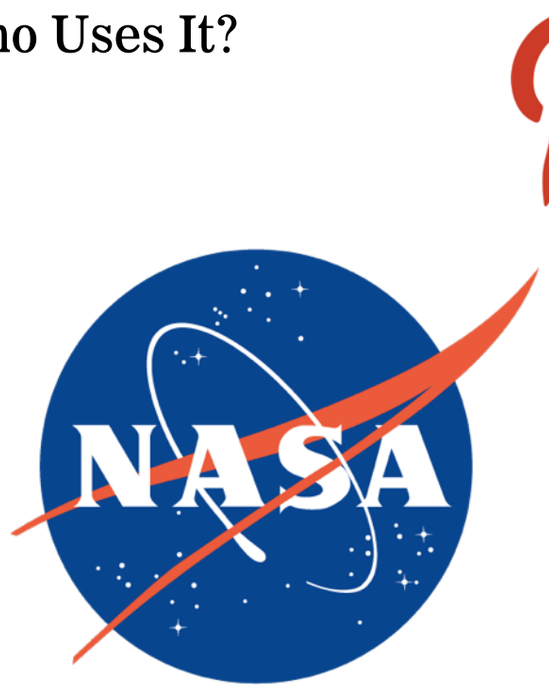
Image source: Wikipedia

What is it like?

- High level
  - Interpreted, not compiled
- Whitespace dependent
  - Gotta indent!
- Dynamic
  - Change types on the fly



Who Uses It?



Pinterest

Google

YAHOO!

reddit

Dropbox

…a lot of companies

PYTHON SETUP

Install Python and Sublime Text

- We'll be using Python 2.7.10
  - Anaconda, an environment manager for Python
- Run Python
  - Launch the iPython QtConsole (run system commands using !)
- Install Atom (text editor)
  - Others include Sublime, Visual Studio Code

Your First Program

- Traditionally, the first program in a new language is Hello World

```
print “Hello, world!”
```

- Interactive Mode: type and execute immediately

- Run a file: Save code with the `.py` extension

  - `run hello_world.py`

FIRST PROGRAM = ✓

Installing Packages

- Usually, we'll use pip (Pip installs Packages)

```
pip install requests
```

- Packages give us more functionality without having to write it ourselves.

- Anaconda gives us many packages by default, like `numpy` and `requests`

Variables

```
a = 4

b = ‘something’

my_variable = True

does_exist = None
```

Basic Types (strings, integers, floats, booleans, None)

```
a_string = ‘I am a single-quoted string’

another_string = “I am a double-quoted string”

an_integer = 4

a_float = 5.2

a_none_value = None
```

Convert Types

```
str(4)                          >>> ‘4’

str(5.0)                        >>> ‘5.0’

int(‘7’)                        >>> 7

float(‘7.0’)                    >>> 7.0
```

Operators (+, -, *, /, **, %)

```
5 + 5                          >>> 10

‘something’ + ‘neat’    >>> ‘somethingneat’

4 ** 2                         >>> 16

4.0 - 2.7                      >>> 1.29999 ~ 1.3

5 % 2                          >>> 1
```

Comparisons (<, >, <=, >=, ==, !=)

```
5 > 4                          >>> True

5 == 2                         >>> False

6 >= 6                         >>> True

5 != 5                         >>> False
```

Boolean Operators (and, or, not)

```
5 > 3 and 7 == 7          >>> True

not 5 == 5                >>> False

5 == 5                    >>> True

5 > 6 or 5 > 1            >>> True
```

Exercise

Create a file called hello_name.py and do the following:

1. Print your name.
2. Print your age.
3. Print how many days old you are.
4. Print if your age is divisible by 4.

Comments

```
# this is a Python comment
```

```
# this is a group of multiple

# Python comments
```

IO (input and output)

```
my_name = raw_input('What is your name: ')
print my_name
```

```
>>> What is your name: Brian
>>> Brian
```

format

```
a_string = '{} is interesting'.format('Programming')
print a_string
```

```
>>> Programming is interesting
```

Conditional statements (if, else, elsif)

```
is_running = True

if is_running:

  print “It’s running!”

else:

  print “It’s not running!”
```

Loops (while, for)

```
counter = 0

while counter < 5:

   print “Loop has run”

   counter = counter + 1
```

```
stuff = [‘jar’, ‘dirt’]

for item in stuff:

   print item
```

Exercise

Create a program called letters.py that prints the number of letters in a word.

1. Ask the user to input a word.
2. Print out the length of the word.
3. Repeat until the user types “EXIT”

Lists

```
my_list = [‘apple’, 4, True, 3.0, ‘nice’]
```

Dictionaries

```
my_dictionary = {
   ‘guitar’: ‘A music instrument’,
   ‘age’: 100 }
```

Exercise

Create a program that prints the frequency of letters in a sentence.

1. Ask the user to input a sentence.
2. Print out each letter and the number of times each letter occurs.
3. Repeat until the user types “EXIT”

DATA ANALYSIS

Data Analysis

- Collect Data (manually or by finding a dataset)
- Clean Data
- Analyze Data
- Reflect
- Report

Collect Data

- Surveys
- APIs (talk to a program)
- Data Scraping

Clean Data

- Are there gaps in the data?
- Is each data point valid?
- Are there outliers that skew the results?

Analyze Data

- You have the data, now what do you want from it?
  - Sums, totals, averages, median, ranges
- How do you get those results?
  - Python! Or some other language
  - Third party libraries (`numpy`)
  - API calls (outside algorithms and functions)

Reflect and Report

- Reflect
  - How are these results accurate? Inaccurate?
  - Can we tweak our methods, explore alternatives?
- Report
  - Present results and conclusions.

Exercise

- What are the most common words in articles about politics?
  - Collect data
  - Clean
  - Analyze
  - Reflect
  - Report

Exercise

- How do people feel about the city of Seattle?
  - Collect data
  - Clean
  - Analyze
  - Reflect
  - Report

HISTORICAL ANALYSIS

Word Counts

- Analyze Titanic passengers
  - Collect (import data using `csv`)
  - Clean (run counts, filter out any anomalies)
  - Analyze (get averages, modes, other data)
  - Reflect (may not be necessary for this)
  - Report (display results)

SENTIMENT ANALYSIS

Word Counts

- What's sentiment?
  - Attitude, feeling, emotion
- How do we measure it?
  - Natural language processing (NLP)
  - Machine learning

Word Counts

- Analyze tweets about pizza and the Seahawks
  - Collect (import sentiment data using `csv`)
  - Clean
  - Analyze
  - Reflect
  - Report

RESOURCES

Python

- http://learnpythonthehardway.org/book/
- https://docs.python.org/2/
- https://docs.python.org/3/
- http://teamtreehouse.com/

Data Science and Analytics

- http://www.clips.ua.ac.be/pattern
- http://www.numpy.org/
- http://www.data.gov/
- https://www.udacity.com
- General Assembly

Q&A

THANK YOU!

- brian.hague@generalassemb.ly
- linkedin.com/in/brianahague
- brianhague.me