Introduction to Python 3

Chang Y. Chung

May, 2015
Shh

Python is ...
Shh

- Python is . . .
- slow.
Python is slow

► A tight loop like below runs 10 to 100 (or more) times slower than C or java.

```python
1 total = 0
2 for i in range(1000):
3    for j in range(1000):
4        total += i           # how many times this statement runs?
5
6 print total
7 # 4999500000000
```
Python is slow

- A tight loop like below runs 10 to 100 (or more) times slower than C or Java.

```python
total = 0
for i in range(1000):
    for j in range(1000):
        total += i  # how many times this statement runs?

print total
# 499950000000
```

- Although you can re-write the above and make it run almost, but not quite, as fast.

```python
print sum([1000 * i for i in xrange(1000)])
# 499950000000
```
Why is Python slow

- Interpreted, not compiled.
- Almost no automatic optimization.
- High-level, versatile programming constructs tend to be larger, more complicated, and slower.
- A simple piece of code may have a huge performance implication. E.g. `range(1000)` creates and returns a 1000-element list every time it is called.
Why Python is not slow

- Faster programming constructs (e.g., `xrange()` vs. `range()`, comprehension vs. `for` loop)
- Modules written in C (e.g., cPickle vs. pickle)
- NumPy and SciPy for scientific computation.
- Python/C API (http://docs.python.org/2/c-api)
- Cython (http://cython.org) takes Python code and generates efficient C code.
- PyPy Just-In-Time (JIT) compiler. (http://pypy.org)
Implementations

- The reference implementation (in C) is called CPython, which Guido van Rossum authored, starting in 1989

- Guido is also known as Benevolent Dictator For Life (BDFL. See http://tinyurl.com/5pg99q)
Implementations (cont.)

- There are other implementations as well.
- IronPython (.NET CLR http://ironpython.net)
- Jython (Java VM http://www.jython.org/)
- pyjs (JavaScript http://pyjs.org/)
- Skulpt (web browser http://www.skulupt.org)
- CodeSkulptor (web browser http://www.codeskulptor.org)
Python 2 or 3?

- Python 3.0 (2008) broke backward compatibility.
  - Can’t use 2 modules in 3 and vice versa.
- "2 is legacy, 3 is the present and future." (http://tinyurl.com/omgx9tk)
  - 3.4 is expected in early 2014.
  - 2.0 was released in 2000.
  - 2.7 (2010) will be the last 2.x branch.
- Many of 3’s major futures have been backported to 2.6 and 2.7, but not all.
- Other implementations in general still lack support for Python 3.
Editors and IDE’s

- EMACS comes with python.el (24.2 and up) and python-mode.el (newer). See (http://tinyurl.com/y67za8d)
- VIM configuration links at http://tinyurl.com/apx3avc
- IDLE (http://tinyurl.com/c7j2k3x)
- (Semi-) commercial editors, e.g., Komodo, PyCharm, Sublime, ...
- IPython (http://ipython.org) and IPython notebook.
- And many others. See http://tinyurl.com/leqyjw7.
IPython and IPython Notebook

- A comprehensive environment for interactive and exploratory computing.
- One of the six core packages of SciPy stack.
PyPI and pip

- Python Package Index (PyPI) is the repository of software for Python at http://pypi.python.org/pypi.
- As of a day in Jan 2014, it has about 38,800 packages.
- Python Indexing Project (pip) (http://www.pip-installer.org) is the standard tool for installing packages (or modules) from PyPI.
- Some examples of using pip. At the shell prompt:

  1. $ pip
  2. $ pip list
  3. $ pip install SomePackage
  4. $ pip install --user SomePackage
  5. $ pip install --upgrade SomePackage
  6. $ pip uninstall

- Once a package is successfully installed, then you can import the module within your script.
Installing SciPy Stack

- It is possible to install all the packages one by one (and all the dependencies). It could turn out to be tricky.
- An alternative is to download and install free or commercial distributions. Some names are: Anaconda, Enthought Canopy, Python(x,y), WinPython, ...
- Check out Wakari.IO (https://www.wakari.io) for playing with SciPy stack on the cloud, without local installation.
Choose the best one that fits each description:

1. Standard module supporting object (de-)serialization, which is written in C.
2. Compiler that turns Python source into efficient C code.
5. Provides a rich architecture for interactive (scientific) computing. Version 1.0 was released in 2013.
NumPy

- Provides the `ndarray` object.
- `ndarray` implements an efficient homogeneous multidimensional array.
- Element-wise and vectorized matrix operations are provided.
- Lots of modules use / built on NumPy.
- Documentation at http://docs.scipy.org/doc.
Scipy

- Collection of mathematical algorithms and utility functions built on NumPy.
- Organized into subpackages: cluster, constants, fftpack, integrate, interpolate, io, linalg (linear algebra), ndimage (N-dimentional image processing), odr (orthogonal distance regression), optimize, signal (signal processing), sparse (sparce matrices), spatial, special (functions), stats, weave (C/C++ integration)
- Documentation at http://docs.scipy.org/doc
Matplotlib

- Provides comprehensive 2D and simple 3D plotting.
- Simple plot, Subplots (multiple axes), Histograms, Path, Simple 3D plot (surface, wireframe, scatter, bar), Streamlines (of a vector field), Ellipses, Bar charts, Pie charts, Filled (curves and polygons), Financial charts, Polar plots, ..., including TeX expressions support (internal or external) and Sketch plots (XKCD style)
- Screenshots are (with source code) at http://matplotlib.org/users/screenshots.html.
pandas

- “Python Data Analysis Library” (Release 0.12 as of 2013).
- Series, DataFrame, and Panel objects
- Reading/writing data to and from: CSV, text file, Excel, SQL db, and fast HDF5 (scientific data file formats and libraries developed at NCSA), JSON, HTML Table, STATA.
- Labeling columns, iteration, Hierarchical Indexing, Transformation, Selection, Missing Data, Merge, Grouping (or split-apply-combine), Reshaping (or pivoting), Time Series, I/O tools, R interface (via rpy2).
- Wes McKinney, “10-minute tour of pandas” (http://vimeo.com/59324550) or workshop (http://www.youtube.com/watch?v=MxRMYjXZos)
Learning Resources

► Websites:
  ▶ Google’s Python Class (2 day class materials including video and exercises) https://developers.google.com/edu/python.
Learning Resources

► Three advanced level tutorial videos:
  ▶ technical (old)
    http://www.youtube.com/watch?v=E_kZDvwofHY.
  ▶ idioms (new)
    http://www.youtube.com/watch?v=OSGv2VnC0go.
  ▶ functional style
    http://www.youtube.com/watch?v=Ta1bAMOMFOI.
Learning Resources

Books:

- “6 Free E-Books” mentioned on http://tinyurl.com/m2y9rad.
Learning Resources

▶ Any cool computer language has:
  ▶ Zen (read and memorize!)
    http://www.python.org/dev/peps/pep-0020/
  ▶ Koans (unit testing) http://tinyurl.com/7n6yfvn
  ▶ Challenges (old) http://www.pythonchallenge.com/

▶ Need more challenges?
  ▶ Try the Project Euler http://projecteuler.net
Learning Resources

► MOOC’s using Python extensively:
  ▶ “Introduction to Interactive Programming in Python” (Coursera, http://tinyurl.com/c95qh2q)
  ▶ “Coding the Matrix: Linear Algebra through Computer Science Applications” (Coursera, http://tinyurl.com/awkbdho)
Learning Resources

- **Twitter:**
  - "teaching python 140 character at a time":
    - http://twitter.com/raymondh

- **Gallery**
  - IPython Notebook gallery (including social data):
    - http://tinyurl.com/c5tj9xh