# Setting up a Sane Development Environment

Brian Forst

# Beginners and Intermediates

- Part 1: Bootstrapping
  - Bootstrapping beginners and people new to Python
- Part 2: Packages, Programs, and Dependencies
  - Useful for new people and developers
  - Easier if you have some familiarity with development
  - May even be new to veteran developers

# Notes

- We are trying to do everything in Python 3
  - Actually, Python 3.3 and up
  - Wall of Superpowers: https://python3wos.appspot.com
- Hands – who is developing or deploying on Windows?
  - Install scripts are in our presentation page/repository on GitHub

# Interactive

- Please ask questions if something isn't clear
- I'm happy saying "no"
  - But will have time to sit down with you after ☺

# PART ONE – BOOTSTRAPPING

# So you want to use Python

- Hooray!
- Oh, Python is already on our computer, so easy!
- Let's take a look!

# What Python Are We Using?

- $ python --version
- $ which python
- $ python -c "import os; print os.__file__"
- $ which python3
- $ echo $PATH

# System Python

- Interpreter:
  - /usr/bin/python
  - /System/Library/Frameworks/Python.framework/Versions/2.7/bin/python2.7
- Resources:
  - /System/Library/Frameworks/Python.framework/Versions/2.7/
- **Do NOT change things in the system Python!**
  - Treat these as system resources that are out of your control

# Installing Our Own Version

- Go here: http://python.org/download
- Download Python 3.3.x (for Mac OS X 10.6 and later)
- Mount the installer and install

# Update the Path

- Open “Applications” → “Python 3.3”
- Run “Update Shell Profile.command”

# What Changed?

- $ which python
  - Still points to the system python
- $ which python3
  - This is the one we installed
- $ ls -l /usr/local/bin/python3
  - Located in /Library/Frameworks/Python.framework/Versions/3.3/bin/python3
- $ echo $PATH

# What does Python give us?

- Batteries included
  - http://docs.python.org/3/library/index.html
- Many datatypes, text processing, regular expressions, math utilities, data compression, CSV format, database connectivity, cryptographic functions, logging, cmd-line argument parsing, threads/multiprocessing, networking, audio, images, GUIs, testing/debugging environments, access to the guts of the Python interpreter.

# Sometimes we want more

- Python Package Index (PyPI)/Cheeseshop
  - https://pypi.python.org/pypi
- ~31,000 packages
  - Some of amazing quality and utility
- How do we get packages from it?

# Install Distribute and Pip

- http://pythonhosted.org/distribute
- Modify the steps at the bottom of the page to:
  - $ curl -O http://python-distribute.org/distribute_setup.py
  - $ python3 distribute_setup.py
  - $ easy_install pip

# High Quality Library: Requests

## With Requests (6 lines)

```
import requests

r = requests.get('https://api.github.com', auth=('user', 'pass'))

print r.status_code
print r.headers['content-type']
```

## Without Requests (18 lines)

```
import urllib2

gh_url = 'https://api.github.com'

req = urllib2.Request(gh_url)

password_manager = urllib2.HTTPPasswordMgrWithDefaultRealm()
password_manager.add_password(None, gh_url, 'user', 'pass')

auth_manager = urllib2.HTTPBasicAuthHandler(password_manager)
opener = urllib2.build_opener(auth_manager)

urllib2.install_opener(opener)

handler = urllib2.urlopen(req)

print handler.getcode()
print handler.headers.getheader('content-type')
```

# Get Requests with Pip

- $ pip search requests
- $ pip install requests
- I always “search” before I “install”

# --help is your Friend

- `$ pip [command] --help`
- Commands I use frequently:
  - search: PyPI for packages matching the given pattern
  - install: installs packages and dependencies
  - list: lists installed packages
  - freeze: lists installed packages in the "requirements" format
  - uninstall: uninstalls packages (but not dependencies)
  - show: shows information about the requested package
- http://www.pip-installer.org/en/1.3.1/index.html

So …

# Remember

- Don't mess with the system Python install
- Run with python3
  - It's the future and, rapidly, the present
- If you need code to do something:
  - Look in the standard library
  - Look in PyPI
  - Only after, write code
- Use pip whenever possible
  - Install/Uninstall
  - Upgrade
  - Manage development
- --help is your friend
  - so is help()

Thanks!

Questions?