Charming The Snake
Python For System Admins

Tony Williams
Systems Engineer

Please download https://bit.ly/xw19-python

What We Will Talk About

- Simple types
- Lists
- IPython
- Dictionaries
- Conditionals
- Looping
- Functions
- Standard modules
  - Subprocess
  - os and paths
- Requests
- plistlib

Numbers

The interpreter acts as a simple calculator: you can type an expression in it and it will write the value. Expression syntax is straightforward: the operators +, -, * and / work just like in most other languages (for example, Pascal or C); parentheses (()) can be used for grouping.

For example:

```
>>> 2 + 2
4
>>> 50 - 5 * 6
20
>>> (50 - 5) * 6
270
```

Numbers

```
8 / 5
8 % 5
8 / 5.0
8 / 2.0
round(8/5.0)
int(round(8/5.0))
type(5)
type(5.0)
```

More Numbers

The equal sign (=) is used to assign a value to a variable. Afterwards, no result is displayed before the next interactive prompt.

```
>>> width = 20
>>> height = 5 * 9
>>> width * height
900
```

If a variable is not “defined” (assigned a value), trying to use it will give you an error:

```
>>>
>>> n  # try to access an undefined variable
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'n' is not defined
```

Lists and iteration

```python
a_list = [1, 2, 3, 45]
a_list[0]
a_list[-1]
a_list[1:3]

for item in a_list:
    print(item)

mixed = [4, "Tony's", 3.14, True, 'bees']
for thing in mixed:
    print (thing, type(thing))

show = 'My quote has it\'s text: "strings are good"'
print(show)
show2 = "My quote has it's text: \"strings are good\""
print(show2)
```

Starting IPython

command	description
ipython	run IPython
jupyter qtconsole	runs ipython in QT window
jupyter notebook	runs the IPython notebook server
ipython –help	IPython man page

Dictionaries

```python
staff_ids = {'Alice': 312, 'Ted': 519, 'Carol': 781}

staff_ids['Ted']

for key in staff_ids:
    print(staff_ids[key])

staff_ids['Ted'] = 783

for key in staff_ids:
    print (key, staff_ids[key])

hgttg = {'Life': True, 'Meaning': 42, 'Motto': "Don't Panic"}
print(hgttg["Meaning"])
```

Conditionals

```python
num = 4

if num == 4:
    print("Four")

if num > 4:
    print("Huge")
else:
    print("Not huge")

if num > 3:
    print("Biggest")
elif num > 2:
    print("Big")
else:
    print("Small")
```

More Loops

```
tm = 1
while tm <= 10:
    print(tm)
    tm = tm + 1
```

```
for x in range(1,11):
    print(x)
```

Iterating

```
collection = ['hey', 5, 'd']
for x in collection:
    print(x)
```

```
[print(i) for i in collection]
```

```
list_of_lists = [ [1, 2, 3], [4, 5, 6], [7, 8, 9]]
for list in list_of_lists:
    for x in list:
        print(x)
```

Functions

- Use `def` keyword
- May use `return` keyword

```
def hello_world():
    print("Hello World!")

hello_world()
```

```
def times_two(x):
    return x * 2

times_two(21)
```

Running Commands

Call external system commands

```python
import subprocess

cmd = ['find', '/usr', '-iname', 'zip-*']
subprocess.run(cmd)
ret = subprocess.run(cmd, capture_output=True)

ret.stdout
ret.stderr
ret.returncode
```

More Subprocess

```python
out = subprocess.check_output(['ls'])
out

out = subprocess.check_output(['find', 'shjsd'])

try:
    out = subprocess.check_output(['find', 'shjsd'])
except subprocess.CalledProcessError as CP_err:
    print("Error in find: %d" % CP_err.returncode)
```

Joining Paths

```
import os
plugin_path = os.path.join("/", "Library", "Internet Plug-Ins")
print(plugin_path)
```

Manipulating Paths

```
from os import path
path.basename(plugin_path)
path.dirname(plugin_path)
path.splitext("com.apple.Safari.plist")
```

Tests on Files

```
who

from os.path import *
who

exists(silverlight_plugin_path)
isdir(silverlight_plugin_path)
islink("/etc")
```

glob

```
import glob
ess = glob.glob("/Applications/Utilities/" "S*.app")
print(ess)
```

Requests

```
import requests
url = 'https://icanhazdadjoke.com/'
r = requests(url)
r.status_code
r.text
r.headers
```

```
headers = {'Accept': 'text/plain'}
r = requests.get(url, headers=headers)
r.text
```

JSON

```
headers = {'Accept': 'application/json'}
r = requests.get(url, headers=headers)
r.text
r.json()
r.json()['joke']
```

Queries

```
url = 'https://api.openbrewerydb.org/breweries?by_city=new_york'
r = requests(url)
r.json()
for brewery in r.json():
    print(brewery['name'], "Street: ", brewery['street'])
url = 'https://api.openbrewerydb.org/breweries'
query = {'by_city': 'new_york'}
r = requests.get(url, params=query)
r.json()
r.url
```

Bigger Queries

```
query = {'by_city': 'williamsburg'}
r = requests.get(url, params=query)
len(r.json())
query = {'by_city': 'williamsburg', 'by_state': 'virginia'}
r = requests.get(url, params=query)
len(r.json())
r.url
```

plistlib

```
import plistlib
prefs = {'fname': 'Tony', 'sname': 'Williams'}
fp = open('example.plist', 'wb')
plistlib.dump(prop, fp)
fp.close()
```

```
with open('example.plist', 'rb') as fd:
    pl = plistlib.load(fd)
    print(pl['fname'])

pl['fname'] = 'Anthony'

with open('example.plist', 'rb') as fd:
    plist.dump(pl, fd)
```

XML

```python
auth=('xworld', 'passwd')
base_url=('https://twxworld.jamfcloud.com/JSSResource/')
url = base_url + 'categories'
cat = requests.get(url, auth=auth)
cat.text
cat.xml

import xml.etree.ElementTree as ET
root = ET.fromstring(cat.text)
categories = root.findall('category')
for i in categories:
    print(i.find('id').text, i.find('name').text)
```

PUT and POST With Requests

```
url = base_url + 'categories/id/3'
cat = requests.get(url, auth=auth)
root = ET.fromstring(cat.text)
root.find('name').text
root.find('name').text = "Changed"

new = ET.tostring(root)
ret = requests.put(url, auth=auth, data=new)
ret.status_code
ret.text

root.find('name').text = 'Inserted'
root.find('id').text = '0'
new = ET.tostring(root)
url = base_url + 'categories/id/0'
ret = requests.post(url, auth=auth, data=new)
ret.status_code
ret.text
```

Further Places

- Dive Into Python 3 - A good tutorial for experienced programmers
- A Byte of Python - Good tutorial for beginners
- A Byte of Python PDF - The above as a PDF
- Python for NonProgrammers - An exhaustive list