Python Packages Make Life A Little Easier

Robert A. Petit III
IBS796 Python - Week 3
9/19/16

Python Packages

- Multiple Python modules organized together
- Allows you to import functions
- No need to reinvent the wheel
- Many more positives

Using a Python Package

- You already have!
- You must ‘import’ the package.
  - In R it is ‘library(“the-package-name”)’
- Examples:

```
import os
import random
import sys

sys.argv[1]
random.random()
os.path.basename(‘/path/to/some/file.txt’)
```

Importing Specific Functions

- Sometimes you don’t need the whole package
- Python allows you to import a specific function from a package

```
from os.path import basename
from random import random
from sys import argv

argv[1]
random()
basename('/path/to/some/file.txt')
```

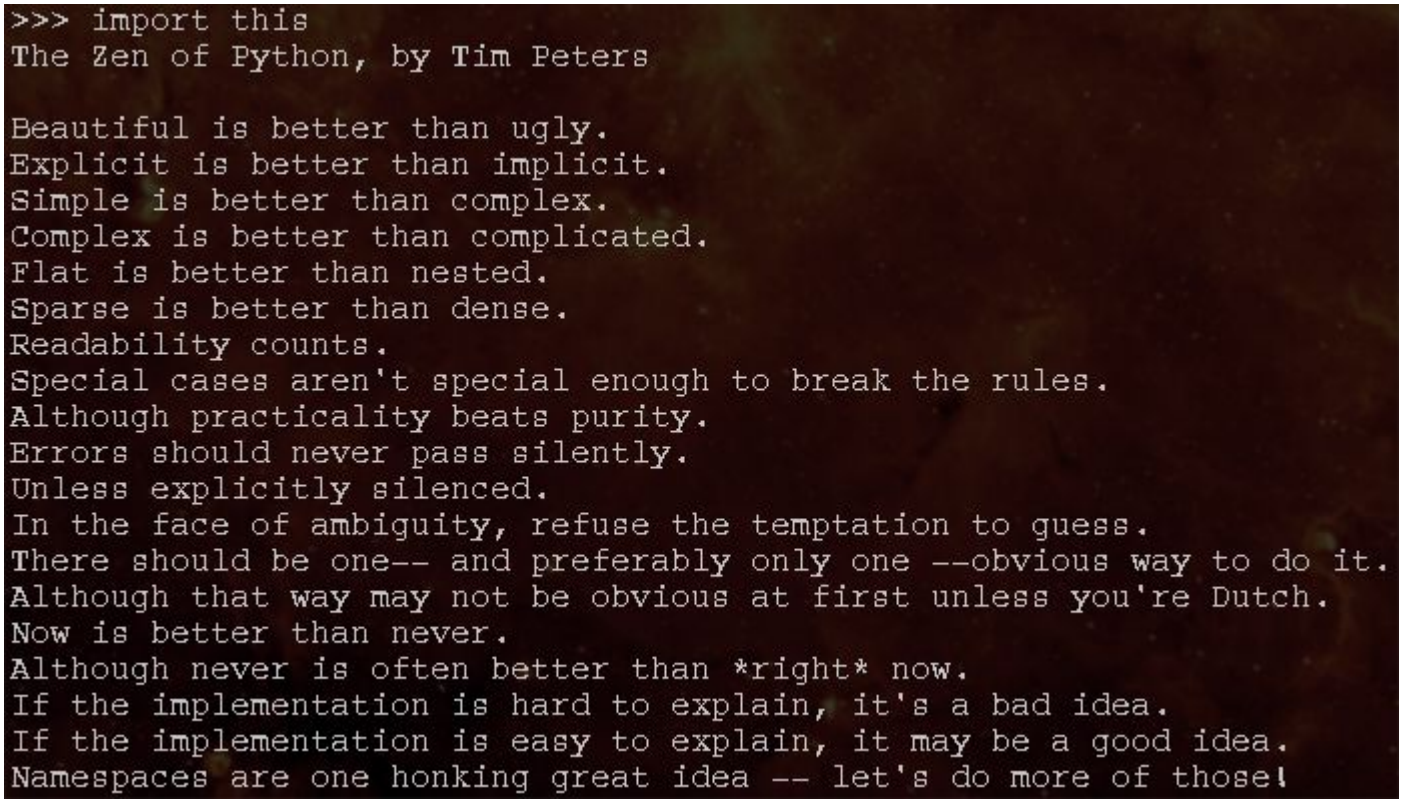
You Can Even Rename Packages

- Sometimes, you don’t want to repeat some long function name over and over
- You can use ‘as’ to rename

```
from os.path import basename as bname
from random import random as rng


rng()
bname('/path/to/some/file.txt')
```

The Zen of Python



Some examples

- “Explicit is better than implicit”

```
from os.path import *
# vs
from os.path import basename, splitext, isdir
# or..
import os
```

- “Readability counts”

```
files = []
if rng() > 0.5:
    files.append(bname(file))

# vs…
if random() > 0.5
    files.append(basename(file))
```

PyPi - The Python Package Index

- https://pypi.python.org/pypi

- A large index of publicly available Python packages
  - Maintains multiple versions of packages

- Allows users to easily download and install Python packages

pip - Automating Package Installation

- pip allows Python packages from PyPi to easily be installed
- If you don't have pip, get it!

```
pip install "package-name"

pip install Django
pip install Django==1.8  # Specific Version
pip install --upgrade Django # Upgrade the package

# Install to User's Home Directory
pip install -U Django

# Uninstall a package
pip uninstall Django

# Learn more!
pip --help
```

Biopython - Python's "Bio" package

- Biopython provides a number of tools for biological data analyses
- Many of the standard formats are easily parsed
- Sequence analysis and manipulation
- Biopython Documentation
- Installation:

```
pip install biopython
```

- Example Usage:

```
from Bio.Seq import Seq
from Bio.SeqUtils import GC

sequence = Seq("ATGCATCGTACGTACG")
print(GC(sequence))
```

Numpy

- The goto statistical/numerical analysis Package
- You will be using this one a lot.
- Numpy Documentation
- Installation:

```
pip install numpy
```

- Example Usage:

```
import numpy as np #<-- very common usage

a = np.array([0,1,2,3,4,5,6,7,8,9])
np.mean(a)
np.std(a)
np.var(a)
```

A number of Packages to investigate!

- SciPy - Scientific Analyses
- Pandas - Data Structures
- Requests - Working with HTTP
- BeautifulSoup - HTML/XML parsing
- Jupyter (Ipython) - Documentation and live notebooks
- Many many many more!

Homework

- If you didn’t finish Week 2’s HW, finish it and be ready to show it off

- Repeat Week 2’s homework except using any available packages.
- One Requirement: use argparse package to take command line arguments
  - Set the default types of the arguments
  - Learn about positional and optional arguments