

Python Workshop

Getting Started with Python

Mike Fienen

U.S. Geological Survey
Wisconsin Water Science Center, Middleton, Wisconsin USA

USGS National Groundwater Workshop, August 2012



- 1 Preliminaries
- 2 Resources
- 3 Modules and Packages
- 4 Scripts
- 5 Writing, Editing, and Running Scripts

What is Python?

- Python is a high-level, general programming language (relying much on C underneath) that is interpreted at run time through a command-line interface or scripting.

What is Python?

- Python is a high-level, general programming language (relying much on C underneath) that is interpreted at run time through a command-line interface or scripting.
- Python is free, open-source, and (generally) platform-independent.

What is Python?

- Python is a high-level, general programming language (relying much on C underneath) that is interpreted at run time through a command-line interface or scripting.
- Python is free, open-source, and (generally) platform-independent.
- Python is built into many other programs and can wrap other languages like Java, C, FORTRAN, etc.

What is Python?

- Python is a high-level, general programming language (relying much on C underneath) that is interpreted at run time through a command-line interface or scripting.
- Python is free, open-source, and (generally) platform-independent.
- Python is built into many other programs and can wrap other languages like Java, C, FORTRAN, etc.
- Python has a massive user community worldwide and growing within USGS.



Guido van Rossum

Python Versions and Packages

- The two main version branches are 2.x and 3.x
- We *strongly* recommend sticking with 2.x for now—Specifically **2.7.2 or 2.7.3**
- To do almost any substantial mathematics or statistics, you also need `numpy`
- Basic 2-D graphics are possible using `matplotlib`
- More advanced statistics are also available using `scipy`
- Distributions already including many packages:
 - ▶ `pythonxy`: for Windows only

<http://pythonxy.com>



- ▶ EPD: Enthought Python Distribution for all Platforms

<http://www.enthought.com>



Expanding Python with Packages

Major capabilities (Numpy, Scipy, etc.) installed from a website using an installer (.msi, .exe, .dmg, .rpm)

For other packages, a couple options...

- Python Package Index <http://pypi.python.org>
 - ▶ For packages within PyPi, can use `easy_install` at command line.
 - ▶ Packages at PyPi are installed using eggs. For more than you ever wanted to know about eggs, see <http://tinyurl.com/4jd5wud>
- Here's a blog post explaining for Win 7 <http://tinyurl.com/7hf9ml6>

Getting Help: Active and Helpful Python Community

- Stackoverflow <http://stackoverflow.com>

Getting Help: Active and Helpful Python Community

- Stackoverflow <http://stackoverflow.com>
- A Useful Glossary <http://docs.python.org/glossary.html#glossary>

Getting Help: Active and Helpful Python Community

- **Stackoverflow** <http://stackoverflow.com>
- **A Useful Glossary** <http://docs.python.org/glossary.html#glossary>
- **Python Official Docs** <http://www.python.org/doc/>
- **Numpy and Scipy Official Docs**
<http://docs.scipy.org/doc/>

Getting Help: Active and Helpful Python Community

- **Stackoverflow** <http://stackoverflow.com>
- **A Useful Glossary** <http://docs.python.org/glossary.html#glossary>
- **Python Official Docs** <http://www.python.org/doc/>
- **Numpy and Scipy Official Docs**
<http://docs.scipy.org/doc/>
- **Numpy and Scipy Cookbook**
<http://www.scipy.org/Cookbook/>

Getting Help: Active and Helpful Python Community

- **Stackoverflow** <http://stackoverflow.com>
- **A Useful Glossary** <http://docs.python.org/glossary.html#glossary>
- **Python Official Docs** <http://www.python.org/doc/>
- **Numpy and Scipy Official Docs**
<http://docs.scipy.org/doc/>
- **Numpy and Scipy Cookbook**
<http://www.scipy.org/Cookbook/>
- **Matplotlib** <http://matplotlib.sourceforge.net/index.html>
- **Matplotlib Gallery (ridiculously cool)** <http://matplotlib.sourceforge.net/gallery.html>

Getting Help: Active and Helpful Python Community

- **Stackoverflow** <http://stackoverflow.com>
- **A Useful Glossary** <http://docs.python.org/glossary.html#glossary>
- **Python Official Docs** <http://www.python.org/doc/>
- **Numpy and Scipy Official Docs**
<http://docs.scipy.org/doc/>
- **Numpy and Scipy Cookbook**
<http://www.scipy.org/Cookbook/>
- **Matplotlib** <http://matplotlib.sourceforge.net/index.html>
- **Matplotlib Gallery (ridiculously cool)** <http://matplotlib.sourceforge.net/gallery.html>
- **If you already know some MATLAB** http://www.scipy.org/NumPy_for_Matlab_Users/

Command Line: Launch and Enter Commands

Type “python” to launch python

```
c:\>python
Enthought Python Distribution (EPD) free version -- www.enthought.com
Version: 7.1-1 (32-bit)
(type 'upgrade' or see www.enthought.com/epd/upgrade to get the full EPD)

Python 2.7.2 |EPD_free 7.1-1 (32-bit)| (default, Jul  3 2011, 15:40:35)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "packages", "demo", "upgrade" or "enthought" for more information.
>>>
>>>print 'hello world'
hello world
```

Command Line: Launch and Enter Commands

Type “python” to launch python

```
c:\>python
Enthought Python Distribution (EPD) free version -- www.enthought.com
Version: 7.1-1 (32-bit)
(type 'upgrade' or see www.enthought.com/epd/upgrade to get the full EPD)

Python 2.7.2 |EPD_free 7.1-1 (32-bit)| (default, Jul  3 2011, 15:40:35)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "packages", "demo", "upgrade" or "enthought" for more information.
>>>
>>>print 'hello world'
hello world
```

All commands that result in output display it on the screen

```
>>> a=['this','is','a','list']
>>> a
['this', 'is', 'a', 'list']
>>> a[0]
'this'
```


Packages, Modules, Namespaces

First, a couple definitions.

Script: A text file containing variables, functions, and definitions that can be loaded for use at command line or in another script (a.k.a. “code”)

Module: A script containing definitions for a specific purpose

Package: A group of modules

Namespace: A base name for a module used to organize and keep track of its provenance.

Packages, Modules, Namespaces at Command Line

There are several ways to import a module or package.

Packages, Modules, Namespaces at Command Line

There are several ways to import a module or package.

- Preserving the namespace as the module name

```
>>>import numpy
```

To use the function `sqrt` must type

```
>>>s9 = numpy.sqrt(9)
>>>s9
3.0
```

Packages, Modules, Namespaces at Command Line

There are several ways to import a module or package.

- Preserving the namespace as the module name

```
>>>import numpy
```

To use the function `sqrt` must type

```
>>>s9 = numpy.sqrt(9)
>>>s9
3.0
```

- You can also provide an alias...

...for the namespace

```
>>>import numpy as np
>>>s9 = np.sqrt(9)
>>>s9
3.0
```

...or for the function

```
>>>from numpy import sqrt as pow_to_minus_pfive
>>>s9 = pow_to_minus_pfive(9)
>>>s9
3.0
```

Packages, Modules, Namespaces at Command Line

There are several ways to import a module or package.

- Preserving the namespace as the module name

```
>>>import numpy
```

To use the function `sqrt` must type

```
>>>s9 = numpy.sqrt(9)
>>>s9
3.0
```

- You can also provide an alias...

...for the namespace

```
>>>import numpy as np
>>>s9 = np.sqrt(9)
>>>s9
3.0
```

...or for the function

```
>>>from numpy import sqrt as pow_to_minus_pfive
>>>s9 = pow_to_minus_pfive(9)
>>>s9
3.0
```

- Or import the function without the namespace (*danger!*)

```
>>>from numpy import *
>>>s9 = sqrt(9)
>>>s9
3.0
```

Module Example

```
import this
import antigravity
```

Module Example

```
import this
import antigravity
```

- These are kind of humorous but (especially) the antigravity example raises an important issue: modules not only contain a listing of code and definitions, but can also *execute*!
- So, use caution and only load modules from trustworthy sources.

Using Scripts

- Scripts allow you to save the logic and definitions of a program much in the way compiled source code does for FORTRAN, C, and other languages.
- The basic idea is to contain all code, in the same way you would type it into an interactive command window, in a file with the extension `.py`
- You can then execute the script by typing
`python <scriptname>.py`
at the command line.

Using Scripts

- Scripts allow you to save the logic and definitions of a program much in the way compiled source code does for FORTRAN, C, and other languages.
- The basic idea is to contain all code, in the same way you would type it into an interactive command window, in a file with the extension `.py`
- You can then execute the script by typing
`python <scriptname>.py`
at the command line.
- Make a textfile called “hw.py” with the following code in it:

```
print ``Hello World!``
```
- Now, type `python hw.py` at the command line.

Composing scripts in a text editor

There are a couple important Python idiosyncrasies that influence text editor choice.

- 1 You don't close loops in python, so indentation is paramount!

Composing scripts in a text editor

There are a couple important Python idiosyncrasies that influence text editor choice.

- 1 You don't close loops in python, so indentation is paramount!

```
for i in stuff:  
    print i
```

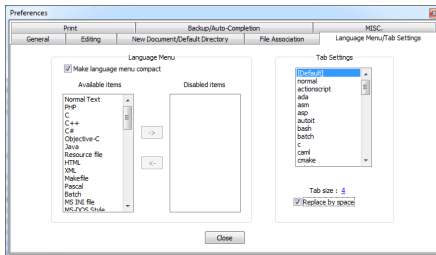
Composing scripts in a text editor

There are a couple important Python idiosyncrasies that influence text editor choice.

- 1 You don't close loops in python, so indentation is paramount!

```
for i in stuff:  
    print i
```

- 2 Tabs aren't consistent from editor to editor so change them to spaces
 - ▶ They must be consistent within a script though.



Composing scripts in a text editor

There are a couple important Python idiosyncrasies that influence text editor choice.

- 1 You don't close loops in python, so indentation is paramount!

```
for i in stuff:  
    print i
```

- 2 Tabs aren't consistent from editor to editor so change them to spaces
 - ▶ They must be consistent within a script though.
- 3 Code highlighting and code folding can be nice
- 4 An IDE lets you run the code in context and sometimes includes autocomplete and other features.

Some Suggested Environments/Editors

This list is neither exhaustive nor fully endorsed!

- Editors

Notepad++, TextPad, TextWrangler, VIM, Emacs,
Ultra Edit, Komodo Edit

- Integrated Development Environments (IDEs)

Wing IDE, Komodo IDE, IDLE,
Python Tools for Visual Studio10, NetBeans, Eclipse

Some Suggested Environments/Editors

This list is neither exhaustive nor fully endorsed!

- Editors

Notepad++, TextPad, TextWrangler, VIM, Emacs,
Ultra Edit, Komodo Edit

- Integrated Development Environments (IDEs)

Wing IDE, Komodo IDE, IDLE,
Python Tools for Visual Studio10, NetBeans, Eclipse

And a brief aside on version control...

- Git and Github.com allow you to keep track of and collaborate on code. Worth a look!

Some Suggested Environments/Editors

This list is neither exhaustive nor fully endorsed!

- Editors

Notepad++, TextPad, TextWrangler, VIM, Emacs,
Ultra Edit, Komodo Edit

- Integrated Development Environments (IDEs)

Wing IDE, Komodo IDE, IDLE,
Python Tools for Visual Studio10, NetBeans, Eclipse

And a brief aside on version control...

- Git and Github.com allow you to keep track of and collaborate on code. Worth a look!

OK, that was kind of an endorsement