

The Python Programming Language

Neil Blakey-Milner

`nbm@mithrandr.moria.org`

`http://mithrandr.moria.org/`

Outline

- Why Python?
- Python Syntax
- Standard Library
- Third Party Libraries

Why Python?

How do you evaluate a programming language?

- Simplistic: CPU time, memory usage, lines of code
- Developer time most expensive
- Developer time comes down to
 - ease of use
 - low overheads
 - solid standard and third-party libraries
 - ...

What's so great about Python?

- Python Language
- Standard Library
- Third Party Libraries
- Open Source
- Community

Python Language

- Simple
- Elegant
- Reliable
- Robust

Standard Library

- Batteries Included!
- Protocols: HTTP, FTP, SMTP, POP3, IMAP, NNTP, ...
- Markup support: HTML, SGML, XML
- File Formats: Mailbox, CSV, ZIP, tar
- Servers: HTTP, XMLRPC
- Debugging, logging, profiling, regular expressions

Third Party

- Twisted
 - DNS, FTP, IRC, Jabber, MSN, Oscar, SIP, ...
- Cheetah
 - Powerful templating language
- Webware
 - Servlet-based application server
- Zope
 - Python Application Server

Open Source

- You all know the advantages

Community

- Friendly skilled Open Source community.

Who uses Python?

- Industrial Light and Magic
- Google
- Computer games (EVE, Temple of Elemental Evil, URU Ages of Myst, Ultimate Online 2)
- Thawte
- RedHat anaconda, Gentoo portage
- NASA

Python Syntax

Strings

- “String”
- 'string'
- “””John said, “Foo is a bad dog”, and left.”””
- ""Foo is Peter's Dog""
- “I have %d apples in my %s” % (3, “bag”)

Assignment

- $a = 1$
- $b = a + 5$
- $c = (b * 3) + 2$
- $a, b = 6 + 4, 1 + 3$
- $a, b = b, a$

print and raw_input

- print "Hello"
- print a
- name = raw_input('Give me your name')

Lists/Sequences

- `l = []`
- `l2 = ["a", 2, b * 3]`
- `print l2[0]`
- `l.append(3)`
- `l2.remove(2)`

Dictionaries

- `d = {}`
- `d['hello'] = 'there'`
- `del d['hello']`
- `d.get('foo', 1)`
- `d.has_key('foo')`
- `d.keys()`
- `d.items()`

indentation

- Indentation alone determine which block of instructions a particular line belongs to.
- No visual/semantic conflict
- 'pass' does nothing, creates an empty block.

for

- for fruit in ["apple", "banana", ...]:
 print "%s is a fruit" % (fruit)
- for line in file('/tmp/fstab'):
 print line
- for letter in "Hello there":
 print letter

while

- `x = 1`
 `while x < 10:`
 `print x`
 `x += 1`

if

- `x = 1`
`y = 5`
`if x > y:`
 `print "%d is greater than %d" % (x,`
 `y)`

functions

- ```
def twice(a):
 b = a * 2
 return b
```

# Classes

- `class a:`  
    `pass`
- `class classname (parent1, ...):`  
    `classvar = 1`  
    `def method1(self, arg1, ...):`  
        `return "stuff"`

# Standard Library

- Batteries Included!

# email

```
import smtplib
from email.MIMEText import MIMEText

fp = open('/tmp/message')
Create a text/plain message
msg = MIMEText(fp.read())

me = 'me@my.domain'
you = 'you@your.domain'

msg['Subject'] = 'That stuff you wanted'
msg['From'] = me
msg['To'] = you

s = smtplib.SMTP()
s.connect()
s.sendmail(me, [you], msg.as_string())
s.close()
```

# mailbox

```
import email
import mailbox
```

```
maildir = mailbox.Maildir('Maildir', email.message_from_file)
```

```
for mail in maildir:
 print mail["Date"]
```

# Simple mailing list software

```
import smtplib
import email
import mailbox

maildir = mailbox.Maildir('Maildir', email.message_from_file)

to = ['first1@company1', 'first2@company2', ...]
listprefix = '[ListName] '

s = smtplib.SMTP()
s.connect()
for mail in maildir:
 mail['List-Id'] = foo
 if listprefix not in mail['Subject']:
 mail['Subject'] = listprefix + mail['Subject']
 mail['From'] = 'Original sender hidden <list@list.server>'
 s.sendmail(me, [you], mail.as_string())
s.close()
```

# Third Party Libraries

- Most of them are “Pythonic”

# SQLObject

```
from SQLObject import *

__connection__ = MySQLConnection(db='foo', user='root')

class Subscriber(SQLObject):
 address = StringCol(alternateID=True, length=50)
 name = StringCol(length=50)
 added = DateTimeCol()
 admin = BoolCol(default=False)
 bounces = IntCol(default=0)

Subscriber.createTable()
```

# Cheetah Templates

- Plays in the same area as Smarty, Mason, or Velocity

# Webware

- Servlet-based application server
- Supports auto-reload when modules change.
- Means not having to do work over and over as per PHP model.