python.

RICE UNIVERSITY • COMP 140

AN EXPERIMENTAL TALK
FOR AN EXPERIMENTAL COURSE
AUG 26 & 27, 2008

Daniel Sandler **dsandler@rice**

http://www.cs.rice.edu/~dsandler/python/

1.

2.

3.

4.

1.

A young mind is corrupted.

Me, circa 2000

Be
Be Operating System
Release 1.1d7.1
© Copyright 1996, Be, Inc. All rights reserved.
Be, the Be Logo, BeBox, and Be OS are trademarks of Be, Inc.

(very busy)

Many tasks, many languages

C++	**Scheme**	**perl**
(work)	*(fun)*	*(utility belt)*
OOP, popular	pretty, powerful	useful

Many tasks, many languages

C++	**Scheme**	**perl**
(work)	*(fun)*	*(utility belt)*
OOP, popular	pretty, powerful	useful
painful development cycle	unreadable by others	unreadable by *anyone*

```
print pack"C*",split/\D+/,`echo "16iII*o\U@{$/=$z;[(pop,pop,unpack"H*",<>
)]}\EsMsKsN0[lN*1lK[d2%Sa2/d0<X+d*lMLa^*lN%0]dsXx++lMlN/dsM0<J]dsJxp"|dc`
```

then,
I discovered
a language
called

then,
I discovered
a language
called

python!

I am lazy.

Me, circa 2006

Me, circa 2008

(still lazy)

(aww!)

I am lazy.

- I am too lazy to wait for the compiler
- I am too lazy to switch between programming languages all the time
- I am too lazy to look up documentation
- I am too lazy to try to decipher line noise

Stuff I need to do:

- Sketch new ideas quickly
- Hack together analysis tools in the field
- Turn prototypes into final research code
- Build web applications for course & departmental use

2.

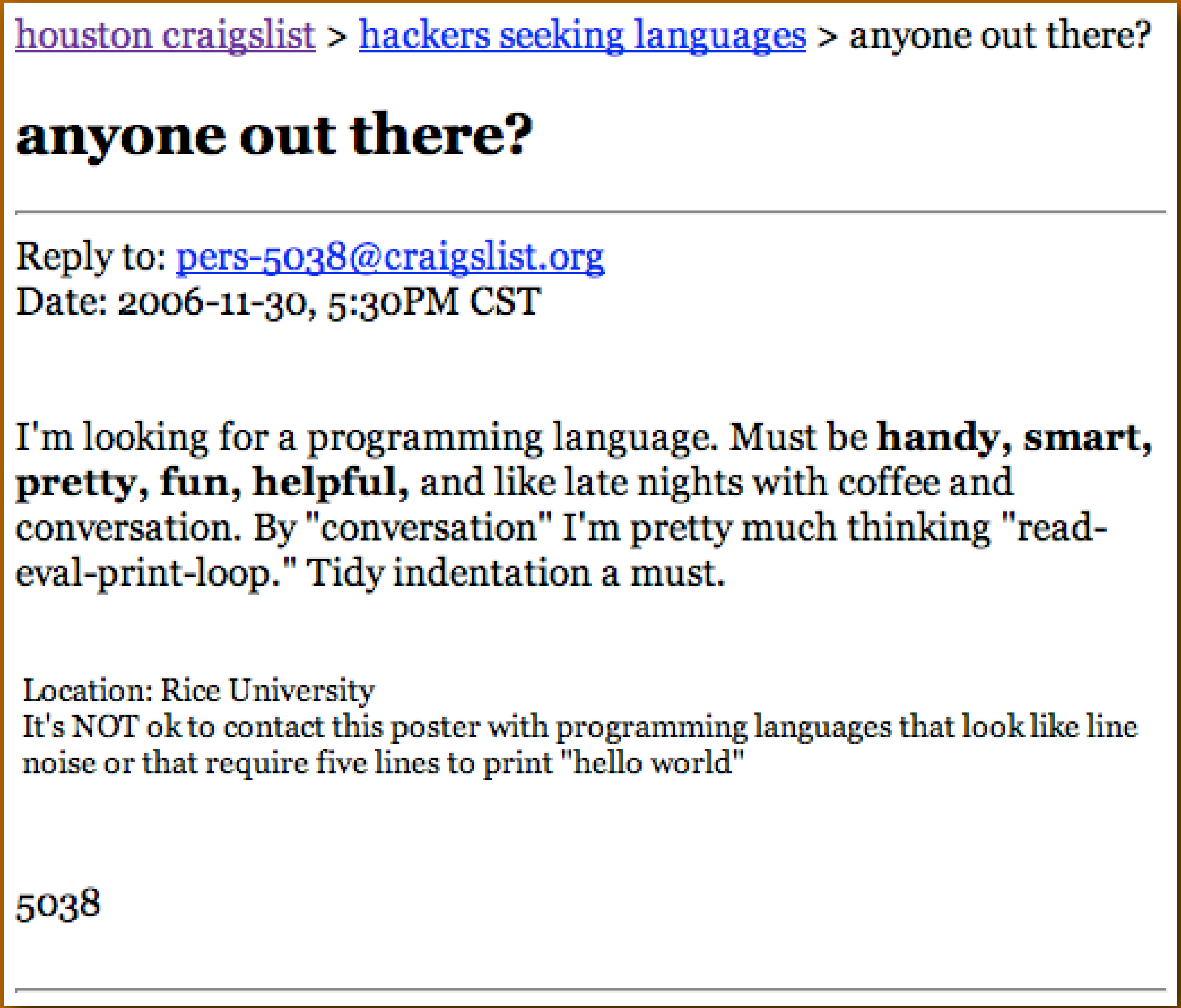
Seriously, tell us about python already.

python is…

- an interpreted programming language
- object-oriented
- dynamically typed
- blah
- blah
- blah

python is…

- handy
- smart
- fun
- helpful
- pretty



houston craigslist > hackers seeking languages > anyone out there?

anyone out there?

Reply to: pers-5038@craigslist.org
Date: 2006-11-30, 5:30PM CST

I'm looking for a programming language. Must be **handy, smart, pretty, fun, helpful,** and like late nights with coffee and conversation. By "conversation" I'm pretty much thinking "read-eval-print-loop." Tidy indentation a must.

Location: Rice University
It's NOT ok to contact this poster with programming languages that look like line noise or that require five lines to print "hello world"

5038

handy

- built-in libraries, everything I might need
  - data structures, concurrency
  - object serialization
  - cross-platform gui
  - sockets/smtp/mime/http/imap/xml/rpc/etc.
  - posix stuff, compression
  - regexps, unicode
  - cgi/httpd
- **other modules:** databases, scientific computing, image processing, crypto, …

smart

- python is not just a "scripting language"
- pick your favorite programming paradigm
  - very fancy OOP
  - functional programming: not just supported, but natural

lambda

fun

- every Python is an interactive workshop
- the “read-eval-print loop” (REPL)
  - ask a question, get an answer
  - the only calculator you’ll ever need
  - experiment with live objects
    - (no waiting for the compiler)

helpful

- Every object, class, function, module is self-documenting
  - “Carry your documentation with you”
    - (It’s actually a property on the object)
- Essential, don’t-leave-home-without-them functions to use in the REPL:

help(*foo*)

dir(*foo*)

pretty

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello, world");
    }
}
```

pretty

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello, world");
    }
}
```

pretty (2)

- Ever seen this?

```
if x:
    x = 2*x
else:
    while x>0:
        x -= 2

    x = math.sqrt(x)
```

The perfect date, indeed

- **handy:** built-in libraries ftw
- **smart:** a real programming language
- **fun:** play around, develop quickly
- **helpful:** documentation is always there
- **pretty:** ugly code is hard to write

3.

The (clean) Python phrasebook.

Simple types

- 1, -2.5, 0xDA51D, 3+5j, 9999999999L
- 'spam', "King Arthur's spam", '''multi-
  line
  spam'''
- True, False
- None

Compound types

- List (mutable sequence)
  - [1, [2, 3], None, "eggs"]
- Dictionary (mutable hash table)
  - {"eggs": "The finest eggs in all Lilliput"
    "spam": "Spiced ham from Camelot"}
- Tuple (immutable sequence)
  - ("eggs", 100, True)

Naming, accessing

- x = 1 *# creates the name x and sets its value*
  # to 1. Oh, hey, this is a comment.
- y = ["eggs", 2, 3]
- z = {"a": 1, "b": 2}

- x + 1   *# result is 2*
- y[1]   *# result is also 2*
- z["b"]   *# ...still 2*

String formatting

- "Hello **%s** (**%d** years old)." **%** (whom, age)
- "Hello **%(name)s** (**%(age)d** years old)." **%** aDict

Functions, conditionals

```
def fib(n):
    """Recursively computes the nth Fibonacci
    number."""
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-1) + fib(n-2)
```

A simple program: “wc”

```
            import java.lang.*;
            import java.io.*;

            public class wc {
                public static void main(String[] argv) throws IOException {
open file           BufferedReader input = new BufferedReader(
                                               new InputStreamReader(System.in));
initialize          boolean done = false;
                    long count = 0;
                    while (!done) {
read                    String line = input.readLine();
                        if (line == null) {
bail                        done = true;
                        } else {
                            boolean inword = false;
each char                   for (int i = 0; i < line.length(); i++) {
                                char ch = line.charAt(i);
                                if (!inword) {
new word,                           if (ch != ' ') {
increment                               inword = true;
counter                                 count += 1;
                                    }
                                } else {
                                    if (ch == ' ') {
end word                                inword = false;
                                    }
                                }
                            }
                        }
                    }
print               System.out.println(count);
                }
            }
```

“wc” in python (1)

read

bail

every
char

new word

end
word

print

```
import sys

done = False
count = 0
while not done:
  line = sys.stdin.readline()
  if line == '': # EOF
    done = True
  else:
    inword = False
    for char in line:
      if not inword:
        if not char.isspace():
          inword = True
          count += 1
      else:
        if char.isspace():
          inword = False

print count
```

“wc” in python (2)

```
import sys
done = False
count = 0
while not done:
  line = sys.stdin.readline()
  if line == '': # EOF
    done = True
  else:
    count += len(line.split())
print count
```

length of any sequence

split string
(default: into words)

“wc” in python (3)

```
import sys
count = 0
for line in sys.stdin:
  count += len(line.split())
print count
```

iterate lines in a file

“wc” in python (4)

```
import sys
print len(sys.stdin.read().split())
```

read whole file into a string

the Java version

```
import java.lang.*;
import java.io.*;

public class wc {
    public static void main(String[] argv) throws IOException {
        BufferedReader input = new BufferedReader(
                                   new InputStreamReader(System.in));
        boolean done = false;
        long count = 0;
        while (!done) {
            String line = input.readLine();
            if (line == null) {
                done = true;
            } else {
                boolean inword = false;
                for (int i = 0; i < line.length(); i++) {
                    char ch = line.charAt(i);
                    if (!inword) {
                        if (ch != ' ') {
                            inword = true;
                            count += 1;
                        }
                    } else {
                        if (ch == ' ') {
                            inword = false;
                        }
                    }
                }
            }
        }
        System.out.println(count);
    }
}
```

4.

Ride the snake.

Beyond the interpreter

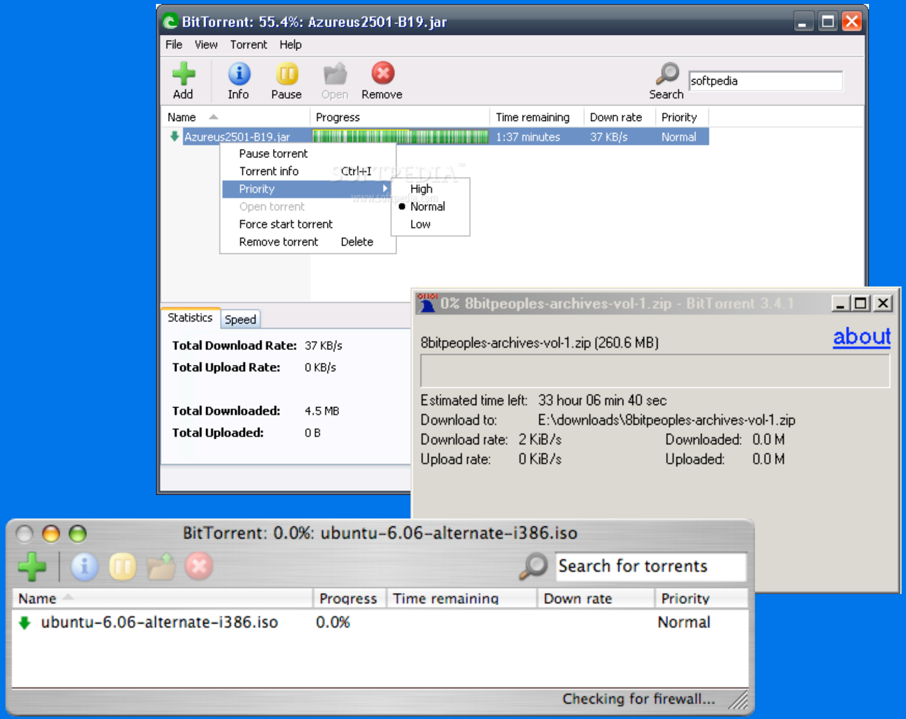
- Need speed?
  - Implement performance-critical code in C, wrap it in a Python interface for reuse
- Missing call/cc?
  - Stackless Python
- Stuck in another runtime?
  - Jython: Python syntax + JVM
  - IronPython: Python syntax + .NET CLR

Scientific computing

- Numeric python - http://numpy.scipy.org/
- pylab/matplotlib - http://matplotlib.sf.net/
- gnuplot.py, pydot
- *Next time you reach for matlab or gnuplot, try python instead*

End-user applications

- GUI toolkits:
  - Tk
  - wxWidgets
  - MFC (Windows
  - Cocoa (OS X)
- BitTorrent



BitTorrent: 55.4%: Azureus2501-B19.jar

Name	Progress	Time remaining	Down rate	Priority
Azureus2501-B19.jar		1:37 minutes	37 KB/s	Normal

Total Download Rate: 37 KB/s
Total Upload Rate: 0 KB/s
Total Downloaded: 4.5 MB
Total Uploaded: 0 B

0% 8bitpeoples-archives-vol-1.zip - BitTorrent 3.4.1

8bitpeoples-archives-vol-1.zip (260.6 MB)

Estimated time left: 33 hour 06 min 40 sec
Download to: E:\downloads\8bitpeoples-archives-vol-1.zip
Download rate: 2 KiB/s Downloaded: 0.0 M
Upload rate: 0 KiB/s Uploaded: 0.0 M

BitTorrent: 0.0%: ubuntu-6.06-alternate-i386.iso

Name	Progress	Time remaining	Down rate	Priority
ubuntu-6.06-alternate-i386.iso	0.0%			Normal

Checking for firewall...

Industry

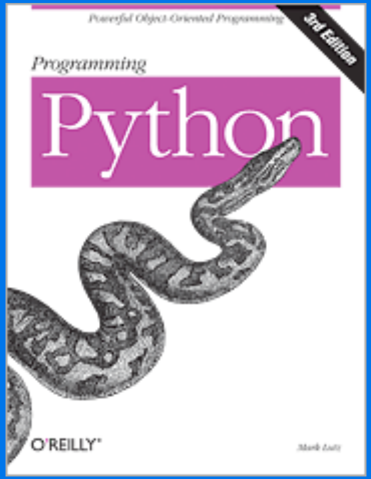
- **Indexing** the known universe (Google)
- **Exploring** the known universe (NASA)
- **Recreating** the known universe (ILM)
- **Destroying** the known universe (Eve Online)

Education

- Good for teaching
  - All the simplicity of Scheme
  - Familiar infix math
  - Widely used outside the academy
- A trend in CS education
  - MIT intro curriculum has gone Python
  - And now…Rice's new **COMP 140**

Further reading

- *Dive Into Python*
  http://diveintopython.org/
- *Programming Python* (O'Reilly)
- More links, plus these slides:
  http://www.cs.rice.edu/~dsandler/python/



(updated 2006)

1.

(my python story)

2.

(about the language)

3.

(syntax)

4.

(resources)

16 tonnes

Brought to you by

the Ministry of Silly Talks