Python for Earth Scientists

Andrew Walker

andrew.walker@bris.ac.uk

for Earth Scientists:
27 & 29 Sept. 2011

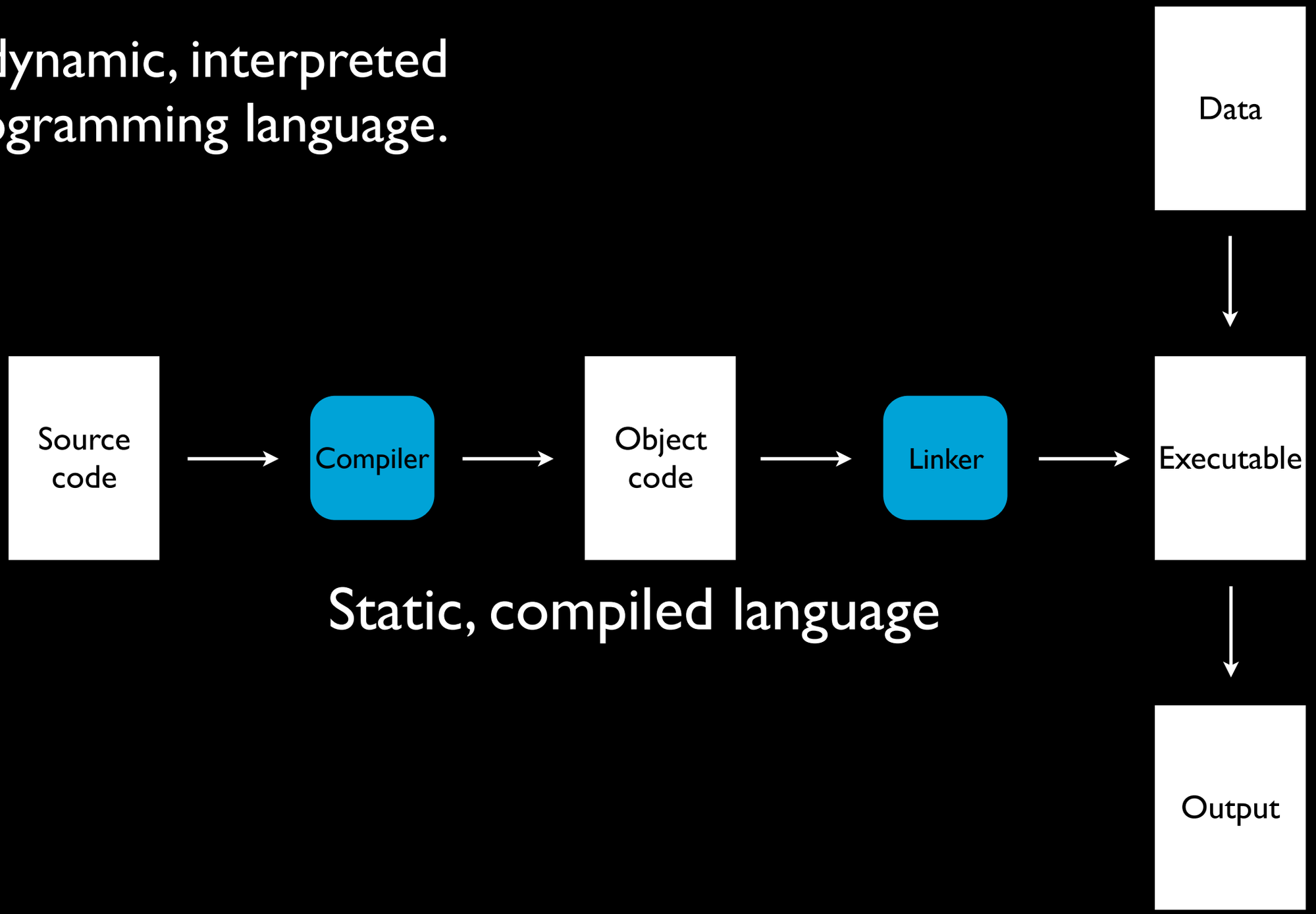
Python is:

A dynamic, interpreted
programming language.

for Earth Scientists:
27 & 29 Sept. 2011

Python is:

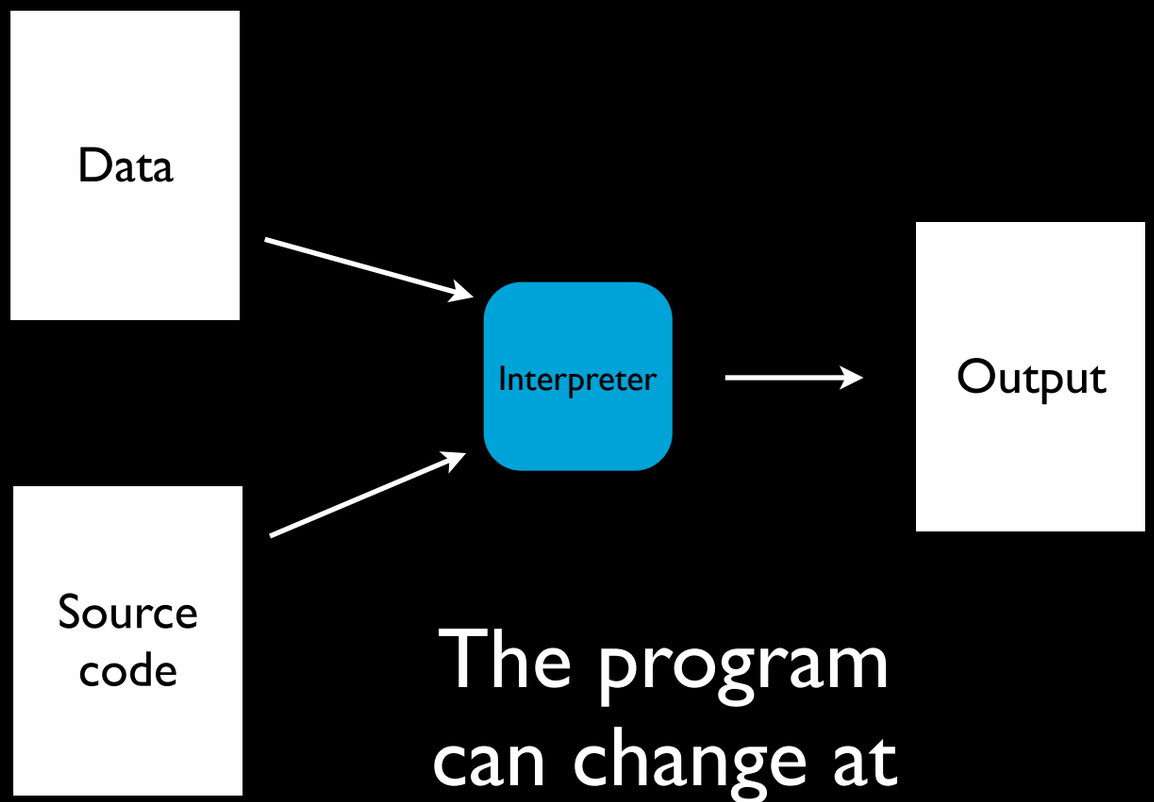
A dynamic, interpreted programming language.



Static, compiled language

Python is:

A dynamic, interpreted programming language.

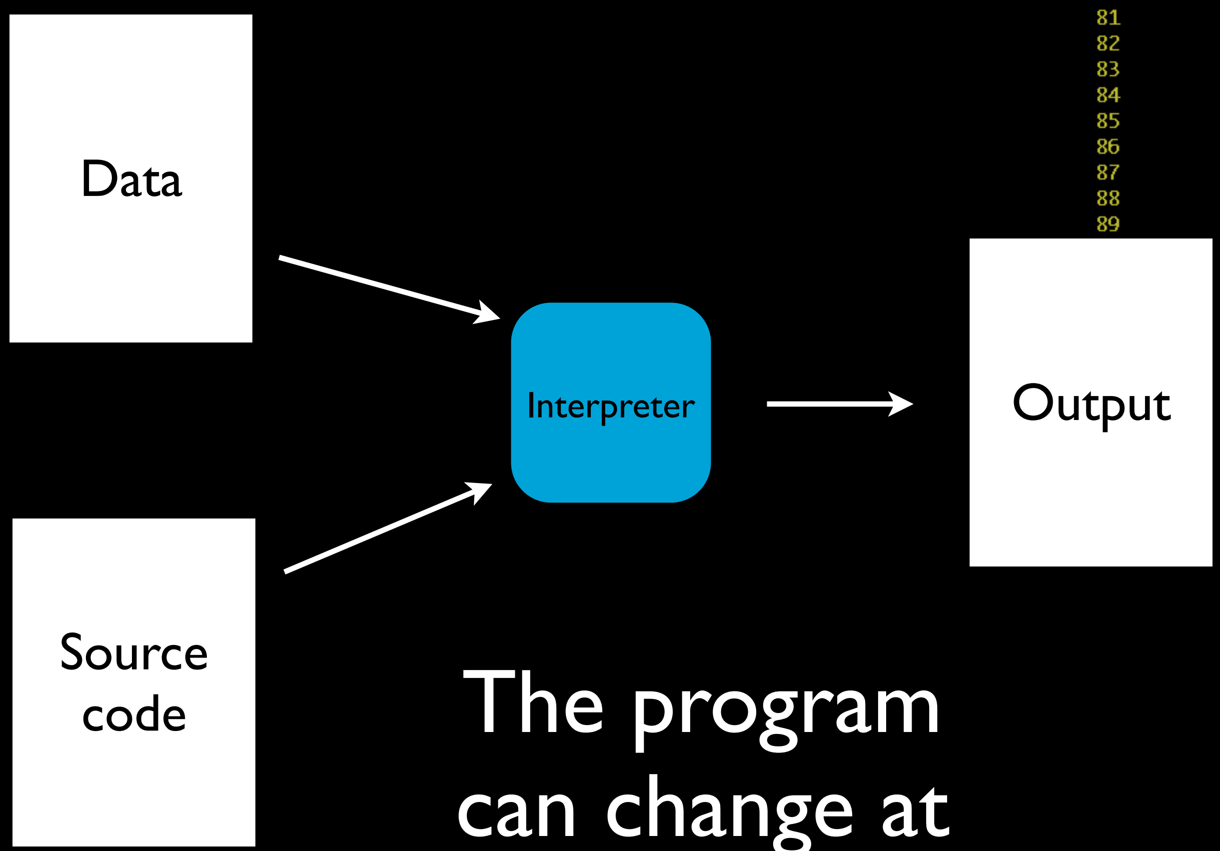


The program can change at run time

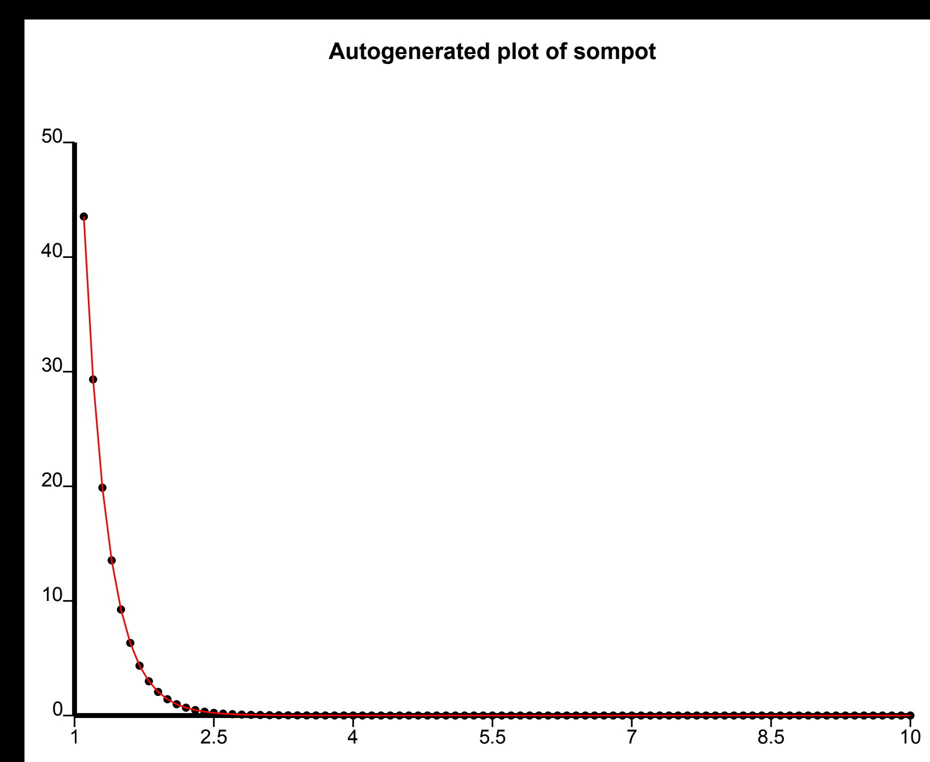
Python is:

A dynamic, interpreted programming language.

```
    <parameter name="a" dictRef="gulp:buckingham.a">
      <scalar units="gulpunits:eV">2.409505000000e3</scalar>
    </parameter>
    <parameter name="roh" dictRef="gulp:buckingham.roh">
      <scalar units="gulpunits:Ang^-1">2.649000000000e-1</scalar>
    </parameter>
    <parameter name="c" dictRef="gulp:buckingham.c">
<!--Modified from 0 to 10 to show how it changes graph)-->
      <scalar units="gulpunits:eV^-6">-10.000000000000e0</scalar>
    </parameter>
<!-- The chunk of stuff within the expression will be the same for all
     instances of this type of potential - so this can be included with
     Xinclude. The idea is to provide enough info (with a dictRef) so that
     e.g. Fortran codes can use the potential if they are not network enabled) -->
    <expression dictRef="mypotentialDict:buckingham">
      <scalar units="eV"/>
      <m:math>
        <m:lambda>
          <m:bvar><m:ci>R</m:ci></m:bvar>
          <m:apply>
            <m:minus/>
            <m:ci>
              <m:apply>
                <m:times/>
                <m:ci>a</m:ci>
                <m:ci>
                  <m:apply>
                    <m:exp/>
                    <m:ci>
                      <m:apply>
                        <m:divide/>
                        <m:ci>
                          <m:apply>
                            <m:minus/>
                            <m:ci>R</m:ci>
                          </m:apply>
                        </m:ci>
                        <m:ci>roh</m:ci>
                      </m:apply>
                    </m:ci>
                  </m:apply>
                </m:ci>
              </m:apply>
            </m:ci>
          </m:apply>
            </m:ci>
```



The program can change at run time



Python is:

A dynamic, interpreted
programming language...

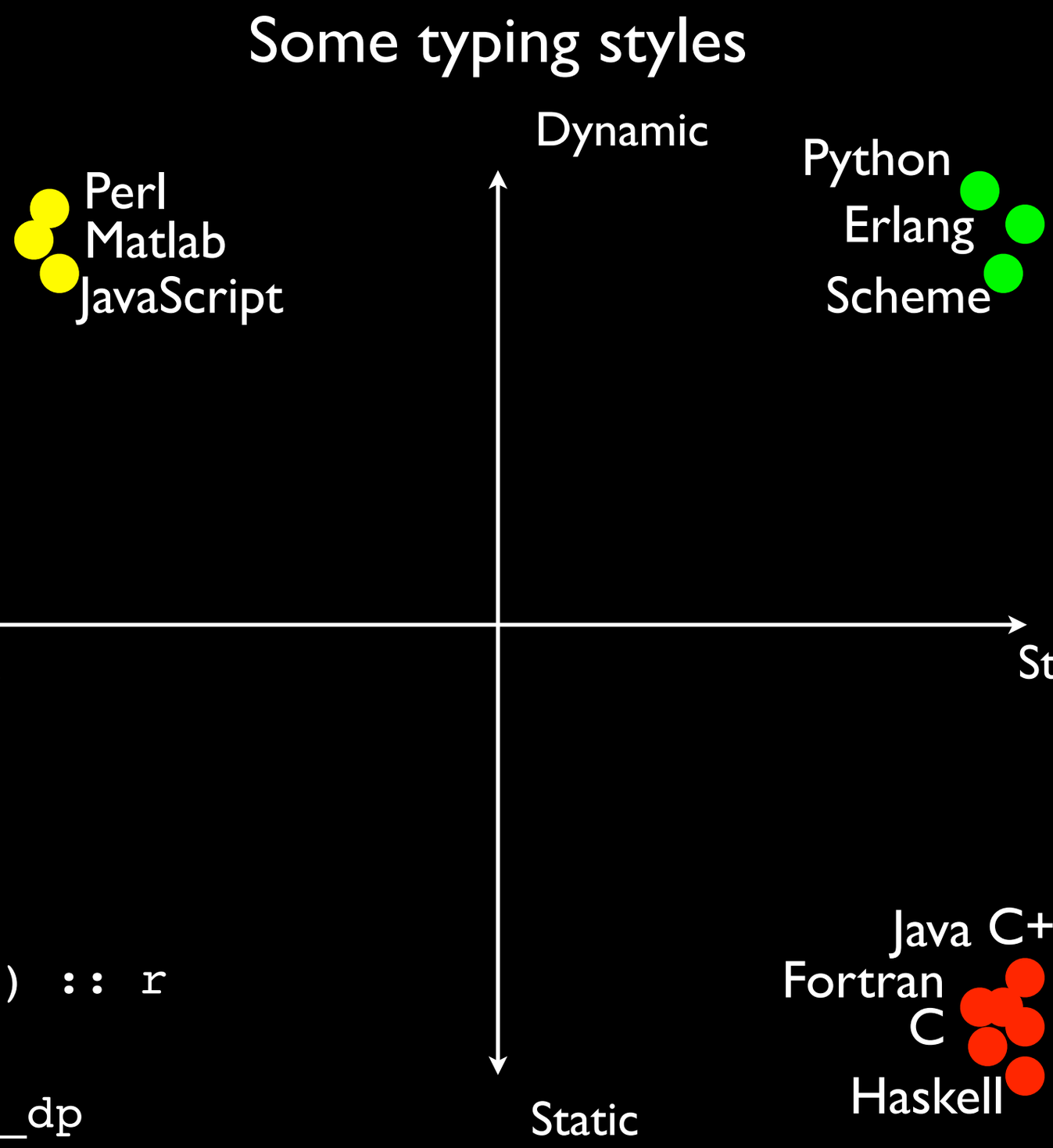
... strongly typed but
dynamically typed.

for Earth Scientists:
27 & 29 Sept. 2011

Python is:

A dynamic, interpreted programming language...

... strongly typed but dynamically typed.



Weak

```
x = 1 + "2.0"
```

Dynamic

```
a = 5
...
a = "cat"
```

Static

```
real(dp) :: r
...
r = 5.2_dp
```

Python is:

A dynamic, interpreted
programming language...

... strongly typed but
dynamically typed ...

... imperative and object
oriented.

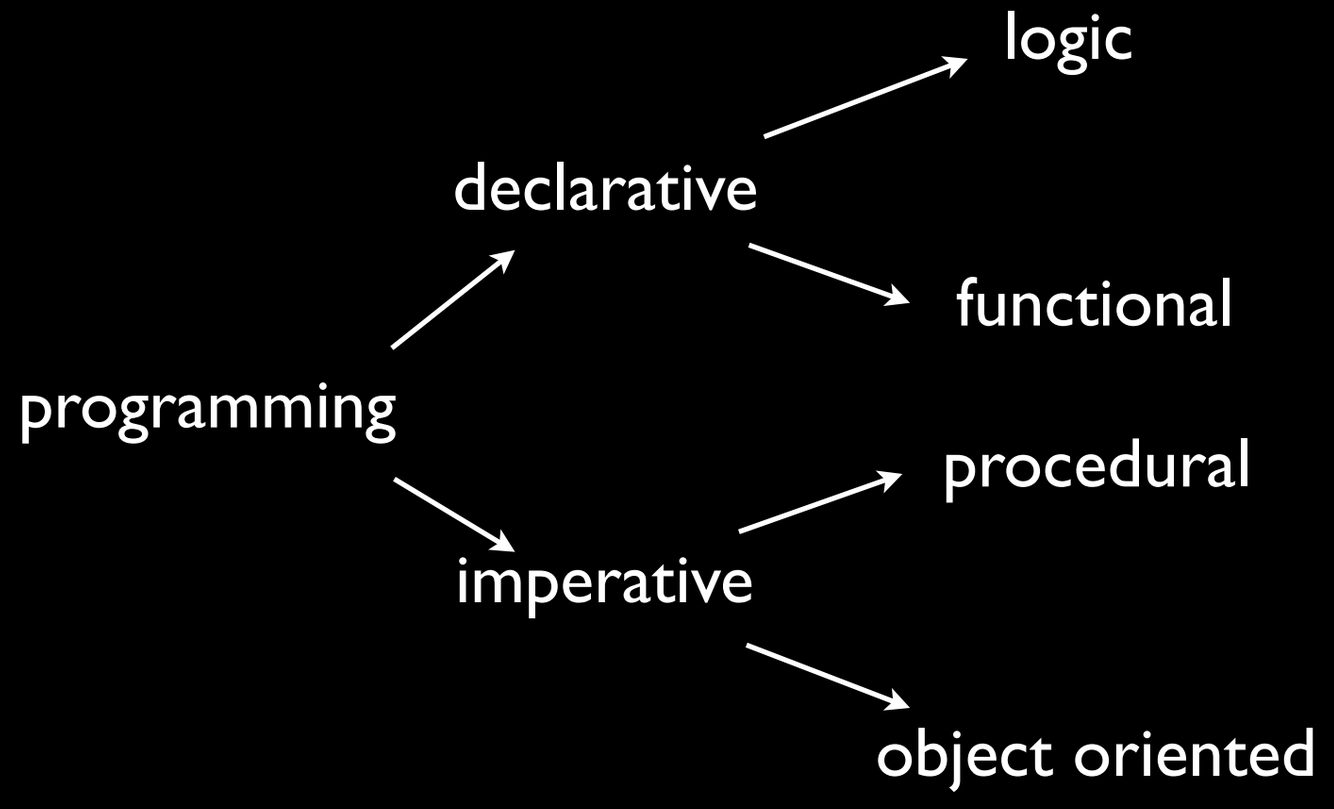
Python is:

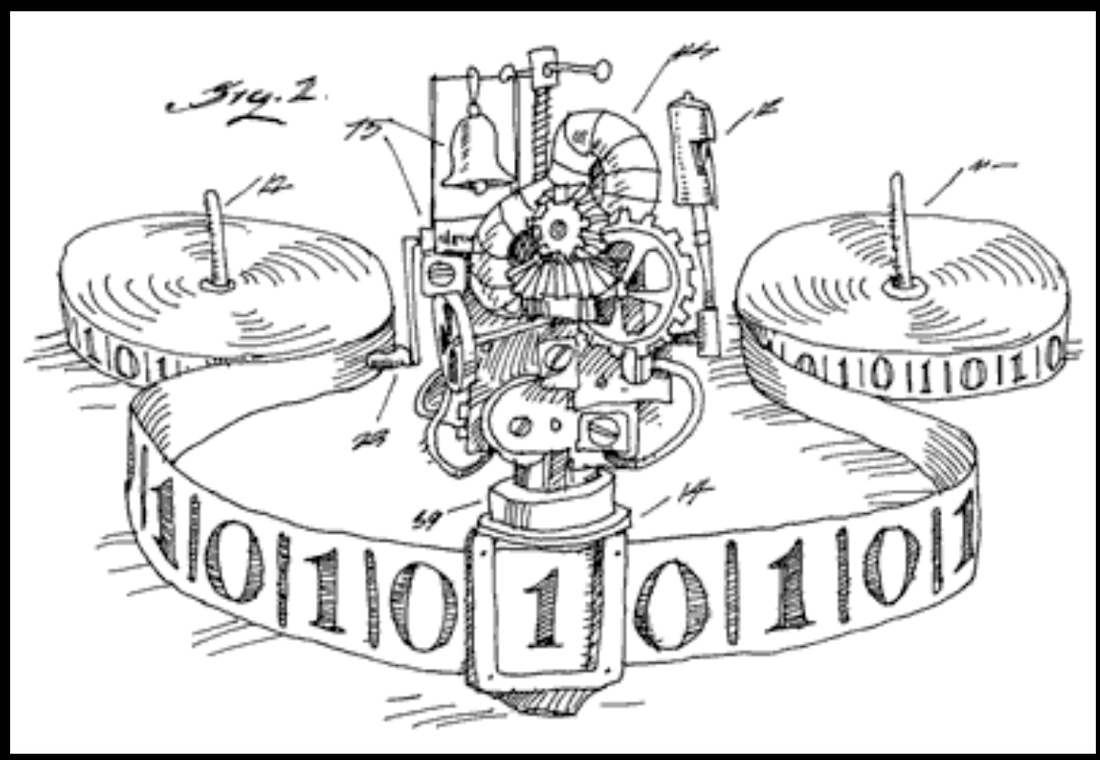
A dynamic, interpreted programming language...

... strongly typed but dynamically typed ...

... imperative and object oriented.

(Can look procedural)





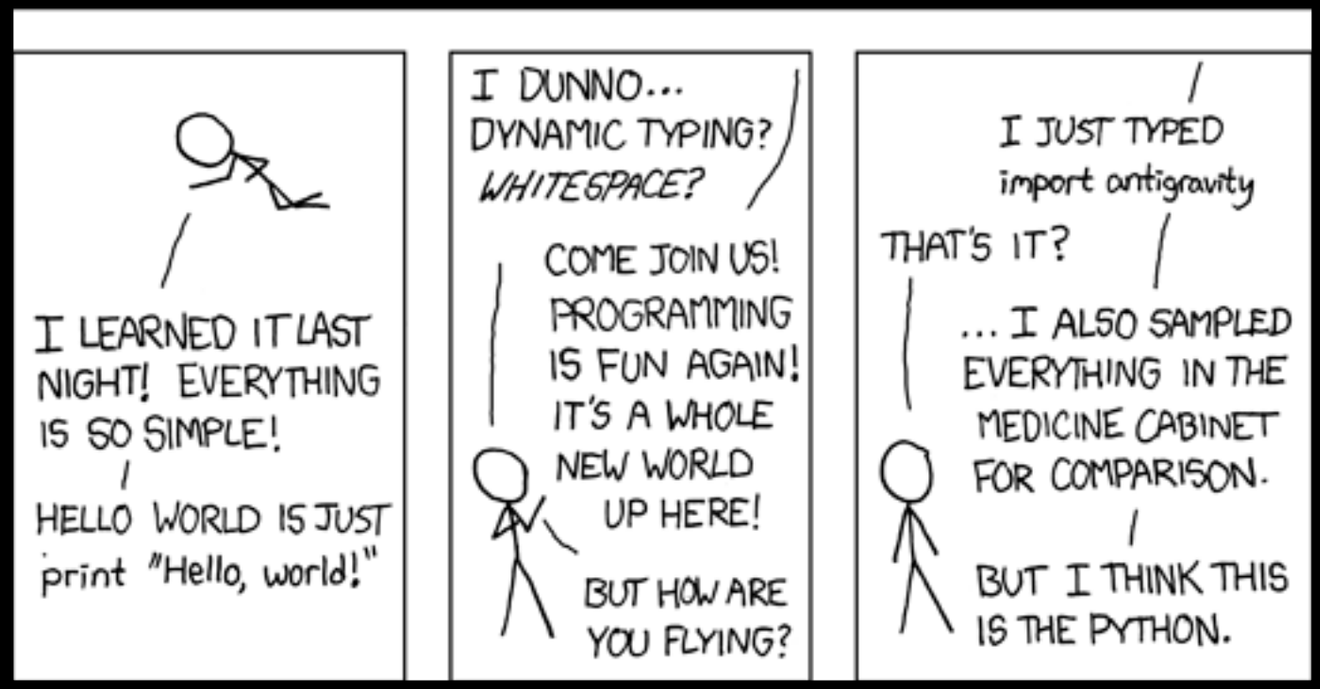
Python is:

A dynamic, interpreted programming language...

... strongly typed but dynamically typed ...

... imperative and object oriented ...

... fun to learn and use.



http://xkcd.com/353/

Portable

Good as ‘glue’

Open

Multi-paradigm

Modern

Testable

Designed

Opinionated

Comes with
batteries included

Evolving

Encourages rapid
(agile) development

Free

Today

Now.	I'll talk for ~30 mins on Python's syntax and landscape
2 pm.	Practical 1: First steps
3 pm.	I'll say something about types, modules, scripts and the standard library
4 pm.	Practical 2: Scripts and modules

Thursday

1 pm.	I'll explain object oriented programming
2 pm.	Practical 3: Object oriented programming
3 pm.	I'll introduce SciPy, NumPy and Matplotlib
4 pm.	Practical 4: NumPy, SciPy and Matplotlib

We should finish by 5 pm. Don't worry if you don't finish all the exercises.

Syntax

```
print “Hello, world!”
```

Starting with this is compulsory

```
a = 1 # an integer
b = 2.7 # a float
c = “Hello, world” # a string
print a
print b
print c
x1 = 6 # This is OK
1x = 7 # This is an error
```

Variable assignment, comments and three types

Syntax

```
a = 1
b = 10
c = a + b # What is c?
(4 < 5) # True
(4 > 5) # False
((4 < 5) or (4 > 5)) # True
```

Some mathematical and logical operators.

Syntax

```
if sky == ‘blue’:
    birdsong = True
elif sky == ‘black’:
    birdsong = False
else:
    pass  #do nothing
```

If, elif, else. Note indentation. No case statement.

Syntax

```
a = 0
while (a < 10):
    print a
    a = a + 1
```

While loop

```
for a in range(10):
    print a
```

Iteration

Syntax

```
def add_ten (value):
    value = value + 10
    return value
```

Function definition

```
b = add_ten(5)
print b
```

Function use

Landscape

```
#!/usr/bin/env python
"""
List files in the working
directory

"""
import os

def main():
    print os.getcwd()
    for fn in os.listdir('.'):
        print fn

if __name__ == '__main__':
    main()
```

The interpreter

python

Language parser and implementation

Host computer & operating system: access to disks, network, users etc.

Landscape

```
#!/usr/bin/env python
"""
List files in the working
directory

"""
import os

def main():
    print os.getcwd()
    for fn in os.listdir('.'):
        print fn

if __name__ == '__main__':
    main()
```

The interpreter

python

Language parser and implementation

User module

Third party module

Standard library module

Extension module (compiled)

Host computer & operating system: access to disks, network, users etc.

The interpreter

Language parser and implementation

CPython	Jython	IronPython	PyPy
C	Java	C#	Python
Reference implementation. Runs on windows, unix, mac etc. Extension modules are .so/.dll	Targets Java virtual machine. Can load Java libraries as extension modules	Targets .NET framework. Can integrate with silverlight, visual basic etc.	Compiler and implementation. Includes JIT compiler and can be very fast

Standard library

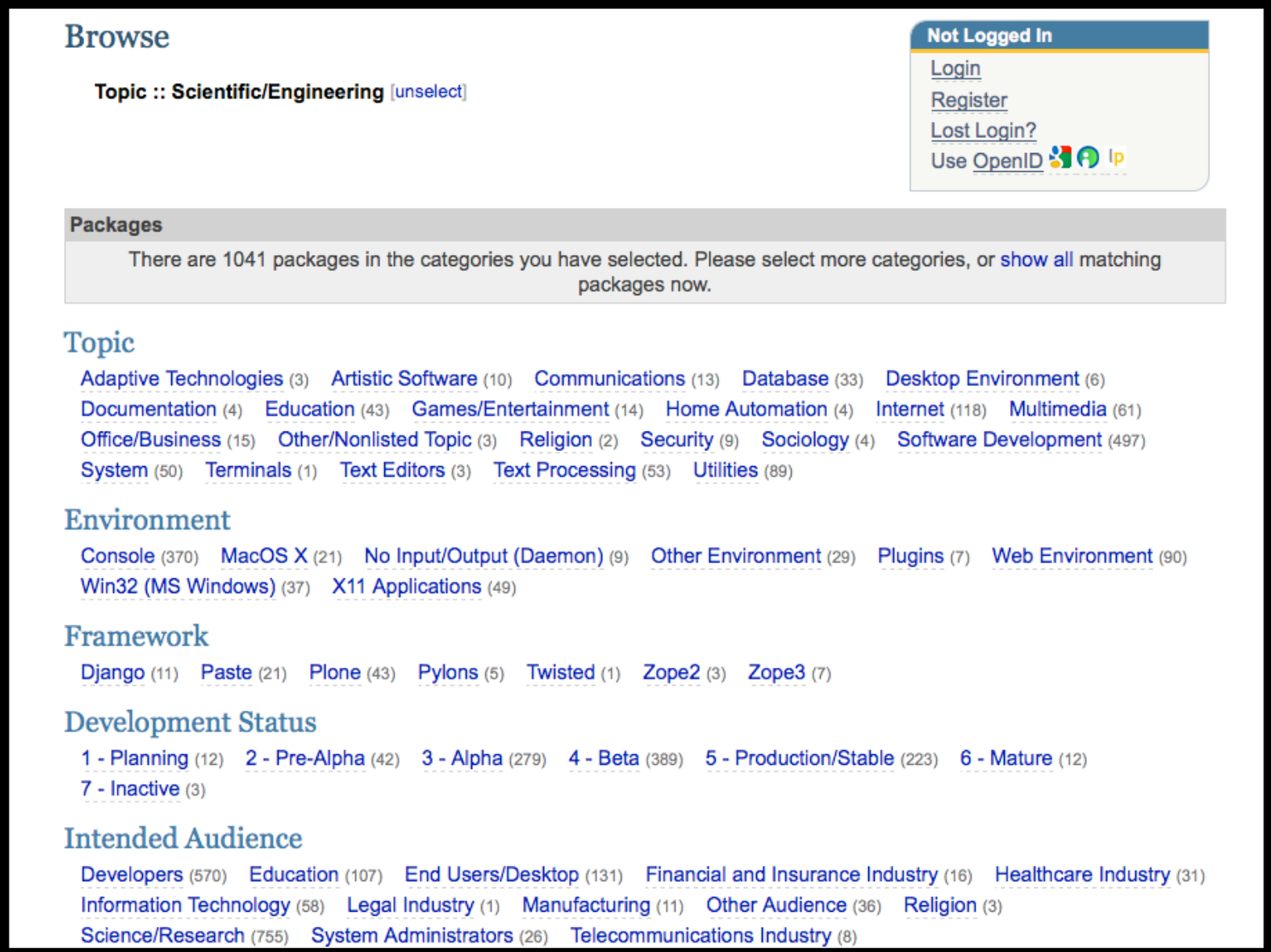
~300 modules designed to be shipped with interpreter

- Mathematical functions
- Advanced string processing
- Many data types and specialised algorithms
- Threading and multiprocessing
- Access to OS facilities
- Networking: sockets, http, email, ftp
- Data storage, database access
- Compression, encryption, hashing
- File formats, XML
- Graphics

http://docs.python.org/library/

Package index

16725 free to use packages (collections of modules) in a searchable index



Browse

Topic :: Scientific/Engineering [unselect]

Not Logged In
Login
Register
Lost Login?
Use OpenID

Packages

There are 1041 packages in the categories you have selected. Please select more categories, or show all matching packages now.

Topic

Adaptive Technologies (3) Artistic Software (10) Communications (13) Database (33) Desktop Environment (6)
Documentation (4) Education (43) Games/Entertainment (14) Home Automation (4) Internet (118) Multimedia (61)
Office/Business (15) Other/Nonlisted Topic (3) Religion (2) Security (9) Sociology (4) Software Development (497)
System (50) Terminals (1) Text Editors (3) Text Processing (53) Utilities (89)

Environment

Console (370) MacOS X (21) No Input/Output (Daemon) (9) Other Environment (29) Plugins (7) Web Environment (90)
Win32 (MS Windows) (37) X11 Applications (49)

Framework

Django (11) Paste (21) Plone (43) Pylons (5) Twisted (1) Zope2 (3) Zope3 (7)

Development Status

1 - Planning (12) 2 - Pre-Alpha (42) 3 - Alpha (279) 4 - Beta (389) 5 - Production/Stable (223) 6 - Mature (12)
7 - Inactive (3)

Intended Audience

Developers (570) Education (107) End Users/Desktop (131) Financial and Insurance Industry (16) Healthcare Industry (31)
Information Technology (58) Legal Industry (1) Manufacturing (11) Other Audience (36) Religion (3)
Science/Research (755) System Administrators (26) Telecommunications Industry (8)

http://pypi.python.org/

Distribution

The interpreter

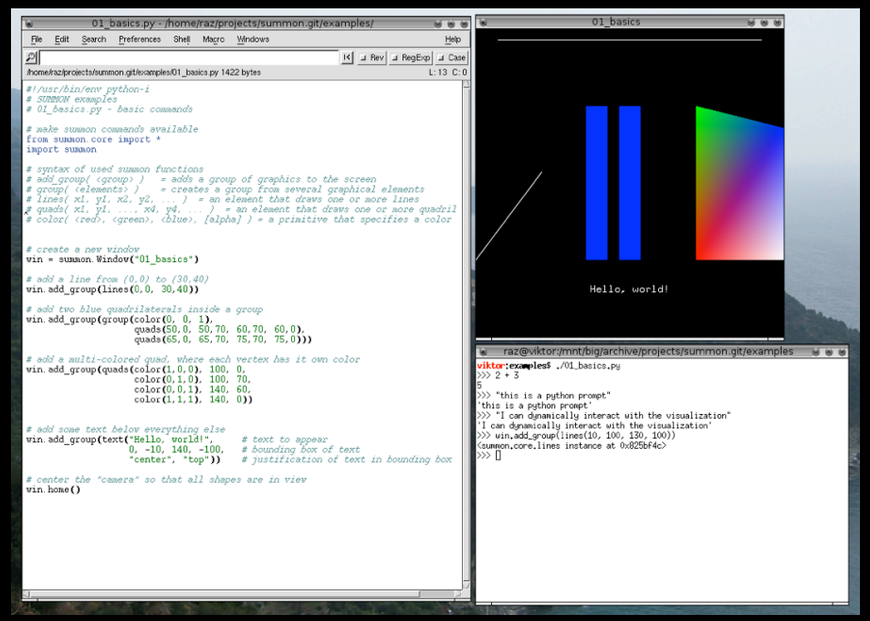
python

Language parser and implementation

Standard library modules

Third party modules

Module management and upgrade tool



Integrated development environment

Interpreter ± collection of modules ± other stuff = Python distribution

python

for Earth Scientists:
27 & 29 Sept. 2011

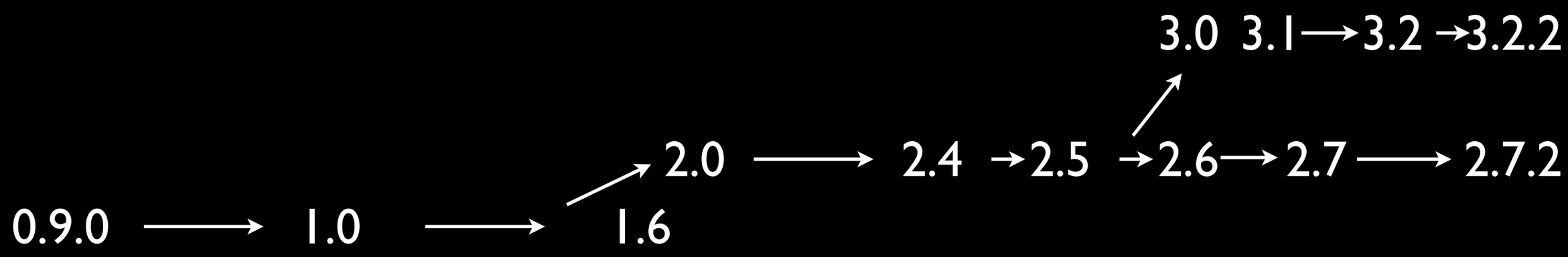
Some distributions

- Official CPython distribution
- Distribution with OS: Mac, Linux
- Commercial: ActivePython, Enthought
- Specialised: stackless, embedded

http://wiki.python.org/moin/PythonDistributions

Versioning

In principle language version is separate from interpreter version and versioning of modules. However, CPython and Python stay in lock-step



Development

Interpreter developed in a similar way to other large pieces of open source code (distributed version control, bug tracker, etc.)

Language developed by the adoption of Python Enhancement Proposals (PEPs) - open process but ultimately decided on by "BDFL"

(Python Language Moratorium until late 2012)

http://www.python.org/dev/peps/
http://docs.python.org/devguide/#

http://docs.python.org/

http://www.python.org/

for Earth Scientists:
27 & 29 Sept. 2011

Enthought Python Distribution

Commercial python distribution aimed at scientists. Free for academics. What we will use because it is easy to install and has all the modules we need.

http://www.enthought.com/products/epd.php

http://www1.gly.bris.ac.uk/~walker/PythonEarthSci/