



An Introduction to Python (A One-Hour Tour)

by Binh Q. Nguyen

ARL-TN-0328

July 2008

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory

Adelphi, MD 20783-1197

ARL-TN-0328

July 2008

An Introduction to Python (A One-Hour Tour)

Binh Q. Nguyen

Computational and Information Sciences Directorate, ARL

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.					
1. REPORT DATE (DD-MM-YYYY) July 2008		2. REPORT TYPE Summary		3. DATES COVERED (From - To) FY08	
4. TITLE AND SUBTITLE An Introduction to Python (A One-Hour Tour)				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Binh Q. Nguyen				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: AMSRD-ARL-CI-NT 2800 Powder Mill Road Adelphi, MD 20783-1128				8. PERFORMING ORGANIZATION REPORT NUMBER ARL-TN-0328	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT This tutorial highlights and goes over essential features of the Python programming language while it is still evolving, but sufficiently stable and mature for the development of diverse solutions to computational, networking, and visualization problems. Although the technical details are kept to a minimum to fit diverse background and interests of the audience, they can be used as review materials for experienced and occasional developers of Python applications. The tutorial was presented to a team of engineers, scientists, and summer students on Wednesday 18 June 2008 at the U.S. Army Research Laboratory in Adelphi, MD.					
15. SUBJECT TERMS The Python programming language, tutorial					
16. Security Classification of:			17. LIMITATION OF ABSTRACT U	18. NUMBER OF PAGES 20	19a. NAME OF RESPONSIBLE PERSON Binh Q. Nguyen
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U			19b. TELEPHONE NUMBER (Include area code) (301) 394-1781




An Introduction to Python

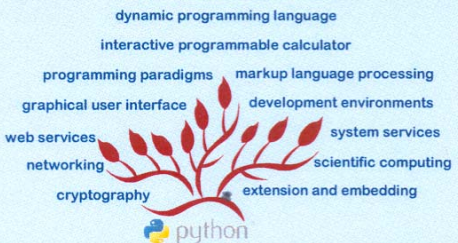
(A One-Hour Tour)

Binh Q. Nguyen
U.S. Army Research Laboratory
 Adelphi, Maryland
 June 2, 2008

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.




topics



dynamic programming language
 interactive programmable calculator
 programming paradigms markup language processing
 graphical user interface development environments
 web services system services
 networking scientific computing
 cryptography extension and embedding

python

June 2, 2008 An Introduction to Python 2

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.




objectives

- high-level overview of the language
 - features
 - utilities
 - examples
 - sources
- information and suggestion to
 - decide whether to use Python
 - download and run the Python interpreter
 - use it interactively

June 2, 2008 An Introduction to Python 3

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.


interactive calculator ...


```

>>> a = 12.34      # float
>>> i = 4
>>> x = a * i + 11   # float number x = 60.35999999999999
>>> a = 12          # integer a = 12
>>> a * i + 11       # a, i, 11 are integers ==> integer 59
>>> (1 + 2j) * (3 - 4j) # complex number (11+2j)
>>> 2**2048         # big number
323170060713110073007148766886699519604441026697154840321303
45427524655138967890893197201411522913463887179609218960194
941195591504909210950881523864482331206308773673009960917501
977503896521067960576383840675682767922186426197661618380943
384761704705816458520363050428675758915410658086075523991239
303855219143333896683424208849747865645694948561760353283220
580778056593310261927084603141502585928641771167250436037184
61857357598351152301645904403697613232872312271258847108202
09725157101726931323469678542586586978554539972583525986392
155251663894373355436021354332296046453184786049521481935558
53611059596230656L

```

dynamic data types

*** int, float, long, complex, big numbers ***

June 2, 2008
An Introduction to Python
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.
4


built-in types ...


```

>>> a = 0x0f; b = 0360; c = 3+4j; d = 2.3e10
>>> oct(a|b), oct(a*b), oct(a&b), hex(a<<2), hex(b>>1), hex(-a)
('0377', '0377', '0', '0x3c', '0x78', '-0x10')
>>> c.imag          # Bitwise Operations
4.0                ^, |, &, >>, <<, ~
>>> c.real          # Binary Operations
3.0                +, -, *, /, ** (pow()), //, %, divmod()
>>> abs(c)          # Unary Operations
5.0                +, -, ~
>>> c + c.conjugate() # Conversion Operations
(6+0j)             abs(), bool(), int(), float(), long(), complex()
>>> 5.9 // 2        2.0
>>> 5.9 % 2         1.900000000000
>>> divmod(5.9, 2) (2.0, 1.900000000000)

```

Bitwise Operations
^, |, &, >>, <<, ~

Binary Operations
+, -, *, /, ** (pow()), //, %, divmod()

Unary Operations
+, -, ~

Conversion Operations
abs(), bool(), int(), float(), long(), complex()

June 2, 2008
An Introduction to Python
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.
5


... built-in types ...


```

>>> x=True; y=False; z=None
>>> x or y and z
True
>>> a = 10
>>> b = 10 * 2
>>> a > b          None
False            absence of a value
>>> a < b          False
True            None, False, zero, empty
>>> a is b         True
False          All other values
>>> a is b         False
>>> a = b          Logical Operations
>>> a is b         and, or, not
True            Comparison Operations
>>> a is not b     <, <=, ==, !=, >, >=, is, is not
False

```

Built-in Constants
absence of a value

Logical Operations
and, or, not

Comparison Operations
<, <=, ==, !=, >, >=, is, is not

June 2, 2008
An Introduction to Python
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.
6


... built-in types ...


- Sequence types** – ordered collection
 - str – "abc", "123", ..., 'ab', 'ab', 'a'@'x', 'a'@'x', 'a'@'x'
 - list – [], [1, 2, 3, 'a', 'b', 'c', ...] – mutable
 - tuple – (1,), (1, 2, 3, 'a', 'b', 'c', ...) – immutable
- Mapping type** – unordered collection
 - dict – {}, {'a': 1, 'b': 2, ...}
 - keys & values (name-value pairs)
 - **unindexable – use keys to retrieve values**
- Set** – unordered collection
 - set – mutable
 - frozenset – immutable

June 2, 2008
An Introduction to Python
7

TECHNOLOGY DRIVEN
WARFIGHTER FOCUSED


other string types


- cStringIO**
 read and write strings as files
- struct**
 string as a packed binary data
 pack(), unpack(), calcsize(), ...
- re**
 regular expression
- ...

June 2, 2008
An Introduction to Python
8

TECHNOLOGY DRIVEN
WARFIGHTER FOCUSED


control-flow statements


- for i in <sequence> :**
 statement(s)
- if <> :**
elif <> :
elif <> :
 ...
else:
- continue** # start next iteration
- break** # stop iteration
- return** # [to caller]
- while <> :**
 statement(s)
- for i in range(10):** # i = 0..9
 sum += i # sum = sum+i
- for x in range(100, 200, 1):**
 if x % 2 == 0:
 s += x
continue # next iteration, skip the remaining statements.
 if x % 3 == 0:
 print '<<d> divides 3' % (x)
 elif x % 5 == 0:
 print '<<d> divides 5' % (x)
 elif x % 7 == 0:
 print 'bailing out'
 break
 else:
 print 'x=', x
- while x > 0:**
 x = x - 1

June 2, 2008
An Introduction to Python
9

TECHNOLOGY DRIVEN
WARFIGHTER FOCUSED


conditional expression


x = A if <condition == True> else B

equivalent form

```

if A <condition == True> :
    x = A
else:
    x = B

```

```

>>> MAXRANGE = 200
>>> for d in [ 166, 239, 192, 241, 207 ]:
    print d, ': within range' if d < MAXRANGE else 'out of range'
166 : within range
239 : out of range
192 : within range
241 : out of range
207 : out of range

```

June 2, 2008

An Introduction to Python

10

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.


bitwise ops & control flow


```

>>> a = 0xff
>>> b = 0x0f
>>> hex(a ^ b)
'0xf0'
>>> hex(a | b)
'0xff'
>>> hex(a & b)
'0xf'
>>> hex(a >> 4)
'0xf'
>>> hex(a << 8 | 0x0c)
'0xff0c'
>>> hex(a ^ a)
'0x0'

```

```

>>> n = 2**16
>>> while n:
    print n,
    n = n >> 1
65536 32768 16384 8192
4096 2048 1024 512 256
128 64 32 16 8 4 2 1
>>> n = 1
>>> while n <= 2**16:
    print n,
    n = n << 1
1 2 4 8 16 32 64 128 256
512 1024 2048 4096 8192
16384 32768 65536

```

June 2, 2008

An Introduction to Python

11

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.


built-in functions


```

>>> x = [ 3, 1, 2, -1]
>>> print 'f=%5.1f i=%d s=%s' % (2.12, 4, 'abc'), x
f= 2.1 i=4 s=abc [3, 1, 2, -1]
>>> sorted(x); len(x)          #[-1, 1, 2, 3] 4
>>> max(x); min(x); sum(x)     #3 -1 5
abs(), ord(), chr(), enumerate(), eval(), hex(),
id(), hash(), raw_input(), open(), type(), zip(), ...

```

June 2, 2008

An Introduction to Python

12

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.



function



```

def function_name([arguments]):
    statement(s)
    [return (type)]

def fact(n):
    if n <= 1: return 1
    return n * fact(n-1)
print fact(5) # 120
def fact (n):
    return 1 if n < 2 else n * fact(n-1)

```

procedural

June 2, 2008

An Introduction to Python

13

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.



class



```

class class_name:
    statement(s)

```

object-oriented

```

class Node:
    def __init__(self, name, links):
        self.name = name # Node ID
        self.links = links # list of links
    def get_links(self): return self.links
    def set_links(self, links): self.links = links

```

June 2, 2008

An Introduction to Python

14

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.



inheritance



```

class DerivedClass(Base1, Base2,...):
    statement(s)

class PropModel: # signal propagation model
    def __init__(self, params):
        self.params = params
    def get_params(self): return self.params
    def get_LER(self): pass # no-op, do nothing

class FSL(PropModel): # Free-Space Loss model
    def __init__(self, params):
        PropModel.__init__(params)
    def get_LER (self): # overriding the base class method
        ...
        return LER # return Link-Error Rate

```

June 2, 2008

An Introduction to Python

15

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.



more on str() ...



```

>>> s='abc ... xyz'
>>> s.isupper()           # False
>>> s.capitalize()       # 'Abc ... xyz'
>>> s.upper()            # 'ABC ... XYZ'
>>> s.endswith('z')      # True
>>> s.startswith('a')    # True
>>> s.endswith('xyz')    # False
>>> s[2:5]               # 'c...'
>>> s[5:]                # '...xyz'
>>> s.split()             # ['abc', '...', 'xyz']
>>> s.find('c')           # 2
>>> s.rfind('.')          # 6
>>> s + '123'            # 'abc ... xyz123'
>>> s = 'ab'*3           # 'ababab'
>>> help(str)

```

June 2, 2008

An Introduction to Python

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

16



more on list() ...



```

>>> x = [1, 'a', 3, 'z'] # initial values
>>> y = range(6, 11)     # y:= [6, 7, 8, 9, 10]
>>> x.pop(1)             # remove 'a' (the 2nd item) from x
>>> x.pop()              # remove 'z' from x, x.pop(-1)
>>> x                    # x = [1, 3]
>>> x.append(4)           # add '4' to the end of the list x
>>> x.insert(1, 2)        # insert '2' to the 2nd position (i=1)
>>> x.append(5)           # add '5' to the end of the list x
>>> x                    # x = [1, 2, 3, 4, 5]
>>> x.extend(y)           # same as x = x + y
>>> x                    # x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>> x[3:7]               # slice x[3:7]:= [4, 5, 6, 7]
>>> x=[1,2,3]*3          # x = [1, 2, 3, 1, 2, 3, 1, 2, 3]
>>> help(list)           # help(x), help(list)

```

June 2, 2008

An Introduction to Python

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

17



stack & queue using list



- ```

x = range(5) # x:= [0,1, 2, 3, 4]
def size(x): return len(x)
def is_empty(): return size(x) == 0

```
- stack – last in first out (LIFO)**

```

push(5): x.append(5) # x:= [0,1, 2, 3, 4, 5]
pop(): x.pop() # 5, x:= [0,1, 2, 3, 4]

```
- queue – first in first out (FIFO)**

```

enqueue(5): x.append(5) # x:= [0,1, 2, 3, 4, 5]
dequeue(): x.pop(0) # 0, x:= [1, 2, 3, 4, 5]

```

June 2, 2008

An Introduction to Python

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

18

---

---

---

---

---

---

---

---

---

---


tuple() and dict ()


**Tuple**

```

>>> t = (1, 2, 3)
>>> 'a' in t
False
>>> 2 in t
True
>>> t[1]
2

```

**Dictionary**

```

>>> d = {'A': 65, 'C': 0x43}
>>> d['B'] = 0x42
>>> d.keys()
['A', 'C', 'B']
>>> d.values()
#[65, 67, 66]
>>> 'D' in d
False
>>> d.get('C')
67 (0x43)

```

indexable  
immutable

unindexable  
mutable

June 2, 2008
An Introduction to Python
19
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

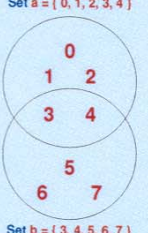
---


set()


```

>>> a = set(range(5))
>>> b = set(range(3,8))
>>> a.intersection(b)
set([3, 4])
>>> a.union(b)
set([0, 1, 2, 3, 4, 5, 6, 7])
>>> a-b
set([0, 1, 2])
>>> b-a
set([5, 6, 7])

```



June 2, 2008
An Introduction to Python
20
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---


handy built-in functions


- range(5) => [0, 1, 2, 3, 4]
- range(2, 13, 3) => [2, 5, 8, 11]
- dir( [object] ) => list of names
- dir() => current names
- dir(str) => [ ..., 'isalnum', ..., 'replace', ..., 'strip', ..., 'upper', 'zfill']
- help(str)
- help(str.strip)

June 2, 2008
An Introduction to Python
21
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---





## creating instances



**[a]<==>[b]<==>[c]**

```

class Node:
 def __init__(self,
 name,
 links=None):
 self.name = name
 self.links = links

 def get_links(self):
 return self.links

 def set_links(self, links):
 self.links = links

a = Node('a')
b = Node('b')
c = Node('c', [b])
a.set_links([b])
b.set_links([a, c])

```

June 2, 2008

An Introduction to Python

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

22

---

---

---

---

---

---

---

---



## functional programming



**filter(), map(), reduce(), lambda, list comprehension**

```

>>> def even(x): return x % 2 == 0
>>> def odd(x): return not even(x)
>>> r = [random.randint(1, 10) for i in range(1000)]
>>> set(filter(even, r)) # set([8, 2, 4, 10, 6])
>>> set(filter(odd, r)) # set([1, 3, 9, 5, 7])
>>> map(float, set(filter(odd, r))) # [1.0, 3.0, 9.0, 5.0, 7.0]
>>> [float(x) for x in set (filter(odd, r))]
>>> [float(x) for x in set ([x for x in r if odd(x)])]

>>> def product(x, y): return x * y
>>> reduce(product, range(1, 6)) # 1*2*3*4*5 = 120
>>> reduce(lambda x, y: x * y, range(1, 6))

```

June 2, 2008

An Introduction to Python

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

23

---

---

---

---

---

---

---

---



## file operation



- `f = open( <filename>, ['r', 'a', 'w', ...] )`
- `f.read()`
- `f.readlines()`
- `f.write(s)`
- `f.seek(n)`
- `f.tell()`
- `f.close()`
- ...
- `help(file)`

June 2, 2008

An Introduction to Python

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

24

---

---

---

---

---

---

---

---



exceptions



- **built-in exceptions:** Exception, IOError, KeyError, MemoryError, RuntimeError, SystemExit, ...
- **user-defined exceptions**
- **raising exceptions:** raise <ExceptionType>
- **handling exceptions:**

```

try:
 ... # do something that may raise an exception
except <>:
 ... # encountering an exception, do this
else:
 ... # do this if an exception did not occur
finally:
 ... # do this whether an exception has occurred

```

June 2, 2008

An Introduction to Python

25

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---



recap on presented topics



- Python – a dynamic computer language
- built-in types: int, float, bool, complex, oct, hex, big numbers, string, list, tuple, set, dictionary
- operations: bitwise, binary, unary, boolean, conversion
- control flow statements:
  - if ... elif ... else, continue, break, pass, for, while
- frequently used functions: dir(), help(), range(), print()
- interactive programmable calculator
- programming paradigms
  - procedural: def func(): return
  - object-oriented: class, inheritance, self, '.' notation
  - functional: filter(), map(), reduce(), lambda
- file operations: open(), read(), write(), close(), ...
- exceptions: try ... except ... else ... finally

June 2, 2008

An Introduction to Python

26

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---



next topics



- modules – creating and importing
  - keywords
  - cryptography
  - Internet & web services
  - markup language processing
  - system services & networking
- scientific computing
- graphical user interface
- extension and embedding
- development environment

June 2, 2008

An Introduction to Python

27

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---


**modules ...**


```

import random
n = random.randint(1, 10) #[1, 10]
from random import *
n = randint(1,10)
from random import uniform
n = uniform(1,10) #[1.0.. 10.0]
import random as rand
n = rand.seed(10.0)

```

June 2, 2008

An Introduction to Python

28

TECHNOLOGY DRIVEN
WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---


**the “builtins” module**


- automatically imported
- to see a list builtin objects:  

```
>>> dir(__builtins__)
```

```
['ArithmeticError', ..., 'zip']
```
- to see their descriptions:  

```
>>> help(__builtins__)
```

```
...lots of output...
```

June 2, 2008

An Introduction to Python

29

TECHNOLOGY DRIVEN
WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---


**modules & main applications**


```

importing module myutils
__name__ == myutils
import myutils as u
p = u.fact(10)
q = u.search('xyz')
o = u.MyClass()
n = u.MAXNUM
...

running myutils application
python myutils.py
__name__ == __main__

#file: "myutils.py"
MAXNUM = 20
def fact(n): ...
def search(x): ...
class MyClass: ...

def test(): ...

#main application
if __name__ ==
 '__main__':
 test()

```

June 2, 2008

An Introduction to Python

30

TECHNOLOGY DRIVEN
WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---





key words



```

>>> from keyword import kwlist
>>> for kw in kwlist:
 print kw,
 and as assert break class continue
 def del elif else except exec finally for
 from global if import in is lambda not
 or pass print raise return try while
 with yield

```

June 2, 2008

An Introduction to Python

31

TECHNOLOGY DRIVEN WARFIGHTER FOCUSED

---

---

---

---

---

---

---

---



cryptographic modules ...



sha, md5, hashlib, hmac

```

>>> import sha, md5, hmac
>>> sha.new('abc-xyz').hexdigest()
'55e6a20533a2b7c99761d0874b48a0413d164337'
>>> md5.new('abc-xyz').hexdigest()
'096252a7b87b04f2b7928c04d5a40174'
>>> key = 'shared_secret_key'
>>> text = 'abc-xyz'
>>> hash_algorithm = hashlib.sha256
>>> hmac.new(key, text, hash_algorithm).hexdigest()
'4b23a5d8f806332180acffc57d986108cb0e7d1678098
5f237e1512d39a43a63'

```

June 2, 2008

An Introduction to Python

32

TECHNOLOGY DRIVEN WARFIGHTER FOCUSED

---

---

---

---

---

---

---

---



... cryptographic modules



```

>>> import hashlib #(standard library)
>>> hashlib.sha1('abc-xyz').hexdigest()
'55e6a20533a2b7c99761d0874b48a0413d164
337'
>>> hashlib.sha512('abc-xyz').hexdigest()
'd311bb080815ae77e0cae3706fb94c8059c69
0286cd566cf2be98a21e7659d5f7aa37590d
a94f082e4f37b60c46e713eab187a93f59632
3132fd3e493f258013'
>>> hashlib.md5('abc-xyz').hexdigest()
'096252a7b87b04f2b7928c04d5a40174'

```

June 2, 2008

An Introduction to Python

33

TECHNOLOGY DRIVEN WARFIGHTER FOCUSED

---

---

---

---

---

---

---

---


Internet & web services


```

>>> import webbrowser as w
>>> w.open('http://www.yahoo.com')
>>> import urllib as u
>>> f = u.urlopen(url)
>>> x = f.read()
>>> f.close()
>>> u.urlretrieve(url, filename='xyz.htm')

```

Other libraries: ftp, http, pop, smtp, ...

June 2, 2008

An Introduction to Python

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

34

---

---

---

---

---

---

---

---


markup languages


- HTML parsers
  - HTML
  - XHTML
- XML parsers
  - DOM
  - SAX
  - Expat
- SGML parsers

June 2, 2008

An Introduction to Python

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

35

---

---

---

---

---

---

---

---


system services


os, sys, threading, socket,  
interprocess communications (IPC), ...

```

os.getcwd(), os.getenv('HOME'), ...
os.path.basename(p), os.path.isfile(f), ...
sys.stderr, sys.stdout, sys.stdin, ...
threading.Thread(): run(), isAlive(), ...
socket(): ntohl(n), gethostname(), ...
Popen (IPC): communicate(), poll(), wait(), ...

```

June 2, 2008

An Introduction to Python

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

36

---

---

---

---

---

---

---

---


**extension and embedding**


- extension – *calling C/C++ functions from Python programs*
- embedding – *calling Python functions from C/C++ programs*
- open-source tools:
  - SWIG *Simplified Wrapper & Interface Generator*
  - Boost.Python *C++ and Python interoperability*
  - Pyrex *Mixing Python & C to generate extensions*
  - f2py *Fortran to Python interface generator*
  - ...
  - <http://wiki.python.org/moin/IntegratingPythonWithOtherLanguages>

June 2, 2008
An Introduction to Python
37
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---


**scientific computing & visualization**


- math – standard library
- add-on:
  - SAGE – math
  - SymPy – symbolic math
  - PyChem – multivariate analysis
  - NumPy – numerical Python
  - SciPy – scientific Python
  - PyOpenGL – binding to OpenGL
  - VTK – visual tool kit
  - ...

June 2, 2008
An Introduction to Python
38
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---


**graphical user interface**


- Tkinter – the standard GUI of Python
- other tool kits:
  - wxPython
  - pyKDE
  - pyGtk
  - PyQt
  - ...

June 2, 2008
An Introduction to Python
39
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

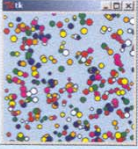


**A Tkinter application**

```

from Tkinter import *
from random import randint, choice
SIZE = 200
canvas = Canvas(Tk(), w=SIZE, height=SIZE, bg='lightblue')
colors=['green','yellow','orange','red','magenta','blue','white']
for i in range(300):
 x, y = randint(0, SIZE), randint(0, SIZE)
 d = randint(5, 10)
 canvas.create_oval(x, y, x+d, y+d, fill=choice(colors))
canvas.pack()
canvas.mainloop()

```



June 2, 2008 An Introduction to Python 40  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

**Turtle graphics**

```

import turtle
colors=['green','yellow','orange','red','magenta','blue','lightblue']
ncolors = len(colors)
w = h = 220
PEN_WIDTH = 20
turtle.setup(width=w, height=h)
turtle.title('Rainbow Colors by BQN')
pen = turtle.Pen()
pen.up()
x, y = pen.position()
pen.goto(x, (PEN_WIDTH - h)/2)
pen.down()
pen.width(PEN_WIDTH)
for i in range(10, 0, -1):
 pen.color(colors[i % ncolors])
 pen.circle(PEN_WIDTH * i / 2)
turtle.done()

```



June 2, 2008 An Introduction to Python 41  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

**development tools**

- Freely available
  - IDLE -- comes with Python
  - IPython -- comes with SciPy
  - DrPython -- requires wxPython
  - Eclipse+PyDEV -- requires Java
  - ...
- Commercial IDE
  - Wing IDE
  - Komodo IDE
  - ...

June 2, 2008 An Introduction to Python 42  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

RDECOM

topics presented

dynamic programming language

interactive programmable calculator

programming paradigms

markup language processing

graphical user interface

development environments

web services

system services

networking

scientific computing

cryptography

extension and embedding

python

June 2, 2008

An Introduction to Python

43

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

RDECOM

<http://www.python.org>

- tutorial for the current version (2.5.2)
- library reference
- language reference
- news & announcements
- links to other relevant sites
- binary and source files for downloading
  - Windows™
  - Linux™
  - ...

June 2, 2008

An Introduction to Python

44

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

RDECOM

**Thanks**

**bnguyen@arl.army.mil**

June 2, 2008

An Introduction to Python

45

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---



| No. of<br>Copies     | Organization                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1<br>(Elect<br>Copy) | ADMNSTR<br>DEFNS TECHL INFO CTR<br>ATTN DTIC OCP<br>8725 JOHN J KINGMAN RD STE 0944<br>FT BELVOIR VA 22060-6218                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 1                    | US ARMY RSRCH LAB<br>ATTN AMSRD ARL CI OK TP TECHL LIB T LANDFRIED<br>BLDG 4600<br>ABERDEEN PROVING GROUND MD 21005-5066                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 22                   | US ARMY RSRCH LAB<br>ATTN AMSRD AR CI NT D GWYN<br>ATTN AMSRD ARL CI A TOTH<br>ATTN AMSRD ARL CI J GOWENS<br>ATTN AMSRD ARL CI N G RACINE<br>ATTN AMSRD ARL CI NT B NGUYEN (5 HC, 1 ELECT)<br>ATTN AMSRD ARL CI NT B RIVERA<br>ATTN AMSRD ARL CI NT G CIRINCIONE<br>ATTN AMSRD ARL CI NT K MARCUS<br>ATTN AMSRD ARL CI NT L M SCOTT<br>ATTN AMSRD ARL CI NT N IVANIC<br>ATTN AMSRD ARL CI NT R HARDY<br>ATTN AMSRD ARL CI NT R PRESSLEY<br>ATTN AMSRD ARL CI OK T TECHL PUB<br>ATTN AMSRD ARL CI OK TL TECHL LIB<br>ATTN AMSRD ARL CI R NAMBURU<br>ATTN AMSRD ARL SE R P AMIRTHARAJ<br>ATTN IMNE ALC IMS MAIL & RECORDS MGMT<br>ADELPHI MD 20783-1197 |
| TOTAL                | 24 (22 HC and 2 Elect)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |