Python Tutorial

4/17

環境架設

安裝Python

Windows: Download

Linux & Mac: 內建

Python 簡介

特色

優點:

直譯

缺點:

直譯

特色

1. 不用分號
2. 不用宣告型態
3. 條件判斷式不用小括號
4. 有許多Open source library

使用

1. python
2. python xxx.py

第一支Python程式

```
>>>print "Hello World"
```

變數宣告

```
>>> var_int = 1

>>> var_float = 1.0

>>> var_str = "This is a string"

>>> var_bool = True
```

印出變數&取得變數型態

```
>>> print var_int

>>> print type(var_int)
```

變數運算

```
>>> print 3+4

>>> print 3/4

>>> print 3/4.0

>>> print "String One "+"String Two"

>>> print "String" + 3

>>> print "String" + str(3)
```

變數型態轉換

```
>>> print type(3.0)

>>> print type(str(3.0))



>>> print type("1.0")

>>> print type(float("1.0"))

>>> print type(eval("1.0"))
```

多重變數

一次宣告多個變數

```
>>> var_3,var_4 = 3,4
```

Tuple

```
>>> var_tuple = (1,2,3)
>>> print var_tuple[0],var_tuple[0:2]
```

Add:

```
>>> tuple1 = tuple1+tuple2
```

Append:

```
>>> tuple1 = tuple1 + (1,) + (1,2,3,4)
```

Check:

```
>>>print 3 in (1,2,3)
```

List

```
>>> var_list = [1,2,3]
>>> print var_list[0],var_list[1:3]
>>> range(10)  #[0,1,2,3,4,5,6,7,8,9]
>>> range(6,10) #[6,7,8,9]
```

Add:

```
>>> list1 = list1+list2
```

Append:

```
>>> var_list.append(3)
```

Update:

```
>>> var_list[0] = 2
```

Dict

```
>>> var_dict = {'key1':'value1_str',''key2':3}
>>> print var_dict['key1']
```

Add & Update:

```
>>> var_dict['key3'] = 20
```

list all:

```
>>> var_dict.keys()
>>> var_dict.values()
>>> var_dict.items()
```

Set

```
>>> var_set = {1,2,3,4}
```

Add:

```
>>> var_set.add(5)
>>> var_set.update({3,4,5})
```

- Set內的element皆為唯一

控制流程指令

if...elif...else

```
if a==b:
        print "a=b"
elif a<b:
        print "a<b"
else:
        print "a>b"

if (a<5 and a>3) or a>80:
        …
```

for

```
for i in [1,2,3,4,5]:
	print i #1 2 3 4 5

for j in range(10):
	print i #0 1 2 3 4 5 6 7 8 9

for index in range(len(var_list)):
	print var_list[index]
	break

for key in var_dict.keys():
	print var_dict[key]
	continue
```

switch...case...

沒有

I/O

Output

```
print "output string"

print var

print var1,var2,var3
```

Input

```
name = raw_input()

name = raw_input("What’s your name?")
```

File Read

Flow : open -> read -> close

Read:

```
f = open(filename)  #type:file
data = f.read() #type:str , 完整讀取
        or
for i in f:
        print i #type:str , 一次讀取一行
f.close()
```

File Write

Flow : open(w) -> write -> close

Write:

```
f = open(filename,"w")
f.write("String\n")
f.write(str(data))
f.close()
```

Function & Object

Function

不須參數:

```
def function_name():
        xxx
        return abc # return非必要,型態不限定
```

需要參數:

```
def function_name(arg,arg_optional=default_value):
        xxx
        return value1,value2...#可回傳多個結果,回傳後為tuple , 或是直接用多個變數去接
```

Function

無限參數:

```
def function_name(*args,**kwargs):
    #讀取方式,args -> list , kwargs -> dict
    for arg in args:
        xxx
    for key in kwargs:
        kwargs[key]...
```

Object

```
class object_name:

        var_global = 123    #global variable

def __init__(self,arg1,arg2): #初始設定,非必要,在object被產生時自動執行
        print var_global    #can not found
        print self.var_global
        self.xxx = 1

def function1(self):
        print self.xxx
```

Use Object

```
from object_file import object_name
object = object_name(arg1,arg2)
print object.var_global
object.function1()
```

常用內建function

資料處理

1. len() #計算list 或dict等類型變數之長度(資料數量)
2. range() #生成n~m的連續數列list
3. sorted() #將list內的elements進行排序
4. reversed() #將list倒轉過來

字串處理

1. var_str.split(x) #用字串x 將var_str切割，輸出為list
2. var_str.replace("x","y") #將var_str中的所有x替換為y
3. var_str.strip() #將var_str自左或自右有出現的空白或換行符號移除 ,直到遇到文字
4. var_str.lstrip() #與上述功能相同,但僅自左開始
5. var_str.rstrip() #與上述功能相同,但僅自右開始

Debug

try...except...

- 錯誤排除功能

```
try:
    xxx #may cause error
except:
    print "error"

try:
    xxx #may cause IO error
    yyy #may cause index error
except IndexError:
    print "index error"
except IOError as e:
    print e.strerror
```

常見error message

1. ImportError: No module named …
2. IndexError: list index out of range
3. TypeError
4. ValueError
5. I/OError

Import

How to import?

1. import abc

```
import abc
abc.var_str
abc.function1()
```

2. from abc import var_str,function1
3. from abc import *

```
from abc import *
var_str
function1()
```

- import abc as a

```
a.var_str #abc.var_str
a.function1()
```

內建library

sys,os

- 與系統或是設定相關的參數

import sys,os
常用:
sys.path.append(path) #增加系統參照路徑(暫時)
os.getcwd() # 取得當前路徑
os.chdir(path) #切換當前路徑
sys.stdout.write() #最直接的print
sys.argv #程式啟動時給予的額外參數,格式為list

re

- 字串處理-正則表示式(regex)

ex:

```
import re
account = re.sub("xyz123@gmail.com",r"([^@]+)")
```

json

- 與json格式相關的資料處裡

import json

json.loads(json_str) , json.dumps(json_object) #str轉json或json轉str

json.load(json_filename) #讀取json檔為json object

json.dump(json_str,json_filename)#將json string存成json檔

外部library

numpy

- 較完整的python數值運算library
- 可做矩陣運算

urllib

- 網路存取相關的library

ex:

```
import urllib2
response = urllib2.urlopen('https://www.google.com.tw/')
html = response.read()
```

flask

- 最快速的python架站套件

ex:

```
from flask import Flask
        app = Flask(__name__)
        @app.route("/")
        def hello():
                return "Hello World!"  #若回傳html content,則顯示會與一般網頁相同
        if __name__ == "__main__":
                app.run() #常用參數   app.run(host="0.0.0.0",port=9000,debug=True)
```