Python Basic

for Raspberry Pi

Rev. R610

이세우 (dltpdn@gmail.com)

세부목차

Introduction

Python

- Guido Van Rossum(귀도 반 로섬)
  - 네델란드 암스테르담, 1989 python 개발
  - 구글(전), 드롭박스 근무
- 영국 BBC 코미디, “Monty Python’s Flying Circus”
- MIT 1학년에게 LISP 대신 Python 가르치다
- 2가지 버전 : 2.x, 3.x
  - 여기서는 2.7.x 만 다룬다
- 대화형 인터프리터 언어
- 플랫폼 독립적
- 동적 테이타 타입
- 빠른 개발 목적
- 간단하고 쉬운 문법(?)
- 객체지향언어
- 다양한 내장 객체 자료형
- Garbage Collection

Introduction

❖ API Document

- Standard API : https://docs.python.org/2/library/
  - Built in Function : https://docs.python.org/2/library/functions.html
  - String Method : https://docs.python.org/2/library/stdtypes.html#string-methods
- Global Modules : https://docs.python.org/2/py-modindex.html
- External Modules : https://pypi.python.org/pypi

❖ Help system

- 대화형 콘솔에서 help()를 이용하면 문서를 볼 수 있다
  - 한줄 아래 : 엔터, 아래 방향키, j
  - 한줄 위로 : 위 방향키, k
  - 한 페이지 아래 : 스페이스키, f
  - 한 페이지 위로 : b
  - 빠져나올때 : q

```
>>> help(range)
>>> help(‘’.split)
>>> help(‘’.join)
>>>import random
>>>help(random)
```

Introduction

❖ 개발환경 IDE

- Eclipse + Pydev :
  - 속도가 느리다
  - 다른 플러그인과 함께 사용가능
- PyCharm
  - 속도 빠르다
  - 사용 높음
  - 유료와 무료
- 그 밖에 개발용 에디터
  - Sublime
  - Notepad++
  - brakets

세부목차

Syntax

❖ Syntax

- 대소문자 구분
- 괄호 대신 들여 쓰기
- pass
  - 함수나 조건문의 내용이 없을 경우
- 주석
  - # 한줄 주석
  - ‘’’(홑따옴표 * 3), “””(쌍따옴표*3) : 여러줄 주석
- 문장의 끝
  - 세미콜론(;) 없이 줄바꿈기호로 대신
  - 문장의 끝 의미 없이 줄바꿈 하고자 할때는 ₩(역 슬레쉬)
- 한글 지원
  - # -*- coding:utf-8 -*-
    - 파일의 시작에 표시
  - u’한글’
    - 문자열의 앞에 u 표시

```
# -*- coding:utf-8 -*-
'''
Created on Dec 6, 2015
@author: xcoda
'''
'''
여러줄 주석
'''

""""""
 여러줄 주석
"""""
#한줄 주석
```

Syntax

❖ Data type

- int, float, bool(True, False), str, None
- list[1,2,3], tuple(1,2,3), dict{1:'a', 2:'b'}, set{1,2,3}
- type()
  - 동적으로 타입을 확인

❖ 변수

- 선언문 없음
- 값의 할당에 따라 데이타 타입 동적 바인딩
- 모든 변수는 객체
- 이름
  - 소문자_소문자
- id()
  - UID(주소번지) 확인
- int.bit_length()
  - 값의 표현에 사용한 비트 길이

```
print type(1)
print type(3.14)
print type(True)
print type(False)
print type('a')
print type('abcd')
print type(None)

list = ['one', 'two', 'three']
tuple = ('one', 'two', 'trhee')
set = set(['one', 'two', 'trhee'])
dict = {'a': 'one', 'b':'two', 'c': 'three'}

print type(list) #list
print type(tuple) #tuple
print type(set) #set
print type(dict)  #dictionary
```

Syntax

콘솔입출력

- input(prompt)
  - 사용자 입력을 즉시 평가
- raw_input(prompt)
  - 사용자 입력을 문자열로 반환
- print exp1, exp2…
  - 콘솔 출력
  - 튜플 형식

문자열

- ' ', " "
  - 홑따옴표, 겹따옴표
- == 연산
  - id가 달라도 내용이 같으면 True
- len(str)
  - 문자열의 길이
- 포맷 문자
  - %s: 문자열, %d:정수, %f:실수
- 인덱싱 : 튜플
  - str[0], str[0:4]

```
input1 = raw_input() #abc 입력
input2 = raw_input() #abc 입력


print input1, id(input1) # abc 4382273632
print input2, id(input2) #abc 4382273680
print input1 == input2  #True

msg =  'my age is %d' %25
print msg
```

Syntax

- **연산자**
  - ‘abc’ + ‘def’
    - ‘abcdef’
  - ‘abc’ + 2
    - 오류 발생
  - ‘abc’ * 3
    - ‘abcabcabc’
  - 7 / 4.0
    - 1.75
  - 7 // 4.0 : 소수점 아래 버림 연산
    - 1.0
- **논리 연산**
  - a == b, a != b, a > b, a >=b, a < b, a<=b
  - a and b
  - a or b
  - not a
  - a in b
  - a not in b

Syntax

❖ 형 변환

- 정수로
  - int(“10”)
  - int(True)
- 실수로
  - float(‘3.14’)
  - float(123)
- 문자열로
  - str(10)
  - ‘%s’ %10

Syntax

❖ **조건문**

- If condition :
      pass
- If condition :
      pass
  else :
      pass
- If condition1 :
      pass
  elif conditon2 :
      pass
  else :
      pass

```
if True:
   print "수행됨!”
if False:
   print "수행됨2!”

isWait = False
if isWait:
   print ”wait"
num=10
if num%2==0:
   print "{} is even".format(num)
else:
   print "{} is odd".format(num)

if num>0:
   print "{} is positive".format(num)
elif num<0:
   print "{} is negative".format(num)
else:
   print "{} is zero".format(num)

isMan = True
result = "Man" if isMan else 'Women'
print result
```

Syntax

❖ **반복문**
- for x in range(0,10) :
  pass

```
print range(10)
print range(5,-1)

names = ['aaa', 'bbb', 'ccc']
names.append('ddd')
for i in range(len(names)):
   print i, 'th :', names[i]

print '-'*80

print range(0,10, 1)
print range(10,0, -1)
print range(10,-1, -1)
for i in range(len(names)-1, -1, -1):
   print i, names[i]


friends = [{'name':'aaa', 'isMan' : True, }, {'name':'bbb',
'isMan':False}, {'name':'ccc', 'isMan':False}, {'name':'ddd',
'isMan':True}]
for i in range(len(friends)-1, -1, -1):
   if(friends[i]['isMan'] == False):
      del friends[i]
      #friends.remove(friends[i])
      print i, 'removed'
```

Syntax

- **반복문**
  - while True:
        pass

```
count1 = 0
while count1 < 10:
    print count1
    count1 += 1



names = ['aaa', 'bbb', 'ccc']
names.append('ddd')
names.append('eee')

idx = 0
while idx < len(names):
    print idx, ':' , names[idx]
    idx+= 1


idx = len(names)-1
while idx >= 0:
    print idx, ":", names[idx]
    idx -= 1
```

연습문제

❖ Double dice

- 주사위 2개를 10번 던져
- 두 눈의 합 출력
  - 7
  - 11
  - 같은 눈
- 난수 발생
  - import random
  - random.randint(start, end)

```
<<출력 예시>>

1: 6, 3
2: 6, 1
seven!
3: 4, 4
double!
4: 5, 1
5: 5, 3
6: 1, 6
seven!
7: 4, 6
8: 4, 6
9: 2, 2
double!
10: 6, 6
double!
```

연습문제

❖ **Double dice**

```
import random

for x in range(1,11):
    n1 = random.randint(1, 6)
    n2 = random.randint(1, 6)
    print '%d: %d, %d' %(x,n1, n2)
    if n1 + n2 == 7:
        print("seven!")
    elif n1 + n2 == 11:
        print ("eleven")
    if n1 == n2:
        print("double!")
```

세부목차

자료구조

❖ List

- list()
- list = [1,2,3,4]
- list[0]
  - 1
- list[1] = 5
  - [1,5,3,4]
- list.append(10)
  - [1,5,3,4,10]
- list.remove(5)
  - [1,3,4,10]
- del list[10]
  - [1,3,4]
- list.pop() #4
  - [1,3]
- Range(len(list))

```
friends = ['cat', 'dog', 'elephant', 'snake', 'flog' ]
for item in friends:
   print item


friends = ['cat', 'dog', 'elephant', 'snake', 'flog' ]
for i in range(len(friends)):
   print i, ':', friends[i]
```

자료구조

❖ Tuple

- 읽기 전용 List
- tupe()
- Tup = 1,2,3,4
- Tup = (1,2,3,4)
- Tup[1]
    - 2
- Tup[2] = 5
    - Error

```
tuple1 = ("one","two","three")
printtuple1[0]
#tuple1[0]="four"
#tuple1.append("four")

list1 = list(tuple1)
printlist1

result = list1 == tuple1
print"list1 == tuple1 : ", result

tuple2 = tuple(list1)
printtuple2

tuple3 = (10,)
printtype(tuple3)

tuple4 =10,20,30
print"tuple4 : ",tuple4

num1, num2, num3=tuple4
printnum1, num2, num3

first="girl"
second="boy”
second, first=first, second

printfirst, second
```

자료구조

❖ Set

- 순서가 없음
- 중복 허용 없음
- Set(list)
- S = {1,2,3,4,5}
- S.add(6)
- S.union(s2)
- S.intersection(s2)
- S1 - s2
- S.discard(2)
- Set.clear()

```
set3={"kim","lee"}
list1 = ["park","cho","lee"]
tuple1 = ("one","two")
set3.update(list1)
print set3
set3.update(tuple1)
print set3
set3.discard("park")
print set3
set3.discard(”zzz”)
set3.clear()
print set3

for item in set1:
   print item

list3 = [10,20,30,10,10,30,40,50,50]
set4 = set(list3)
print"set4 : ",set4
list4 = list(set4)
print"list4 : ",list4
```

```
set1 = {10,20,30,40,50}
print"len(set1) :", len(set1)
set1.add(60)
set1.add(70)
printset1
set2 = {60,70,80,90,100}

resultSet = set1.union(set2)
print"set1 U set2 : ", resultSet
resultSet2 = set1.intersection(set2)
print"set1 n set2 : ", resultSet2
resultSet3 = set1-set2
print"set1 - set2 : ", resultSet3
```

자료구조

❖ Dictionary

- Key : value
- Dict()
- Dic = {'a' : 1, 'b':2}
- Dic['a']
  - 1
- Dic['b'] = 5
  - {'a':1, 'b': 5}
- Dic.keys()
- Dic.values()
- Dic.items()
- Dic.clear()

```
dict1 =
{'num':1,'name':'kim','isMan':True}

printtype(dict1), dict1, len(dict1)

dict1['num'] =999
print'after editing:', dict1


deldict1['num']
printdict1


dict1.clear()
printdict1

dict1['new'] =123
printdict1
```

```
dict2 =
{'car':'bmw','house':'aprtment','pho
ne':'android'}
printdict2.keys()
printdict2.values()
printdict2.items()
printdict2.items()[0][0]


forkeyindict2:
   value =  dict2[key]
  printkey, ':', value
print'-------------------'

forkeyindict2.keys():
   value =  dict2[key]
  printkey, ':', value
```

세부목차

함수

❖ Function

- def name():
- Def name(a, b):
- Def name(*args):
  - Tuple type
- Def name(num=0)
- Def name(**kwargs)
  - Dict type

```
def test1():
    pass
test1()

def test2():
    print "test2"
test2()

def test3(a):
    print "test3",a
test3("abc")
test3(999)

def test4(arg1, arg2):
    print "arg1:",arg1
    print "arg2:",arg2
test4("one", "two")
result1 = test4("three", "four")
print "result1:",result1
```

```
def test5():
    print "test5()"
    return

def test6():
    print "test6()"
    return None
result2 = test5()
result3 = test6()
print "result2:",result2
print "result3:",result3
```

함수

❖ Function

```
def getSum(num1, num2):
    result= num1+num2
    return result

print getSum(10, 20)
f1 = getSum
print f1(1,2)

def showSum(num1, num2):
    result = num1+num2
    print "showSum:",result

showSum(100, 200)

def test7(*args):
    print args

test7()
test7(10)
test7("one","two","three")
test7(10,20,30,40,50)
```

```
def test8(arg1, *args):
    print "arg1:",arg1
    print "args:",args

test8("aaa")
test8("aaa","bbb")
test8("aaa","bbb","ccc")

def test9(num=0):
    print "num:",num

test9()
test9(999)

formatStr = "No:{} name:{} addr:{}".format(1, "lee", "seoul")
print formatStr

def test10(num=0, name="Lee", addr="Seoul"):
    result="번호:{} 이름:{} 주소:{}".format(num, name, addr)
    print result
```

함수

❖ Function

```
def test11(**kwargs):
    print type(kwargs)
    print "kwargs : ",kwargs

test11()
test11(num=1)
test11(num=2,name="Park",addr="Ilsan")

def test12(arg1, *args, **kwargs):
    print "arg1:",arg1
    print "args:",args
    print "kwargs",kwargs

test12(999)
test12(999,"one","two","three")
test12(999,"one","two","three",num=3,name="monkey",addr="seoul")
```

연습문제

- **Hanman**
  - 단어 알아 맞히기
  - 주어진 단어 6개
  - 사용자가 예상되는 한글자 또는 단어 입력
  - --- 과 같이 표시
  - 맞으면 맞는 부분만 알파벳 표시
  - 14번 기회

<출력 예시>

```
---
Lives Remaning: 14
Guess a letter or whole word?d
d—
Lives Remaning: 13
Guess a letter or whole word?z
d—
Lives Remaning: 12
Guess a letter or whole word?g
d-g
Lives Remaning: 13
Guess a letter or whole word?o
You win! Well Done!
```

연습문제

❖ Hangman

```
import random

words = ['chicken', 'dog', 'cat', 'mouse', 'frog']
lives_remaining = 14
guessed_letters = ''

def pick_a_word():
    return random.choice(words)


def play():
    word = pick_a_word()
    while True:
        guess = get_guess(word)
        if process_guess(guess, word):
            print('You win! Well Done!')
            break
        if lives_remaining ==0:
            print('You are Hung!')
            print('The word was: ' + word)
            break
```

연습문제

❖ Hangman

```
def get_guess(word):
    print_word_with_blanks(word)
    print 'Lives Remaning:', str(lives_remaining)
    guess = raw_input('Guess a letter or whole word?')
    return guess

def print_word_with_blanks(word):
    display_word = ''
    for letter in word:

        if guessed_letters.find(letter) > -1:
            display_word = display_word + letter
        else:
            display_word = display_word +'-'
    print display_word #, guessed_letters, word

def process_guess(guess, word):
    if len(guess) > 1:
        return whole_word_guess(guess, word)
    else:
        return single_letter_guess(guess,  word)
```

연습문제

❖ Hangman

```
def whole_word_guess(guess, word):
   global lives_remaining
   if guess == word:
      return True
   else :
      lives_remaining  -= 1
      return False

def single_letter_guess(guess,  word):
   global guessed_letters
   global lives_remaining
   if word.find(guess) == -1:
      lives_remaining  -= 1
   guessed_letters = guessed_letters + guess
   if all_letters_guessed(word):
      return True
   return False

def all_letters_guessed(word):
   for letter in word:
      if guessed_letters.find(letter) == -1:
         return False
   return True
```

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

세부목차

Module

❖ Module, Pacakge

- import random
  - random.radnint(1,6)
- imoprt random as r
  - r. radnint(1,6)
- from random import randint
  - randint(1,6)
- from random import *
  - Randint(1,6)

Module

❖ custom module

```
#mymodule1.py

def function1():
   print 'this is function 1'
```

```
#module_test.py

import mymodule1
import mymodule1 as m1
from mymodule1 import function1

mymodule1.function1()
m1.function1()
function1()
```

Module

❖ **custom module**

```
#mymodule1.py

def function1():
   print 'this is function 1'
```

```
#module_test.py

import mymodule1
import mymodule1 as m1
from mymodule1 import function1

mymodule1.function1()
m1.function1()
function1()
```

Module

❖ custom package

- module들을 포함 하는 디렉토리
- __init__.py 파일 포함

```
#mypack/mymodule2.py

def function2():
   print 'this is function 2'
```

```
#module_test.py

import mypack.mymodule2
from mypack import mymodule2
from mypack.mymodule2 import *

mypack.mymodule2.function2()  #this is function 2
mymodule2.function2() #this is function 2
function2() #this is function 2
```

Module

❖ **custom package**

- 중복 되는 이름이 있으면 나중에 import한 것이 선택

```
#mypack/mymodule2.py

def function1():
   print 'this is function 1 in mymodule2'

def function2():
   print 'this is function 2'
```

```
#module_test.py
from mymodule1 import function1
from mypack.mymodule2 import function1

funciton1()   # this is funciton 1 in mypack
```

```
#module_test.py
from mypack.mymodule2 import function1
from mymodule1 import function1

funciton1()   # this is funciton 1
```

Module

❖ **custom package**

- 표준 API와 동일한 Custom 모듈

```
#random.py

def randint(a, b):
   print 'this is custome random.randint(%d,%d)' % (a,b)
```

```
#module_test.py


import random

num = random.randint(1, 6)  #this is custome random.randint(1,6)
print num    #None
```

세부목차

Class

❖ Class

```
class Car:
    pass

myCar = Car()
print type(myCar)
```

```
class Car:
    name = ’Sonanta'

    def drive(self):
        print ’run :’, self.name

myCar = Car()
print type(myCar)
print myCar
myCar.drive()
```

Class

❖ Inheritance

```
classCar(object):
    engine =None

   def__init__(self, engine):
        self.engine = engine

   defdrive(self):
       ifself.engine ==None:
           print"can't drive caunsed by no engine"
       else:
           print'driving..'


classSuperCar(Car):
   def__init__(self, engine):
        super(SuperCar, self).__init__(engine)
       #Car.__init__(self, engine)

   defdriveFast(self):
       print'driving fast very much...'
```

세부목차

Exception

❖ 예외처리

- try :
  - 예외가 예상 되는 statements
- except Exception:
  - 특정한 예외가 발생하면 실행
- except :
  - 예외 종류에 상관 없이 발생하면 실행
- else :
  - 예외가 발생하지 않으면 실행
- finally
  - 예외 발생 여부와 상관 없이 실행

❖ 예외 발생

- rase Exception

❖ 예외 정보

- sys.excinfo()
  - 발생한 예외 종류, 값, trackback

```
try:
   print  num1, '/',num2 , '=' , num1/num2
   print 'after try'
except ZeroDivisionError as zde:
   print " can't divide by zero. ", zde
except Exception as ex:
   print ex
else:
   print 'no Error'
finally:
   print 'finally'
print 'successfully terminated.'
```

Exception

❖ Built-in exceptions

- Exception
  - 모든 예외의 루트 클래스
  - 사용자 정의 예외 클래스 상속 강제 규정 없슴
- StandardError
  - Systemexit를 제외한 모든 내장 예외의 베이스 클래스
- Arithmetic Error
  - OverflowError, ZeroDivisionError, FloatingPointError의 베이스 클래스
- LookupError
  - IndexError, KeyError의 베이스 클래스
- EnvironmentError
  - 외부 발생 예외(IOError, OSError)의 베이스 클래스

Exception

❖ Custom Exception

```
import sys

class MyException(Exception):
   def __init__(self, msg):
      self.msg = msg


def plus(a, b):
   if a >= 0 or b >= 0 :
      return a + b
   else:
      raise MyException('negative parameter')

try:
   print plus(1,2)
   print plus(-1,-2)
except Exception as e:
   print e.msg
   print sys.exc_info()
```

세부목차

File I/O

- **File read**
  - Open(name)
  - f.read()
  - f.close()
- **File write**
  - open(name)
  - f.write(‘content’)
  - f.close()

```
file_name = "file.txt"
#file_name = "no_file.txt"
try:
   f = open(file_name)
   lines = f.read()
   f.close()
   words = lines.splitlines()
   print words
except IOError:
   print 'Can not open the file.'
```

```
file_name = 'newfile.txt'

f = open(file_name, 'w')
f.write('This is file that I made by the python program.')
f.close()
```

File I/O

❖ File read by line

- Open(name)
- f.readline()
- f.close()

```
file_name = 'file.txt'

try:
    f = open(file_name)
    cnt = 0
    while True:
        cnt += 1
        line = f.readline()
        if line == '':
            break
        print "%d : %s" %(cnt, line),
except IOError:
    print 'Can not open the file'
```

세부목차

Thread

Thread

- 하나의 프로세스안에 있는 작은 프로세스
- 병렬처리
- 비동기 처리
- 병렬 프로세스에 비해 메모리 공유 및 제어 용이

3가지 방법

- thread 모듈 (저수준)
  - thread.start_new_thread(fn_name, (x,y,…))
- threading 모듈 (고수준)
  - 직접 생성 , Simple
    - th = threading.Thread(target=fn_name, args=(x,y,…))
    - th.start()
  - 상속 구현, Detail Control
    - class MyThread(threading.Thread)
      - def fun():
    - th = MyThread()
    - th.start()

Thread

❖ Thread 모듈 (저수준)

```
import thread, time

thread_status = [True] * 5

def counter(id, cnt):
    for i in range(cnt):
        print 'id %s --> %s' %(id, i)
    thread_status[id] = False
    print 'Thread %d is dead.' % id

for i in range(5):
    thread.start_new_thread(counter, (i, 5))


#time.sleep(2)
while True in thread_status:
    pass

print 'all threads died, main exiting.'
```

Thread

❖ Threadding 모듈

- 직접 생성

```
import threading


def counter(id, cnt):
   for i in range(cnt):
      print 'id %s --> %s' %(id, i)
   thread_status[id] = False
   print 'Thread %d is dead.' % id

for i in range(5):
   th = threading.Thread(target=counter, args=(i, 5));
   th.start()
   th.join()

print 'all threads died, main exiting.'
```

Thread

❖ Threadding 모듈

- 상속 구현

```
import threading

class MyThread(threading.Thread):
   def __init__(self, cnt):
      threading.Thread.__init__(self)
      self.cnt = cnt

   def run(self):
      for i in range(self.cnt):
         print 'id %s --> %s' %(self.getName(), i)
      print 'Thread %s is dead.' % self.getName()


for i in range(5):
   th = MyThread(5)
   th.start()
   th.join()

print 'all threads died, main exiting.'
```

Thread

❖ Thread 제어

- main 종료시 thread 종료
- thread 종료 시키는 함수가 별도로 없음
- thread의 상태 변수를 통한 작업 유지 여부 결정

```
import threading, time

class MyThread(threading.Thread):
   live = True
   cnt = 0;
   def run(self):
      while True:
         if not self.live:
            print '%s is dead.' % self.getName()
            break;
         print self.getName(), self.cnt
         time.sleep(1)
         self.cnt = self.cnt + 1

   def stop(self):
      self.live = False
```

```
th = MyThread()
th.start()
try:
   for i in range(10) :
      print 'Main',  i
      time.sleep(0.5)
finally:
   print 'Main is dead.'
   th.stop()
```

세부목차

Networking

❖ Socket by server

- soc = socket(AF_INET, SOCK_STREAM)
  - 소켓 스트림 생성
- soc.setsockopt(SOL_SOCKET, SO_REUSEADDR,1)
  - 닫힌 포트를 재사용 하도록 설정
- soc.bind((' ', 1234))
  - 포트 바인딩
- soc.listen(5)
  - 최대 접속수 : 5
- conn, addr = soc.accept()
  - 대기 시작
- conn.recv(1024)
  - 송신
- conn.send('content')
  - 전송
- conn.close()
  - 커넥션 종료
- soc.close()
  - 소켓 종료

Networking

❖ Socket by Client

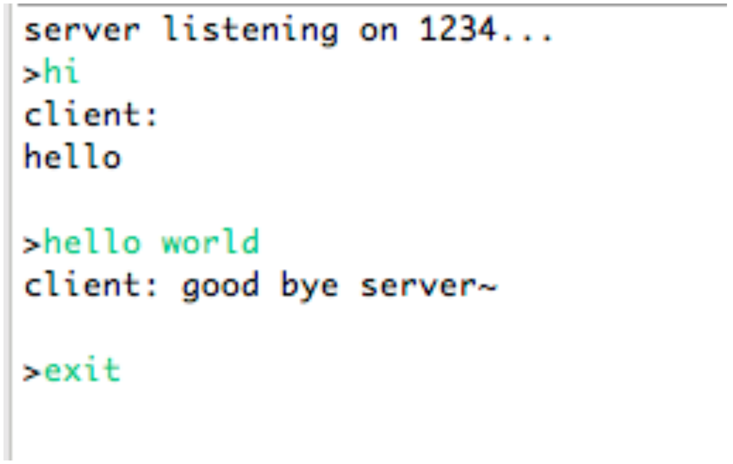
- soc = socket(AF_INET, SOCK_STREAM)
  - 소켓 스트림 생성
- soc.connect((host, port))
- soc.recv(1024)
  - 송신
- soc.send('content')
  - 전송
- soc.close()
  - 소켓 종료

Networking

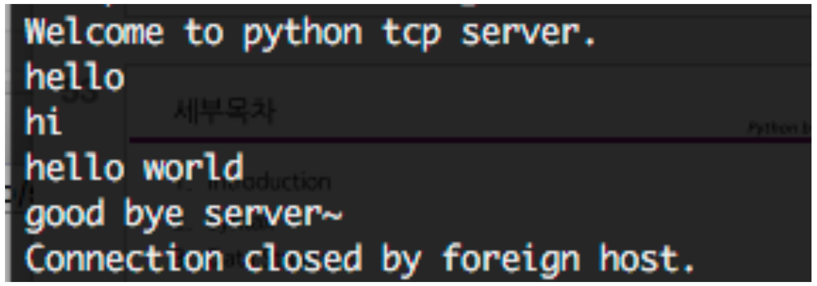
❖ Socket by server

- 서버 실행
- putty(telnet) localhost 1234 접속

```
from socket import *

server = socket(AF_INET, SOCK_STREAM)
server.setsockopt(SOL_SOCKET, SO_REUSEADDR, 1)
server.bind(('', 1234))
server.listen(1)
print "server listening on 1234..."
conn, addr = server.accept()

conn.send('Welcome to python tcp server.')
while True:
   str = raw_input(">")
   if str == "exit":
      break
   conn.send(str+"\n")
   read = conn.recv(1024)
   print 'client:', read

conn.close()
```



```
server listening on 1234...
>hi
client:
hello

>hello world
client: good bye server~

>exit
```



```
Welcome to python tcp server.
hello
hi
hello world
good bye server~
Connection closed by foreign host.
```

Networking

❖ Socket by Client

```
from socket import  *

socket = socket(AF_INET, SOCK_STREAM)
socket.connect( ('', 1234))

read = socket.recv(1024)
print 'server:', read

while True:
    str = raw_input(">")
    if str == "exit":
        break
    read = socket.recv(1024)
    print 'server:', read
    socket.send(str+"\n")

socket.close()
```

Networking

❖ Socket by server using thread

```
from socket import *
import threading
running = True
def recv():
   while running:
      read = conn.recv(1024)
      print 'client:', read
try:
         server = socket(AF_INET, SOCK_STREAM)
         server.setsockopt(SOL_SOCKET, SO_REUSEADDR,
         1)
         server.bind(('', 1234))
         server.listen(1)
         print "server listening on 1234..."
         conn, addr = server.accept()
         th = threading.Thread(target=recv)
         th.start()
         conn.send('Welcome to python tcp server.')
```

```
         while running:
            str = raw_input(">")
            if str == "exit":
               break
            conn.send(str+"\n")

         conn.close()
finally :
         running = False
```

Networking

❖ Socket by Client using thread

```
from socket import *
import threading

running = True
def recv():
   while running:
      read = socket.recv(1024)
      print 'client:', read

try:
   socket = socket(AF_INET, SOCK_STREAM)
   socket.connect(('', 1234))
   th = threading.Thread(target=recv)
   th.start()
   socket.send('Hi! This is a client.')
```

```
   while running:
      str = raw_input(">")
      if str == "exit":
         break
      socket.send(str+"\n")

   socket.close()
finally:
   running = False
```