Python日常小技巧

by 熊崽Kevin

1. 处理好判断条件中的True与False

在Python程序中，条件判断是最常见的代码片段了，而对于条件，有一些值得注意的地方。下面是布尔环境中常见的Ture与False情况：

True	False
True	False
-	None
非空字符串	空字符串
非0整数	整数0
非空容器 len(x) > 0	空容器 len(x) == 0
0, '', [], (), {}, None	其他情况

2. 善用`in`

a) 利用`in`来检查容器内的元素。`in`关键词可以用在`list`、`dict`、`set`、`string`等实现了`__contains__`内建函数的类中。

示例：

```
name = 'Name'

if 'N' in name:
    print 'N in %s' %name
```

b) 尽量用`in`来遍历容器各元素。`in`关键词可以用在`list`、`dict`、`set`、`string`等实现了`__iter__`内建函数的类中。

```
name = ['Jack', 'Paul', 'Tom']

for item in name:
    print item
```

3. 直接交换变量的值，而不使用中间变量

```
a, b = 10, 20
print 'Before a: %s, b: %s' %(a, b) # 10, 20
a, b = b, a
print 'After a: %s, b: %s' %(a, b)  # 20, 10
```

4. 利用`try`来处理容器中得元素访问

在Python中，使用异常检查机制的代价不大（相对Java等其他语言而言），因此不需预先检查容器中对应的key是否存在。

```
a = {'key': 5}

try:
    value = a['key']
except (KeyError, TypeError, ValueError):
    value = None
```

5. 利用`enumerate`函数同时获取列表的索引及值

```
a = ['a', 'b', 'c']

for i, name in enumerate(a):
    print (i, name)
```

在cookbook里介绍，如果你要计算文件的行数，可以这样写：

```
count = len(open(thefilepath,'rU').readlines())
```

前面这种方法简单，但是可能比较慢，当文件比较大时甚至不能工作，下面这种循环读取的方法更合适些。

```
Count = -1
For count,line in enumerate(open(thefilepath,'rU')):
    Pass
Count += 1
```

6. 利用列表映射、解析、过滤机制来创建列表

a) 映射

```
li = [1, 2, 3, 4]
b = [e**2 for e in li] #[1, 4, 9, 16]
```

b) 解析

```
dic = {'a':1, 'b':2, 'c':3, 'd':4}
b = ['%s=%s' %(k, v) for k, v in dic.items()]
 #['a=1', 'c=3', 'b=2', 'd=4']
```

c) 过滤

```
data = [7, 20, 3, 15, 11]
result = [i*3 for i in data if i > 10]
print result
```

7. Create dict from keys and values using `zip`

```
key = ['a', 'b', 'c', 'd', 'e']
data = [7, 20, 3, 15, 11]
d = dict(zip(key, data))
print d #{'a': 7, 'c': 3, 'b': 20, 'e': 11, 'd': 15}
```

注：如果key与data的长度不同，则会按照最短长度进行匹配

8. Python中的`range`与`xrange`

a) for i in range(NUM)会返回列表

b) for i in xrange(NUM)会返回迭代器而不是列表

如果不需要返回列表，则最好使用`xrange`，性能很好。

测试代码:

```
import time, datetime

def test_range(num):
    for i in range(num):
        pass

def test_xrange(num):
    for i in xrange(num):
        pass

if __name__=='__main__':
    num = 100000000
    start = time.time()
    test_range(num)
    print 'cost of range: %s' %(time.time() - start)
    start = time.time()
    test_xrange(num)
    print 'cost of xrange: %s' %(time.time() - start)
```

9. Python中的`lambda`，`filter`，`map`，`reduce`内置函数

a) `lambda`匿名函数

用`lambda`关键字定义一个匿名函数a，对于输入参数x，执行:后的操作并返回。

```
a = lambda x : x + 2
print a(2) #4
```

b) filter(func, seq)

`filter`关键词用func过滤seq中得每个成员，并将func返回为True的成员组成一个新seq返回

```
a = lambda x : x > 2

def a2(x):
    return x <= 2

data = [15, -7, 2, -5, 3, 10]

print filter(a, data)   #[15, 3, 10]
print filter(a2, data)  #[-7, 2, -5]
```

c) map(func, seq)

`map`关键词用func处理seq中的每个成员，并组成一个新的seq返回

```
data = [15, -7, 2, -5, 3, 10]

a3 = lambda x : -x #求反

print map(a3, data)     #[-15, 7, -2, 5, -3, -10]
```

d) reduce(func, seq, init_v)

`reduce`用于迭代计算，func函数必须有两个参数，首先从seq中两个元素取出分别作为func的两个参数(若指定init_v，则取seq的首个元素与init_v作为参数)，返回结果与seq中后续元素迭代计算。

```
def a(x, y):
    return x + y

data = [1, 2, 3, 4, 5, 6, 7, 8, 9]

print reduce(a, data, 0) #45
print reduce(a, data, 1) #46
```

Good luck :)