**Python中配置文件的使用**

因为项目的需要，要用到Python加载配置文件。本文主要参考了[http://www.cnblogs.com/victorwu/p/5762931.html](http://www.cnblogs.com/victorwu/p/5762931.html" \t "_blank)

**配置文件的读取示例**

a) 配置文件中包含一个或多个 section， 每个 section 有自己的 option；

b) section 用 [sect\_name] 表示，每个option是一个键值对，使用分隔符 = 或 : 隔开；

c) 在 option 分隔符两端的空格会被忽略掉

d) 配置文件使用 # 和 ; 注释

下面给出一个配置文件的例子。

[mongodb]

host = amaster

[kafka]

bootstrap\_servers = amaster:9092,anode1:9092,anode2:9092

[log]

log\_file\_name = kuaidaili\_spider.log

[zookeeper]

conn\_str = amaster:2181,anode1:2181,anode2:2181

测试代码：

#-\*- coding: utf-8 -\*-

import ConfigParser

def func():

cp = ConfigParser.SafeConfigParser()

cp.read('kuaidaili\_spider.cfg')

print 'mongodb host',cp.get('mongodb','host')

print 'kafka bootstrap\_servers',cp.get('kafka','bootstrap\_servers')

print 'log file name',cp.get('log','log\_file\_name')

print 'zookeeper conn string',cp.get('zookeeper','conn\_str')

if \_\_name\_\_ == '\_\_main\_\_':

func()

输出结果：

mongodb host amaster

kafka bootstrap\_servers amaster:9092,anode1:9092,anode2:9092

log file name kuaidaili\_spider.log

zookeeper conn string amaster:2181,anode1:2181,anode2:2181

下面的内容来自[http://www.cnblogs.com/victorwu/p/5762931.html](http://www.cnblogs.com/victorwu/p/5762931.html" \t "_blank)

**Unicode 编码的配置**

配置文件如果包含 Unicode 编码的数据，需要使用 codecs 模块以合适的编码打开配置文件。

myapp.conf

[msg]

hello = 你好

config\_parser\_unicode.py

import ConfigParser

import codecs

cp = ConfigParser.SafeConfigParser()

with codecs.open('myapp.conf', 'r', encoding='utf-8') as f:

cp.readfp(f)

print cp.get('msg', 'hello')

**allow\_no\_value**

通常情况下， option 是一个键值对。但是，当 SafeConfigParser 的参数 allow\_no\_value 设置成 True 时，它允许 option 不设置值而只是作为一个标识。

allow\_no\_value.conf

# option as Flag

[flag]

flag\_opt

allow\_no\_value.py

import ConfigParser

cp = ConfigParser.SafeConfigParser(allow\_no\_value = True)

cp.read('myapp.conf')

print cp.get('flag', 'flag\_opt'); # None

allow\_no\_value 默认设置成 False，此时如果配置文件中存在没有设置值的 option，在读取配置文件时将抛出异常

ConfigParser.ParsingError。当 allow\_no\_value 设置成 True 时，如果一个 option 没有设置值，has\_option 方法会返回 True，get 方法会返回 None。

**DEFAULT section**

如果配置文件中存在一个名为 DEFAULT 的 section，那么其他 section 会扩展它的 option 并且可以覆盖它的 option。

db.conf

[DEFAULT]

host = 127.0.0.1

port = 3306

[db\_root]

user = root

pass = root

[db\_huey]

host = 192.168.1.101

user = huey

pass = huey

default\_section.py

print cp.get('db\_root', 'host') # 127.0.0.1

print cp.get('db\_huey', 'host') # 192.168.1.101

**插值 Interpolation**

SafeConfigParser 提供了插值的特性来结合数据。

url.conf

[DEFAULT]

url = %(protocol)s://%(server)s:%(port)s/

[http]

protocol = http

server = localhost

port = 8080

[ftp]

url = %(protocol)s://%(server)s/

protocol = ftp

server = 192.168.1.102

interpolation\_demo.py

import ConfigParser

cp = ConfigParser.SafeConfigParser()

cp.read('url.conf')

print cp.get('http', 'url') # http://localhost:8080/

print cp.get('ftp', 'url') # ftp://192.168.1.102/