Python编程中常用的12种基础知识总结

2013/07/05 | 分类： Python | 2 条评论 | 标签： Python

Python编程中常用的12种基础知识总结：正则表达式替换，遍历目录方法，列表按列排序、去重，字典排序，字典、列表、字符串互转，时间对象操作，命令行参数解析(getopt)，print格式化输出，进制转换，Python调用系统命令或者脚本，Python 读写文件。

1、正则表达式替换
目标：将字符串line中的 overview.gif 替换成其他字符串

```
>>> line = '<IMG ALIGN="middle" SRC=\'#\'" /span>
>>> mo=re.compile(r'(?<=SRC=)"([\w+\.]+)"',re.I)

>>> mo.sub(r'"\1****"',line)
'<IMG ALIGN="middle" SRC=\'#\'" /span>

>>> mo.sub(r'replace_str_\1',line)
'<IMG ALIGN="middle" replace_str_overview.gif BORDER="0" ALT="">'< /span>

>>> mo.sub(r'"testetstset"',line)
'<IMG ALIGN="middle" SRC=\'#\'" /span>
```

注意：其中 \1 是匹配到的数据，可以通过这样的方式直接引用

2、遍历目录方法
在某些时候，我们需要遍历某个目录找出特定的文件列表，可以通过os.walk方法来遍历，非常方便

```
import os
fileList = []
rootdir = "/data"
for root, subFolders, files in os.walk(rootdir):
if '.svn' in subFolders: subFolders.remove('.svn')  # 排除特定目录
for file in files:
  if file.find(".t2t") != -1:# 查找特定扩展名的文件
      file_dir_path = os.path.join(root,file)
      fileList.append(file_dir_path)

print fileList
```

3、列表按列排序(list sort)
如果列表的每个元素都是一个元组(tuple),我们要根据元组的某列来排序的化，可参考如下方法

下面例子我们是根据元组的第2列和第3列数据来排序的,而且是倒序(reverse=True)

```
>>> a = [('2011-03-17', '2.26', 6429600, '0.0'), ('2011-03-16', '2.26', 12036900, '-3.0'),
 ('2011-03-15', '2.33', 15615500,'-19.1')]
>>> print a[0][0]
2011-03-17
>>> b = sorted(a, key=lambda result: result[1],reverse=True)
>>> print b
[('2011-03-15', '2.33', 15615500, '-19.1'), ('2011-03-17', '2.26', 6429600, '0.0'),
('2011-03-16', '2.26', 12036900, '-3.0')]
>>> c = sorted(a, key=lambda result: result[2],reverse=True)
>>> print c
[('2011-03-15', '2.33', 15615500, '-19.1'), ('2011-03-16', '2.26', 12036900, '-3.0'),
('2011-03-17', '2.26', 6429600, '0.0')]
```

4、列表去重(list uniq)
有时候需要将list中重复的元素删除，就要使用如下方法

```
>>> lst= [(1,'sss'),(2,'fsdf'),(1,'sss'),(3,'fd')]
>>> set(lst)
set([(2, 'fsdf'), (3, 'fd'), (1, 'sss')])
>>>
>>> lst = [1, 1, 3, 4, 4, 5, 6, 7, 6]
>>> set(lst)
set([1, 3, 4, 5, 6, 7])
```

5、字典排序(dict sort)

一般来说，我们都是根据字典的key来进行排序，但是我们如果想根据字典的value值来排序，就使用如下方法

```
>>> from operator import itemgetter
>>> aa = {"a":"1","sss":"2","ffdf":'5',"ffff2":'3'}
>>> sort_aa = sorted(aa.items(),key=itemgetter(1))
>>> sort_aa
[('a', '1'), ('sss', '2'), ('ffff2', '3'), ('ffdf', '5')]
```

从上面的运行结果看到，按照字典的value值进行排序的

6、字典,列表,字符串互转
以下是生成数据库连接字符串,从字典转换到字符串

```
>>> params = {"server":"mpilgrim", "database":"master", "uid":"sa", "pwd":"secret"}
>>> ["%s=%s" % (k, v) for k, v in params.items()]
['server=mpilgrim', 'uid=sa', 'database=master', 'pwd=secret']
>>> ";".join(["%s=%s" % (k, v) for k, v in params.items()])
'server=mpilgrim;uid=sa;database=master;pwd=secret'
```

下面的例子 是将字符串转化为字典

```
>>> a = 'server=mpilgrim;uid=sa;database=master;pwd=secret'
>>> aa = {}
>>> for i in a.split(';'):aa[i.split('=',1)[0]] = i.split('=',1)[1]
...
>>> aa
{'pwd': 'secret', 'database': 'master', 'uid': 'sa', 'server': 'mpilgrim'}
```

7、时间对象操作

```
将时间对象转换成字符串
>>> import datetime
>>> datetime.datetime.now().strftime("%Y-%m-%d %H:%M")
  '2011-01-20 14:05'

时间大小比较
>>> import time
>>> t1 = time.strptime('2011-01-20 14:05',"%Y-%m-%d %H:%M")
>>> t2 = time.strptime('2011-01-20 16:05',"%Y-%m-%d %H:%M")
>>> t1 > t2
  False
>>> t1 < t2
  True

时间差值计算,计算8小时前的时间
>>> datetime.datetime.now().strftime("%Y-%m-%d %H:%M")
  '2011-01-20 15:02'
>>> (datetime.datetime.now() - datetime.timedelta(hours=8)).strftime("%Y-%m-%d %H:%M")
  '2011-01-20 07:03'

将字符串转换成时间对象
>>> endtime=datetime.datetime.strptime('20100701',"%Y%m%d")
>>> type(endtime)
  <type 'datetime.datetime'>
>>> print endtime
  2010-07-01 00:00:00

将从 1970-01-01 00:00:00 UTC 到现在的秒数，格式化输出

>>> import time
>>> a = 1302153828
>>> time.strftime("%Y-%m-%d %H:%M:%S",time.localtime(a))
  '2011-04-07 13:23:48'
```

8、命令行参数解析(getopt)

通常在编写一些日运维脚本时，需要根据不同的条件，输入不同的命令行选项来实现不同的功能 在Python中提供了getopt模块很好的实现了命令行参数的解析,下面距离说明。请看如下程序:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import sys,os,getopt
def usage():
print '''''
Usage: analyse_stock.py [options...]
Options:
-e : Exchange Name
-c : User-Defined Category Name
-f : Read stock info from file and save to db
-d : delete from db by stock code
-n : stock name
-s : stock code
-h : this help info
test.py -s haha -n "HA Ha"
'''

try:
opts, args = getopt.getopt(sys.argv[1:],'he:c:f:d:n:s:')
except getopt.GetoptError:
```

```
usage()
sys.exit()
if len(opts) == 0:
usage()
sys.exit()

for opt, arg in opts:
if opt in ('-h', '--help'):
  usage()
  sys.exit()
elif opt == '-d':
  print "del stock %s" % arg
elif opt == '-f':
  print "read file %s" % arg
elif opt == '-c':
  print "user-defined %s " % arg
elif opt == '-e':
  print "Exchange Name %s" % arg
elif opt == '-s':
  print "Stock code %s" % arg
elif opt == '-n':
  print "Stock name %s" % arg

sys.exit()
```

9、print 格式化输出

9.1、格式化输出字符串

```
截取字符串输出,下面例子将只输出字符串的前3个字母
>>> str="abcdefg"
>>> print "%.3s" % str
  abc
按固定宽度输出，不足使用空格补全,下面例子输出宽度为10
>>> str="abcdefg"
>>> print "%10s" % str
     abcdefg
截取字符串，按照固定宽度输出
>>> str="abcdefg"
>>> print "%10.3s" % str
         abc
浮点类型数据位数保留
>>> import fpformat
>>> a= 0.0030000000005
>>> b=fpformat.fix(a,6)
>>> print b
  0.003000
对浮点数四舍五入,主要使用到round函数
>>> from decimal import *
>>> a ="2.26"
>>> b ="2.29"
>>> c = Decimal(a) - Decimal(b)
>>> print c
  -0.03
>>> c / Decimal(a) * 100
  Decimal('-1.327433628318584070796460177')
>>> Decimal(str(round(c / Decimal(a) * 100, 2)))
  Decimal('-1.33')
```

9.2、进制转换

有些时候需要作不同进制转换，可以参考下面的例子(%x 十六进制,%d 十进制,%o 八进制)

```
>>> num = 10
>>> print "Hex = %x,Dec = %d,Oct = %o" %(num,num,num)
  Hex = a,Dec = 10,Oct = 12
```

10、Python调用系统命令或者脚本

```
使用 os.system() 调用系统命令 ， 程序中无法获得到输出和返回值
>>> import os
```

```
>>> os.system('ls -l /proc/cpuinfo')
>>> os.system("ls -l /proc/cpuinfo")
  -r--r--r-- 1 root root 0  3月 29 16:53 /proc/cpuinfo
  0

使用 os.popen() 调用系统命令，程序中可以获得命令输出，但是不能得到执行的返回值
>>> out = os.popen("ls -l /proc/cpuinfo")
>>> print out.read()
  -r--r--r-- 1 root root 0  3月 29 16:59 /proc/cpuinfo

使用 commands.getstatusoutput() 调用系统命令，程序中可以获得命令输出和执行的返回值
>>> import commands
>>> commands.getstatusoutput('ls /bin/ls')
  (0, '/bin/ls')
```

11、Python 捕获用户 Ctrl+C ,Ctrl+D 事件

有些时候，需要在程序中捕获用户键盘事件，比如ctrl+c退出，这样可以更好的安全退出程序

```
try:
    do_some_func()
except KeyboardInterrupt:
    print "User Press Ctrl+C,Exit"
except EOFError:
    print "User Press Ctrl+D,Exit"
```

12、Python 读写文件

```
一次性读入文件到列表，速度较快，适用文件比较小的情况下
track_file = "track_stock.conf"
fd = open(track_file)
content_list = fd.readlines()
fd.close()
for line in content_list:
    print line

逐行读入，速度较慢,适用没有足够内存读取整个文件(文件太大)
fd = open(file_path)
fd.seek(0)
title = fd.readline()
keyword = fd.readline()
uuid = fd.readline()
fd.close()

写文件 write 与 writelines 的区别

Fd.write(str) : 把str写到文件中，write()并不会在str后加上一个换行符
Fd.writelines(content) : 把content的内容全部写到文件中,原样写入，不会在每行后面加上任何东西
```

>>> Python频道微信号：PythonCoder，扫描加关注，碎片时间提升Python开发技能！

0

相关文章

- 30个有关Python的小技巧
- 30 行 Python 代码搞定 X 算法
- Python高级编程技巧
- Python 中的 is 和 id
- Python的defaultdict模块和namedtuple模块
- Vim 7.4 计划已公布，具体发布时间待定
- 理解 Python 字节码
- Python趣文：Import Girlfriend
- Python解释器简介（4）：动态语言
- 可爱的 Python ：Python中的函数式编程，第三部分

发表评论

Comment form

Name*

姓名

邮箱*

请填写邮箱

网站（请以 http://开头）

请填写网站地址

评论内容*

请填写评论内容

(*) 表示必填项

提交评论

2 条评论

1. popucui 说道：
   2013/09/24 下午 8:14

   9.2 进制转换，“有些时候需要作不同进制转换，可以参考下面的例子(%x 十六进制,%d 十进制,%o 十进制)” ，”%o”是指的八进制呢

   0 0

   回复

   - 黄利民 说道：
     2013/09/24 下午 9:23

     谢谢提醒，已修正笔误。

     0 0

     回复

来自微博的评论

meimei_5106 还可以输入140字

顺便说点什么吧……

表情 同步到微博

0条评论

还没有人评论过，赶快抢沙发吧！

获得微博评论箱

« 30个有关Python的小技巧
学习Python编程的11个资源 »

Search for:

Search

Search

微信关注: PythonCoder
分享Python开发相关的技术文章、工具资源和热点资讯。扫描加关注，碎片时间提升Python开发技能！

- 本月热门文章
- 年度热门文章
- 热门标签

0 检测Python程序执行效率及内存和CPU使用

1 13岁Python开发者写的多人游戏编程（下）

2 Eric Raymond对于几大开发语言的评价

3 如何将大量数据放入有限内存

4 理解 Python 字节码

5 Python的中文编码问题

6 Django运行方式及处理流程总结

7 Python的defaultdict模块和namedtuple模块

8 Python和数据科学的起步指南

9 我希望早点就知道的10个Python用法

最新评论

- Re: 15个最受欢迎的Python开源框架
  Hi，Flask 是后来补上的 黄利民
- Re: 15个最受欢迎的Python开源框架
  有的。 Lucius
- Re: 检测Python程序执行效率及内存和CPU使用的7种方法
  到目前为止，还很少去关注执行效率，这个必须点赞 小白菜
- Re: 深刻理解Python中的元类(metaclass)
  好文章。 dongguangming
- Re: 12岁的少年教你用Python做小游戏
  很好作为入门的实例 Sean Zhao
- Re: 用Python的 slots 节省9G内存
  想问下这些技巧/知识点是在哪看到的？有没有较好的书籍推荐？基本语法还可以，但是这些技巧基本没接触过，... 秋
- Re: Python十分钟入门
  最后连接MYSQL的那句 conn MySQLdb.connect(host=’localhost’... puyangsky
- Re: 10 个 Python IDE 和代码编辑器
  Vim 配上Jedi和Pymode， 行云流水般的~~~~ 浪子

关于伯乐在线-Python频道

Python频道分享Python开发技术，前端相关的行业动态。欢迎通过微博和微信【 微信号: PythonCoder 】关注。

欢迎关注更多频道

头条 - 分享和发现有价值的内容与观点
相亲 - 为IT单身男女服务的征婚传播平台
资源 - 优秀的工具资源导航
翻译 - 翻译传播优秀的外文文章
博客 - 国内外的精选博客文章
iOS - 专注iOS技术分享
安卓 - 专注Android技术分享
前端 - JavaScript, HTML5, CSS
Java - 专注Java技术分享
Python - 专注Python技术分享

联系我们

商务合作

Email：bd@Jobbole.com
QQ： 2302462408 （加好友请注明来意）

网站使用问题

请直接联系我们询问或者反馈

 头条 博客 资源 翻译 小组 相亲
本站由 UCloud 赞助云主机，七牛 赞助云存储