**python之sys模块**

2018年01月05日 17:51:31

阅读数：1522

python版本: Python 2.7.6   
1: sys是python自带模块.   
利用 import 语句输入sys 模块。   
当执行import sys后, python在 sys.path 变量中所列目录中寻找 sys 模块文件。然后运行这个模块的主块中的语句进行初始化，然后就可以使用模块了 。

2: sys模块常见函数   
可以通过**dir()**方法查看模块中可用的方法. 结果如下, 很多我都没有用过, 所以只是简单介绍几个自己用过的方法.

$ python

Python 2.7.6 (default, Oct 26 2016, 20:30:19)

[GCC 4.8.4] on linux2

Type "help", "copyright", "credits" or "license" for more information.

>>> import sys

>>> dir(sys)

['\_\_displayhook\_\_', '\_\_doc\_\_', '\_\_excepthook\_\_', '\_\_name\_\_', '\_\_package\_\_', '\_\_stderr\_\_', '\_\_stdin\_\_', '\_\_stdout\_\_', '\_clear\_type\_cache', '\_current\_frames', '\_getframe', '\_mercurial', '\_multiarch', 'api\_version', 'argv', 'builtin\_module\_names', 'byteorder', 'call\_tracing', 'callstats', 'copyright', 'displayhook', 'dont\_write\_bytecode', 'exc\_clear', 'exc\_info', 'exc\_type', 'excepthook', 'exec\_prefix', 'executable', 'exit', 'flags', 'float\_info', 'float\_repr\_style', 'getcheckinterval', 'getdefaultencoding', 'getdlopenflags', 'getfilesystemencoding', 'getprofile', 'getrecursionlimit', 'getrefcount', 'getsizeof', 'gettrace', 'hexversion', 'long\_info', 'maxint', 'maxsize', 'maxunicode', 'meta\_path', 'modules', 'path', 'path\_hooks', 'path\_importer\_cache', 'platform', 'prefix', 'ps1', 'ps2', 'py3kwarning', 'pydebug', 'setcheckinterval', 'setdlopenflags', 'setprofile', 'setrecursionlimit', 'settrace', 'stderr', 'stdin', 'stdout', 'subversion', 'version', 'version\_info', 'warnoptions']

* 1
* 2
* 3
* 4
* 5
* 6
* 7

(1) **sys.argv** 实现从程序外部向程序传递参数   
sys.argv 变量是一个包含了命令行参数的字符串列表, 利用命令行想程序传递参数. 其中,脚本的名称总是 sys.argv 列表的第一个参数。

(2) **sys.path** 包含输入模块的目录名列表。   
获取指定模块搜索路径的字符串集合，可以将写好的模块放在得到的某个路径下，就可以在程序中import时正确找到。在import导入module\_name时，就是根据sys.path的路径来搜索module.name，也可以自定义添加模块路径。   
sys.path.append(“自定义模块路径”)

(3) **sys.exit([arg])** 程序中间的退出, arg=0为正常退出   
一般情况下执行到主程序末尾，解释器自动退出，但是如果需要中途退出程序，可以调用sys.exit函数，带有一个可选的整数参数返回给调用它的程序，表示你可以在主程序中捕获对sys.exit的调用。（0是正常退出，其他为异常）当然也可以用字符串参数，表示错误不成功的报错信息。

(4) **sys.modules**   
sys.modules是一个全局字典，该字典是python启动后就加载在内存中。每当程序员导入新的模块，sys.modules将自动记录该模块。当第二次再导入该模块时，python会直接到字典中查找，从而加快了程序运行的速度。它拥有字典所拥有的一切方法.

(5) **sys.getdefaultencoding() / sys.setdefaultencoding() / sys.getfilesystemencoding()**   
sys.getdefaultencoding()   
获取系统当前编码，一般默认为ascii。   
sys.setdefaultencoding()   
设置系统默认编码，执行dir（sys）时不会看到这个方法，在解释器中执行不通过，可以先执行reload(sys)，在执行 setdefaultencoding(‘utf8’)，此时将系统默认编码设置为utf8。（见设置系统默认编码 ）   
sys.getfilesystemencoding()   
获取文件系统使用编码方式，Windows下返回’mbcs’，mac下返回’utf-8’

(6) **sys.stdin, sys.stdout, sys.stderr**   
stdin , stdout , 以及stderr 变量包含与标准I/O 流对应的流对象. 如果需要更好地控制输出,而print 不能满足你的要求, 它们就是你所需要的. 你也可以替换它们, 这时候你就可以重定向输出和输入到其它设备( device ), 或者以非标准的方式处理它们.

(7) **sys.platform**   
获取当前系统平台. 如：win32、Linux等。

3: **实例**   
(1) sys.argv sys.path

$ cat sys-test.py

#!/usr/bin/python

import sys

print 'The command line arguments are:'

for i in sys.argv:

print i

print '\n\nThe PYTHONPATH is', sys.path, '\n'

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9

运行结果:

$ python sys-test.py my name is ubuntu

The command line arguments are:

sys-test.py

my

name

is

ubuntu

The PYTHONPATH is ['/work/python-practice', '/usr/lib/python2.7', '/usr/lib/python2.7/plat-x86\_64-linux-gnu', '/usr/lib/python2.7/lib-tk', '/usr/lib/python2.7/lib-old', '/usr/lib/python2.7/lib-dynload', '/usr/local/lib/python2.7/dist-packages', '/usr/lib/python2.7/dist-packages', '/usr/lib/python2.7/dist-packages/PILcompat', '/usr/lib/python2.7/dist-packages/gtk-2.0', '/usr/lib/pymodules/python2.7', '/usr/lib/python2.7/dist-packages/ubuntu-sso-client']

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9

(2) **sys.exit**

#!/usr/bin/python

import sys

def exitfunc(value):

print (value)

sys.exit(0)

print("hello")

try:

sys.exit(90)

except SystemExit as value:

exitfunc(value)

print("come?")

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15

运行结果:

hello

90

* 1
* 2

程序首先打印hello，在执行exit(90)，抛异常把90传给values，values在传进函数中执行，打印90程序退出。后面的”come?”因为已经退出所以不会被打印. 而此时如果把exitfunc函数里面的sys.exit(0)去掉,那么程序会继续执行到输出”come?”.

(3) **sys.modules**   
sys.modules.keys() 返回所有已经导入的模块列表   
keys是模块名   
values是模块   
modules返回路径

#!/usr/bin/python

import sys

print(sys.modules.keys())

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print(sys.modules.values())

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print(sys.modules["os"])

运行结果:

['copy\_reg', 'sre\_compile', '\_sre', 'encodings', 'site', '\_\_builtin\_\_', 'sysconfig', '\_\_main\_\_', 'encodings.encodings', 'abc', 'posixpath', '\_weakrefset', 'errno', 'encodings.codecs', 'sre\_constants', 're', '\_abcoll', 'types', '\_codecs', 'encodings.\_\_builtin\_\_', '\_warnings', 'genericpath', 'stat', 'zipimport', '\_sysconfigdata', 'warnings', 'UserDict', 'encodings.ascii', 'sys', 'codecs', '\_sysconfigdata\_nd', 'os.path', 'sitecustomize', 'signal', 'traceback', 'linecache', 'posix', 'encodings.aliases', 'exceptions', 'sre\_parse', 'os', '\_weakref']

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[<module 'copy\_reg' from '/usr/lib/python2.7/copy\_reg.pyc'>, <module 'sre\_compile' from '/usr/lib/python2.7/sre\_compile.pyc'>, <module '\_sre' (built-in)>, <module 'encodings' from '/usr/lib/python2.7/encodings/\_\_init\_\_.pyc'>, <module 'site' from '/usr/lib/python2.7/site.pyc'>, <module '\_\_builtin\_\_' (built-in)>, <module 'sysconfig' from '/usr/lib/python2.7/sysconfig.pyc'>, <module '\_\_main\_\_' from 'sys-test1.py'>, None, <module 'abc' from '/usr/lib/python2.7/abc.pyc'>, <module 'posixpath' from '/usr/lib/python2.7/posixpath.pyc'>, <module '\_weakrefset' from '/usr/lib/python2.7/\_weakrefset.pyc'>, <module 'errno' (built-in)>, None, <module 'sre\_constants' from '/usr/lib/python2.7/sre\_constants.pyc'>, <module 're' from '/usr/lib/python2.7/re.pyc'>, <module '\_abcoll' from '/usr/lib/python2.7/\_abcoll.pyc'>, <module 'types' from '/usr/lib/python2.7/types.pyc'>, <module '\_codecs' (built-in)>, None, <module '\_warnings' (built-in)>, <module 'genericpath' from '/usr/lib/python2.7/genericpath.pyc'>, <module 'stat' from '/usr/lib/python2.7/stat.pyc'>, <module 'zipimport' (built-in)>, <module '\_sysconfigdata' from '/usr/lib/python2.7/\_sysconfigdata.pyc'>, <module 'warnings' from '/usr/lib/python2.7/warnings.pyc'>, <module 'UserDict' from '/usr/lib/python2.7/UserDict.pyc'>, <module 'encodings.ascii' from '/usr/lib/python2.7/encodings/ascii.pyc'>, <module 'sys' (built-in)>, <module 'codecs' from '/usr/lib/python2.7/codecs.pyc'>, <module '\_sysconfigdata\_nd' from '/usr/lib/python2.7/plat-x86\_64-linux-gnu/\_sysconfigdata\_nd.pyc'>, <module 'posixpath' from '/usr/lib/python2.7/posixpath.pyc'>, <module 'sitecustomize' from '/usr/lib/python2.7/sitecustomize.pyc'>, <module 'signal' (built-in)>, <module 'traceback' from '/usr/lib/python2.7/traceback.pyc'>, <module 'linecache' from '/usr/lib/python2.7/linecache.pyc'>, <module 'posix' (built-in)>, <module 'encodings.aliases' from '/usr/lib/python2.7/encodings/aliases.pyc'>, <module 'exceptions' (built-in)>, <module 'sre\_parse' from '/usr/lib/python2.7/sre\_parse.pyc'>, <module 'os' from '/usr/lib/python2.7/os.pyc'>, <module '\_weakref' (built-in)>]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<module 'os' from '/usr/lib/python2.7/os.pyc'>

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15

(4) **sys.stdin/sys.stdout/sys.stderr**   
stdin,stdout,stderr在Python中都是文件属性对象, 他们在python启动时自动与shell环境中的标准输入, 输出, 出错相关. 而python程序在shell中的I/O重定向是有shell来提供的,与python本身没有关系.python程序内部将stdin, stdout, stderr读写操作重定向到一个内部对象.

标准输入

#!/usr/bin/python

import sys

#print('Hi, %s!' %input('Please enter your name: ')) python3.\*版本用input

print('Hi, %s!' %raw\_input('Please enter your name: ')) #python2.\*版本用raw\_input

运行结果:

Please enter your name: er

Hi, er!

等同于:

#!/usr/bin/python

import sys

print('Please enter your name:')

name=sys.stdin.readline()[:-1]

print('Hi, %s!' %name)

标准输出

print('Hello World!\n')

等同于:

#!/usr/bin/python

import sys

sys.stdout.write('output resule is good!\n')

其他实验

#!/usr/bin/python

import sys

for i in (sys.stdin, sys.stdout, sys.stderr):

print(i)

执行结果:

python sys-test1.py

<open file '<stdin>', mode 'r' at 0x7fa4e630f0c0>

<open file '<stdout>', mode 'w' at 0x7fa4e630f150>

<open file '<stderr>', mode 'w' at 0x7fa4e630f1e0>