02_sys_module

June 29, 2019

```
In [1]: import sys
```

```
In [2]: print(dir(sys))
```

```
['__breakpointhook__', '__displayhook__', '__doc__', '__excepthook__', '__interactivehook__',
```

```
In [3]: print('sys.base_exec_prefix =', sys.base_exec_prefix)
        print('sys.base_prefix      =', sys.base_prefix)
        print('sys.prefix           =', sys.prefix)
        print('sys.exec_prefix      =', sys.exec_prefix)
```

```
sys.base_exec_prefix = C:\ProgramData\Anaconda3
sys.base_prefix      = C:\ProgramData\Anaconda3
sys.prefix           = C:\ProgramData\Anaconda3
sys.exec_prefix      = C:\ProgramData\Anaconda3
```

```
In [5]: # big    - big-endian    (most significant byte first) platforms
        # little - little-endian (least significant byte first) platforms

        print('sys.byteorder        =', sys.byteorder)
```

```
sys.byteorder        = little
```

```
In [7]: print('sys.copyright        =', sys.copyright)
```

```
sys.copyright        = Copyright (c) 2001-2018 Python Software Foundation.
All Rights Reserved.

Copyright (c) 2000 BeOpen.com.
All Rights Reserved.

Copyright (c) 1995-2001 Corporation for National Research Initiatives.
All Rights Reserved.

Copyright (c) 1991-1995 Stichting Mathematisch Centrum, Amsterdam.
All Rights Reserved.
```

```
In [8]: print('sys._debugmallocstats() =', sys._debugmallocstats())
```

```
sys._debugmallocstats() = None
```

```
In [9]: print('sys.dont_write_bytecode   =', sys.dont_write_bytecode)
        # make this value to True , to stop creation of .pyc files on python file imports
```

```
sys.dont_write_bytecode   = False
```

```
In [11]: print('sys.getdefaultencoding()       =', sys.getdefaultencoding())
         print('sys.getfilesystemencoding()    =', sys.getfilesystemencoding())
         print('sys.getfilesystemencodeerrors()=', sys.getfilesystemencodeerrors())
```

```
sys.getdefaultencoding()       = utf-8
sys.getfilesystemencoding()    = utf-8
sys.getfilesystemencodeerrors()= surrogatepass
```

```
In [12]: print('sys.getswitchinterval()    =', sys.getswitchinterval())
```

```
sys.getswitchinterval()    = 0.005
```

Reference count

```
In [22]: num1 = 999
         print('sys.getrefcount(1234) =', sys.getrefcount(1234))
         del num1
         print('sys.getrefcount(1234) =', sys.getrefcount(1234))
```

```
sys.getrefcount(1234) = 3
sys.getrefcount(1234) = 3
```

```
In [21]: for i in range(500):
             print(sys.getrefcount(i), end=' ')
```

```
4183 2165 993 451 505 226 213 140 348 138 158 154 159 91 83 100 211 102 76 105 450 87 91 78 78
```

```
In [23]: for w in ["RandomJunkBlahBlah", "python", "version", "error", "Guido"]:
             print(w, sys.getrefcount(w))
```

```
RandomJunkBlahBlah 5
python 28
version 190
error 339
Guido 5
```

```
In [24]: num1 = 888888888888888888888
         print('sys.sys.getsizeof(num1)=', sys.getsizeof(num1))
         # Only the memory consumption directly attributed to the object
         # is accounted for, not the memory consumption of objects it refers to.
```

```
sys.sys.getsizeof(num1)= 36
```

```
In [25]: print('sys.getwindowsversion()=', sys.getwindowsversion())
```

```
sys.getwindowsversion()= sys.getwindowsversion(major=6, minor=1, build=7601, platform=2, servic
```

```
In [27]: sys.hash_info
```

```
Out[27]: sys.hash_info(width=64, modulus=2305843009213693951, inf=314159, nan=0, imag=1000003,
```

```
In [28]: print(hash(9))
         print(hash(9.22))
         print(hash(True))
         print(hash(None))
         print(hash((12, 2,3,23)))
         print(hash(frozenset({12, 23, 4})))

         # unhashables
         # print(hash([12, 2,3,23]))
         # print(hash({12, 2,3,23}))
         # print(hash({'a':1, 'b':2}))
```

```
9
507285462027014153
1
549413229774
-6397297334068172147
-374089322017822008
```

```
In [29]: print('sys.hexversion             =', sys.hexversion)
         print(hex(13))
```

```
sys.hexversion             = 50790896
0xd
```

```
In [31]: sys.implementation
```

```
Out[31]: namespace(cache_tag='cpython-37', hexversion=50790896, name='cpython', version=sys.ver
```

```
In [32]: print('sys.int_info               =', sys.int_info)
```

```
sys.int_info               = sys.int_info(bits_per_digit=30, sizeof_digit=4)
```

```
In [33]: print('sys.__interactivehook__()  =', sys.__interactivehook__())
```

```
sys.__interactivehook__()  = None
```

```
In [34]: print('sys.maxsize                =', sys.maxsize)
```

```
sys.maxsize                = 9223372036854775807
```

```
In [37]: print('sys.maxunicode             =', sys.maxunicode)
```

```
sys.maxunicode             = 1114111
```

```
In [39]: sys.builtin_module_names
```

```
Out[39]: ('_abc',
          '_ast',
          '_bisect',
          '_blake2',
          '_codecs',
          '_codecs_cn',
          '_codecs_hk',
          '_codecs_iso2022',
          '_codecs_jp',
          '_codecs_kr',
          '_codecs_tw',
          '_collections',
          '_csv',
          '_datetime',
          '_functools',
          '_heapq',
          '_imp',
          '_io',
          '_json',
          '_locale',
          '_lsprof',
          '_md5',
          '_multibytecodec',
          '_opcode',
          '_operator',
          '_pickle',
          '_random',
          '_sha1',
          '_sha256',
          '_sha3',
          '_sha512',
          '_signal',
          '_sre',
          '_stat',
          '_string',
          '_struct',
          '_symtable',
          '_thread',
          '_tracemalloc',
          '_warnings',
          '_weakref',
          '_winapi',
          'array',
          'atexit',
          'audioop',
          'binascii',
          'builtins',
          'cmath',
          'errno',
          'faulthandler',
          'gc',
          'itertools',
          'marshal',
          'math',
          'mmap',
          'msvcrt',
          'nt',
          'parser',
          'sys',
          'time',
          'winreg',
          'xxsubtype',
          'zipimport',
          'zlib')
```

```
In [51]: sys.modules.keys()
```

```
Out[51]: dict_keys(['sys', 'builtins', '_frozen_importlib', '_imp', '_thread', '_warnings', '_w
```

```
In [41]: print('sys.version                =', sys.version)
         print('sys.api_version            =', sys.api_version) # C API version for this interp
         print('sys.version_info           =', sys.version_info)
         print('sys.winver                 =', sys.winver) # version number used to form regis
```

```
sys.version                = 3.7.1 (default, Dec 10 2018, 22:54:23) [MSC v.1915 64 bit (AMD64)]
sys.api_version            = 1013
sys.version_info           = sys.version_info(major=3, minor=7, micro=1, releaselevel='final',
sys.winver                 = 3.7
```

```
In [42]: print('sys.getrecursionlimit()', sys.getrecursionlimit())
```

```
sys.getrecursionlimit() 3000
```

```
In [47]: print('Hello world!')
         sys.stdout.write('Hello world!\n')
         sys.stderr.write('Hello world!\n')
```

```
Hello world!
Hello world!
```

```
Hello world!
```

```
In [44]: for i in (sys.stdin, sys.stdout, sys.stderr):
             print(i)
```

```
<_io.TextIOWrapper name='<stdin>' mode='r' encoding='cp1252'>
<ipykernel.iostream.OutStream object at 0x00000000058418D0>
<ipykernel.iostream.OutStream object at 0x000000000575FA20>
```

```
In [52]: for each_path in sys.path:
             print(each_path)
```

```
C:\Users\udhayPrakash\Desktop\PythonMaterial\python3\10_Modules\02_sys
C:\ProgramData\Anaconda3\python37.zip
C:\ProgramData\Anaconda3\DLLs
C:\ProgramData\Anaconda3\lib
C:\ProgramData\Anaconda3

C:\Users\udhayPrakash\AppData\Roaming\Python\Python37\site-packages
C:\ProgramData\Anaconda3\lib\site-packages
C:\ProgramData\Anaconda3\lib\site-packages\win32
C:\ProgramData\Anaconda3\lib\site-packages\win32\lib
C:\ProgramData\Anaconda3\lib\site-packages\Pythonwin
C:\Users\udhayPrakash\AppData\Roaming\Python\Python37\site-packages\IPython\extensions
C:\Users\udhayPrakash\.ipython
```