常用内建模块

os模块

```
In : import os

In : os.getcwd()
Out: '/Users/dongweiming/avalon'

In : os.chdir('..')

In : os.getcwd()
Out: '/Users/dongweiming'

In : os.getenv('SHELL')
Out: '/bin/zsh'

In : os.environ.get('SHELL')
Out: '/bin/zsh'

In : os.listdir('migrations')
Out[]:
['script.py.mako',
 'env.py',
 'alembic.ini',
 'versions',
 '__pycache__',
 'README']


In : os.walk('dir1')
Out: <generator object walk at 0x107a078e0>

In : list(os.walk('dir1'))
Out: [('dir1', ['dir3'], []), ('dir1/dir3', [], [])]
```

```
In : os.mkdir('dir1')
In : ll
total 0
drwxr-xr-x 2 dongweiming 64 Apr  1 12:07 dir1/

In : os.makedirs('dir2/dir3')

In : !tree
.
├── dir1
└── dir2
    └── dir3

3 directories, 0 files

In : !touch 1.txt

In : ls
1.txt  dir1/  dir2/

In : os.remove('1.txt')

In : os.rmdir('dir1')
In : os.rename('dir2', 'dir1')

In : ls
dir1/
```

os.path模块

```
In : p = '/home/dongwm/a.txt'

In : os.path.basename(p) # 获得指定文件路径的文件名字
Out: 'a.txt'

In : os.path.dirname(p) # 获得文件路径的目录名字
Out: '/home/dongwm'

In : os.path.exists(p)  # 判断文件或者目录是否存在
Out: False

In : os.path.exists('/Users/dongweiming/avalon')
Out: True

In : os.path.isdir(p)  # 判断指定路径是否是目录
Out: False

In : os.path.isdir('/Users/dongweiming')
Out: True
```

```
In : os.path.isfile(p)  # 判断指定路径是否是文件
Out: False

# 拼接路径
In : os.path.join('/Users', 'dongweiming', 'avalon/app.py'
Out: '/Users/dongweiming/avalon/app.py'

In : os.path.split(p)  # 路径拆分
Out: ('/home/dongwm', 'a.txt')

In : os.path.splitext(p)  # 获得路径的后缀
Out: ('/home/dongwm/a', '.txt')
```

sys模块

```
In : sys.platform  # 用来构建解释器的操作系统平台
Out: 'darwin'

In : sys.version  # 构建时的版本信息，包含完整的版本号和构建日期、编译器、平台信息等
Out: '3.6.4 (default, Feb 28 2018, 12:22:57) \n[GCC 4.2.1 Compatible Apple LLVM
9.0.0 (clang-900.0.39.2)]'

In : sys.version_info  # 同样是版本信息，但不是字符串，可以直接获得对应类型版本的信息
Out: sys.version_info(major=3, minor=6, micro=4, releaselevel='final', serial=0)

In : sys.path[0]  # 搜索模块的路径列表
Out: '/Users/dongweiming/avalon/venv/lib/python3.6/site-packages'

In : sys.modules.get('xml')  # 已经导入的模块列表
Out: <module 'xml' from '/usr/local/Cellar/python3/3.6.4_2/Frameworks/Python.framework/
Versions/3.6/lib/python3.6/xml/__init__.py'>
```

sys.getrefcount

```
In : d = []

In : sys.getrefcount(d)
Out: 2

In : x = d

In : sys.getrefcount(d)
Out: 3

In : del x

In : sys.getrefcount(d)
Out: 2
```

sys.getsizeof

```
In : for obj in ({}, [], (), 'string', 1, 12.3):
...:     print(obj.__class__.__name__,
               sys.getsizeof(obj))
...:
...:
dict 240
list 64
tuple 48
str 55
int 28
float 24
```

命令行参数 sys.argv

```
In : a, b  = 1, 2

In : a
Out: 1

In : a, b = range(3)
--------------------------------------------------------------------
ValueError                                Traceback (most recent cal
<ipython-input-125-c4cf1d48affb> in <module>()
----> 1 a, b = range(3)

ValueError: too many values to unpack (expected 2)

In : a, b, *c = range(3)

In : a, b, c
Out: (0, 1, [2])

In : a, *b, c = range(3)

In : a, b, c
Out: (0, [1], 2)
```

```python
import sys

script_name, *args = sys.argv

print(f'Script: {script_name}')
print(f'Arguments: {args}')
```

```
❯ python argv.py
Script: argv.py
Arguments: []

❯ python argv.py -v
Script: argv.py
Arguments: ['-v']

❯ python argv.py -v -v -e 'foo'
Script: argv.py
Arguments: ['-v', '-v', '-e', 'foo']
```

csv模块

```
In : import csv

In : with open('test.csv', 'wt') as f:
...:     writer = csv.writer(f)
...:     writer.writerow(('ID', '用户', '类型'))
...:     for i in range(3):
...:         row = (i, f'用户{i}', f'类型{i}')
...:         writer.writerow(row)
...:

In : cat test.csv
ID,用户,类型
0,用户0,类型0
1,用户1,类型1
2,用户2,类型2

In : with open('test.csv', 'rt') as f:
...:     reader = csv.reader(f)
...:     for line in reader:
...:         print(line)
...:
['ID', '用户', '类型']
['0', '用户0', '类型0']
['1', '用户1', '类型1']
['2', '用户2', '类型2']
```

```
In : with open('test.csv', 'rt') as f:
...:     reader = csv.DictReader(f)
...:     for line in reader:
...:         print(line)
...:         print(line['类型'])
...:
...:
OrderedDict([('ID', '0'), ('用户', '用户0'), ('类型', '类型0')])
类型0
OrderedDict([('ID', '1'), ('用户', '用户1'), ('类型', '类型1')])
类型1
OrderedDict([('ID', '2'), ('用户', '用户2'), ('类型', '类型2')])
类型2
```

datetime模块

```
In : import datetime

In : now = datetime.datetime.now()

In : now
Out: datetime.datetime(2018, 4, 2, 15, 56, 59, 5392

In : now.month, now.day, now.hour
Out: (4, 2, 15)

In : today = datetime.date.today()

In : today
Out: datetime.date(2018, 4, 2)

In : today.year, today.day
Out: (2018, 2)

In : d1 = datetime.date(2018, 3, 29)

In : d1
Out: datetime.date(2018, 3, 29)

In : datetime.datetime(2018, 3, 29)
Out: datetime.datetime(2018, 3, 29, 0, 0)
```

```
In : print('seconds     :', datetime.timedelta(seconds=1))
...: print('minutes     :', datetime.timedelta(minutes=1))
...: print('hours       :', datetime.timedelta(hours=1))
...: print('days        :', datetime.timedelta(days=1))
...: print('weeks       :', datetime.timedelta(weeks=1))
...:
seconds      : 0:00:01
minutes      : 0:01:00
hours        : 1:00:00
days         : 1 day, 0:00:00
weeks        : 7 days, 0:00:00

In : hour = datetime.timedelta(hours=1)

In : hour.total_seconds
Out: <function timedelta.total_seconds>

In : hour.total_seconds()
Out: 3600.0

In : today
Out: datetime.date(2018, 4, 2)

In : today + datetime.timedelta(days=1)
Out: datetime.date(2018, 4, 3)

In : today - datetime.timedelta(days=1)
Out: datetime.date(2018, 4, 1)
```

```
In : dt_format = '%Y-%m-%d %H:%M:%S'

In : s = now.strftime(dt_format)
...: print('strftime:', s)
...:
...:
strftime: 2018-04-02 15:56:59

In : d = datetime.datetime.strptime(s, dt_format)
...: print('strptime:', d)
...:
...:
strptime: 2018-04-02 15:56:59

In : d
Out: datetime.datetime(2018, 4, 2, 15, 56, 59)
```

random模块

```
In : for i in range(5):
...:     print(f'{random.random():.4f}', end='\t')
...:
0.1143  0.2254  0.3186  0.9311  0.1959

In : for i in range(5):
...:     print(f'{random.uniform(50, 60):.4f}', end='\t')
...:
...:
52.0958 57.2508 59.3443 53.8576 52.0433

In : t = time.time()

In : random.seed(t)

In : for i in range(5):
...:     print(f'{random.uniform(50, 60):.4f}', end='\t')
...:
56.5432 58.8500 56.4319 51.6879 54.9437

In : random.seed(t)

In : for i in range(5):
...:     print(f'{random.uniform(50, 60):.4f}', end='\t')
...:
56.5432 58.8500 56.4319 51.6879 54.9437
```

```
In : for i in range(5):
...:     print(f'{random.randint(1, 100)}', end='\t')
...:
...:
16    21      41      23      2

In : for i in range(5):
...:     print(f'{random.randrange(0, 101, 5)}', end='\t')
...:
40    20      20      90      25

In : random.sample(range(10), 3)
Out: [5, 2, 8]

In : random.choices(range(10), k=3)
Out: [6, 8, 6]

In : random.choice(['a', 'b', 'c'])
Out: 'a'
```

http.server模块

```
# 浏览器访问 http://localhost:8000/
❯ python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...

# 浏览器访问 http://localhost:9090/
❯ python -m http.server 9090
Serving HTTP on 0.0.0.0 port 9090 (http://0.0.0.0:9090/) ...

❯ python -m SimpleHTTPServer  # Python 2 用法
```

logging模块

```
日志级别      变量值
--------------
CRITICAL    50
ERROR       40
WARNING     30
INFO        20
DEBUG       10
NOTSET      0
```

```
In : import logging

In : logging.warning('Watch out!')
WARNING:root:Watch out!

In : logging.debug('This message won"t be printed')

In : logging.basicConfig(level=logging.WARNING)

In : logger1 = logging.getLogger('package1.module1')

In : logger2 = logging.getLogger('package2.module2')

In : logger1.warning('This message comes from module1')
WARNING:package1.module1:This message comes from module1

In : logger2.warning('This message comes from module2')
WARNING:package2.module2:This message comes from module2

In : logger2.debug('This message won"t be printed')
```

```python
import logging

logging.basicConfig(filename='myapp.log',
                    level=logging.INFO)
logging.info('Started')

print(logging.root.handlers)
```

```
❯ python loging_to_file.py

[<FileHandler /Users/dongweiming/sansa/
introduction-python/2.Python知识/20/
myapp.log (NOTSET)>]

❯ cat myapp.log
INFO:root:Started
```

```
In : logging.basicConfig(filename='myapp2.log',
                         level=logging.INFO)

In : logging.info('Started')

In : cat myapp2.log
cat: myapp2.log: No such file or directory

In : logging.root.handlers
Out: [<StreamHandler <stderr> (NOTSET)>]

In : logging.root.setLevel(logging.INFO)

In : logging.info('Started')
INFO:root:Started

In : cat myapp2.log
Started

In : logging.root.handlers
Out:
[<StreamHandler <stderr> (NOTSET)>,
 <FileHandler /Users/dongweiming/
avalon/myapp2.log (NOTSET)>]
```

最佳使用logging的方案

```
In : import logging
...:
...: logger = logging.getLogger()
...: handler = logging.StreamHandler()
...: formatter = logging.Formatter(
...:         '%(asctime)s %(name)-12s %(levelname)-8s %(message)s')
...: handler.setFormatter(formatter)
...: logger.addHandler(handler)
...: logger.setLevel(logging.DEBUG)
...:
...: logger.debug('This is a %s', 'test')
...:
DEBUG:root:This is a test
2018-04-02 18:34:08,443 root         DEBUG    this is a test
```

内置日志格式

%(name)s	生成日志的Logger名称。
%(levelno)s	数字形式的日志级别，包括DEBUG，INFO，WARNING，ERROR和CRITICAL。
%(levelname)s	文本形式的日志级别，包括’DEBUG’、 ‘INFO’、 ‘WARNING’、 ‘ERROR’ 和’CRITICAL’。
%(pathname)s	输出该日志的语句所在源文件的完整路径（如果可用）。
%(filename)s	文件名。
%(module)s	输出该日志的语句所在的模块名。
%(funcName)s	调用日志输出函数的函数名。
%(lineno)d	调用日志输出函数的语句所在的代码行（如果可用）。
%(created)f	日志被创建的时间，UNIX标准时间格式，表示从1970-1-1 00:00:00 UTC计算起的秒数。
%(relativeCreated)d	日志被创建时间与日志模块被加载时间的时间差，单位为毫秒。
%(asctime)s	日志创建时间。默认格式是 “2003-07-08 16:49:45,896”，逗号后为毫秒数。
%(msecs)d	毫秒级别的日志创建时间。
%(thread)d	线程ID（如果可用）。
%(threadName)s	线程名称（如果可用）。
%(process)d	进程ID（如果可用）。
%(message)s	日志信息。

延伸阅读

1.https://pymotw.com/3/
2.https://pymotw.com/2/
3.https://www.python.org/dev/peps/pep-3132/
4.https://docs.python.org/3/library/datetime.html#strftime-and-strptime-behavior
5.https://zh.wikipedia.org/wiki/%E6%A2%85%E6%A3%AE%E6%97%8B%E8%BD%AC%E7%AE%97%E6%B3%95
6.https://www.python.org/dev/peps/pep-3101/
7.https://www.python.org/dev/peps/pep-0282/