
pydu Documentation

Prodesire

12 07, 2017

Contents

1	1
2	3

CHAPTER 1

pydu python data structures and utils — python Python 23

Python **2.7** **3.6**

2.1 Archive

`pydu.archive.extract(path, to_path='', ext='')`
tarzip to_path .tar.tar.bz2.tar.gz.tgz.tz2.zip path ext

```
>>> from pydu.archive import extract
>>> extract('foobar.tgz', '/tmp')
>>> extract('foobar', '/tmp', ext='.tgz')
>>> extract('foobar', '/tmp')
Traceback (most recent call last):
...   AttributeError: pydu.archive.UnrecognizedArchiveFormat: Path not a_
↪recognized archive format: foobar
```

2.2 Cmd

`pydu.cmd.execute(cmd, wait=True, shell=True)`
subprocess.Popen wait True execute (returncode, stdout) stderr stdout wait False
execute Popen shell Popen

```
>>> from pydu.cmd import execute
>>> execute('echo hello')
(0, b'hello\r\n') # Python 3
>>> execute('echo hello', wait=False)
<subprocess.Popen at 0x22e4010f9e8>
```

`pydu.cmd.cmdline_argv()`
PythonWindowsPython 2 cmdline_argv shell32.GetCommandLineArgvW Unicodesys.argvPython
3 cmdline_argv sys.argv
PyCharmPython

```
>>> from pydu.cmd import cmdline_argv
>>> cmdline_argv()
['/Applications/PyCharm.app/Contents/helpers/pydev/pydevconsole.py', '61253',
↪ '61254']
```

2.3 Compat

```
pydu.compat.PY2
    PythonPython 23

pydu.compat.urljoin(base, url, allow_fragments=True)
    PY2 urlparse.urljoinPY3 urllib.parse.urljoin

pydu.compat.iterkeys(d)
    iter

pydu.compat.itervalues(d)
    iter

pydu.compat.iteritems(d)
    iter

pydu.compat.text_type
    PY2 unicodePY3 str

pydu.compat.string_types
    PY2 (str, unicode)PY3 (str,)

pydu.compat.numeric_types
    PY2 (int, long)PY3 (int,)

pydu.compat.imap()
    PY2 itertools.imapPY3 map

pydu.compat.has_next_attr(x)
```

```
>>> from pydu.compat import is_iterable
>>> is_iterable([])
True
>>> is_iterable(1)
False
```

2.4 Console

```
pydu.console.console_size()
    Windows(width, height)(80, 25)POSIX(width, height)IOError(80, 25)(80, 25)
```

```
>>> from pydu.console import console_size
>>> console_size()
(80, 25)
```


2.5 Dict

```
class pydu.dict.AttrDict (seq=None, **kwargs)
    AttrDict obj['foo'] obj.foo
```

```
>>> from pydu.dict import AttrDict
>>> o = AttrDict(a=1)
o.a
1
>>> o['a']
1
>>> o.a = 2
>>> o['a']
2
>>> del o.a
>>> o.a
Traceback (most recent call last):
... AttributeError: 'a'
```

```
class pydu.dict.CaseInsensitiveDict (data=None, **kwargs)
    collections.MutableMapping copy lower_itemsiter(instance) keys() items() iterkeys()
    iteritems()
```

```
>>> from pydu.dict import CaseInsensitiveDict
>>> cid = CaseInsensitiveDict()
>>> cid['Accept'] = 'application/json'
>>> cid['aCCEPT'] == 'application/json'
True
>>> list(cid) == ['Accept']
True
```

```
class pydu.dict.LookupDict (data=None)
    pydu.dict.LookupDict(data=None) dict
>>> d = LookupDict()
>>> d['key']
None
>>> d['key'] = 1
>>> d['key']
1
```

```
pydu.dict.attrify (obj)
    AttrDict AttrDict
```

```
>>> from pydu.dict import attrify
>>> attrd = attrify({
    'a': [1, 2, {'b': 'b'}],
    'c': 'c',
})
>>> attrd
<AttrDict {'a': [1, 2, <AttrDict {'b': 'b'}>], 'c': 'c'}>
>>> attrd.a
1
>>> attrd.a[2].b
b
>>> attrd.c
c
```

2.6 Inspect

`pydu.inspect.getargspec(func)`
 (args, vargs, varkw, defaults) args varargs varkw * ** None defaults n

```
>>> from pydu.inspect import getargspec
>>> def f(name, address='home', age=25, *args, **kwargs):
...     pass
...
>>> getargspec(f)
ArgSpec(args=['name', 'address', 'age'], varargs='args', keywords='kwargs',
defaults=('home', 25))
```

`pydu.inspect.get_func_args(func)`
 *args *kwargs

```
>>> from pydu.inspect import get_func_args
>>> def f(name, address='home', age=25, *args, **kwargs):
...     pass
...
>>> get_func_args(f)
['name', 'address', 'age']
```

`pydu.inspect.get_func_full_args(func)`
 (,) *args *kwargs

```
>>> from pydu.inspect import get_func_full_args
>>> def f(name, address='home', age=25, *args, **kwargs):
...     pass
...
>>> get_func_full_args(f)
[('name',), ('address', 'home'), ('age', 25), ('*args',), ('**kwargs',)]
```

`pydu.inspect.func_accepts_kwargs(func)`
 True

```
>>> from pydu.inspect import func_accepts_kwargs
>>> def f(**kwargs):
...     pass
...
>>> func_accepts_kwargs(f)
True
```

`pydu.inspect.func_accepts_var_args(func)`
 True

```
>>> from pydu.inspect import func_accepts_var_args
>>> def f(*vargs):
...     pass
...
>>> func_accepts_var_args(f)
True
```

`pydu.inspect.func_supports_parameter(func)`
 True
 False

```
>>> from pydu.inspect import func_supports_parameter
>>> def f(name):
...     pass
...
>>> func_supports_parameter(f, 'name')
True
>>> func_supports_parameter(f, 'unkown')
False
```

```

pydu>>> from pydu.misc import func_has_no_args
>>> def f():
...     pass
...
>>> func_has_no_args(f)
True

```

2.7 Miscellanea

`pydu.misc.unix_timeout(seconds)`
hang seconds *test.py*

```

import time
from pydu.misc import unix_timeout
@unix_timeout(1)
def f():
    time.sleep(1.01)
f()

```

test.py TimeoutError

: unix_timeout Unix

`pydu.misc.trace(obj)`
bash -x *test.py*

```

from pydu.misc import trace
@trace
def f():
    print(1)
    a = 1 + 5
    b = [a]
    print(2)
f()

```

test.py

```

test.py(4):      print(1)
1
test.py(5):      a = 1 + 5
test.py(6):      b = [a]
test.py(7):      print(2)
2

```

`pydu.misc.memoize(obj)`
cache_clear()

```

>>> @memoize
... def foo():
...     return 1
...
>>> foo()
1

```

```
>>> foo.cache_clear()
>>>
```

pydu>>> **class** **memoize_when_activated**(obj)

```
...     @memoize
...     def foo():
...         print(1)
...
>>> f = Foo()
>>> # deactivated (default)
>>> foo()
1
>>> foo()
1
>>>
>>> # activated
>>> foo.cache_activate()
>>> foo()
1
>>> foo()
>>> foo()
>>>
```

2.8 Network

pydu.network.**dotted_netmask**(mask)
mask /xx xxx.xxx.xxx.xxx mask int str

```
>>> from pydu.network import dotted_netmask
>>> dotted_netmask('24')
'255.255.255.0'
>>> dotted_netmask(24)
'255.255.255.0'
```

pydu.network.**is_ipv4_address**(ip)
ip IPV4

```
>>> from pydu.network import is_ipv4_address
>>> is_ipv4_address('8.8.8.8')
True
>>> is_ipv4_address('localhost.localdomain')
False
```

2.9 Request

class pydu.request.**Filename**
unicode None

from_headers(headers):
Content-Disposition headers

pydu.request.**download**(url, dst=None)
URLURLHTTPurl URL dst None

from_url(url)

2.10 Set

```
class pydu.set.OrderedSet(iterable=None)
>>> s = OrderedSet([1, 3, 1, 2])
>>> list(s)
[1, 3, 2]
>>> s.discard(3)
>>> list(s)
[1, 2]
```

2.11 String

`pydu.string.safeunicode(obj, encoding='utf-8')`
 unicode

```
>>> from pydu.string import safeunicode
>>> safeunicode('hello')
u'hello'
>>> safeunicode(2)
u'2'
>>> safeunicode('\xe4\xb8\xad\xe6\x96\x87')
u''
```

`pydu.string.safeencode(obj, encoding='utf-8')`
 utf-8

```
>>> from pydu.string import safeencode
>>> safeencode('hello')
'hello'
>>> safeencode(2)
'2'
>>> safeencode(u'')
'\xe4\xb8\xad\xe6\x96\x87'
```

`pydu.string.lstrips(text, remove)`
 text remove

```
>>> from pydu.string import lstrips
>>> lstrips('foobar', 'foo')
'bar'
>>> lstrips('FOOBARBAZ', ['FOO', 'BAR'])
'BAZ'
>>> lstrips('FOOBARBAZ', ['BAR', 'FOO'])
'BARBAZ'
```

`pydu.string.rstrips(text, remove)`
 text remove

```
>>> from pydu.string import rstrips
>>> rstrips('foobar', 'bar')
'foo'
```

`pydu.string.strips(text, remove)`
 text remove

```
>>> from pydu.string import strips
>>> strips('foobarfoo', 'foo')
'bar'
```