Porting Notes

This section includes notes and tips for updating from Python 2 to Python 3, including summaries of and references for the changes in each module.

References

The notes in this section are based on the *What's New* documents prepared by the Python development team and release manager for each release.

- What's New In Python 3.0
- What's New In Python 3.1
- What's New In Python 3.2
- What's New In Python 3.3
- What's New In Python 3.4
- What's New In Python 3.5

For more information about porting to Python 3, refer to

- Porting Python 2 Code to Python 3 on `docs.python.org`.
- Porting to Python 3, by Lennart Regebro.
- The python-porting mailing list.

New Modules

Python 3 includes a number of new modules, providing features not present in Python 2.

`asyncio`
: Asynchronous I/O, event loop, and other concurrency tools

`concurrent.futures`
: Managing Pools of Concurrent Tasks

`ensurepip`
: Install the Python Package Installer, pip

`enum`
: Defines enumeration type

`ipaddress`
: Classes for working with Internet Protocol (IP) addresses

`pathlib`
: An object-oriented API for working with filesystem paths

`selectors`
: I/O Multiplexing Abstractions

`statistics`
: Statistical Calculations

`venv`
: Create isolated installation and execution contexts

Renamed Modules

Many standard library modules were renamed between Python 2 and 3 as part of PEP 3108. All of the new module names consistently use lowercase, and some have been moved into packages to better organize related modules. Often, code using these modules can be updated to work with Python 3 just by fixing the import statements. A complete list of the renames can be found in the dictionary `lib2to3.fixes.fix_imports.MAPPING` (the keys are the Python 2 name and the values are the Python 3 name) and in the table below.

Renamed Modules

Python 2 Name	Python 3 Name
`__builtin__`	`builtins`
`_winreg`	`winreg`
`BaseHTTPServer`	`http.server`
`CGIHTTPServer`	`http.server`
`commands`	`subprocess`
`ConfigParser`	`configparser`

Cookie	http.cookies
cookielib	http.cookiejar
copy_reg	copyreg
cPickle	pickle
cStringIO	io
dbhash	dbm.bsd
dbm	dbm.ndbm
Dialog	tkinter.dialog
DocXMLRPCServer	xmlrpc.server
dumbdbm	dbm.dumb
FileDialog	tkinter.filedialog
gdbm	dbm.gnu
htmlentitydefs	html.entities
HTMLParser	html.parser
httplib	http.client
Queue	queue
repr	reprlib
robotparser	urllib.robotparser
ScrolledText	tkinter.scrolledtext
SimpleDialog	tkinter.simpledialog
SimpleHTTPServer	http.server
SimpleXMLRPCServer	xmlrpc.server
SocketServer	socketserver
StringIO	io
Tix	tkinter.tix
tkColorChooser	tkinter.colorchooser
tkCommonDialog	tkinter.commondialog
Tkconstants	tkinter.constants
Tkdnd	tkinter.dnd
tkFileDialog	tkinter.filedialog
tkFont	tkinter.font
Tkinter	tkinter
tkMessageBox	tkinter.messagebox
tkSimpleDialog	tkinter.simpledialog
ttk	tkinter.ttk
urlparse	urllib.parse
UserList	collections
UserString	collections
xmlrpclib	xmlrpc.client

See also

- The six package is useful for writing code that runs under both Python 2 and 3. In particular, the `six.moves` module allows your code to import renamed modules using a single import statement, automatically redirecting the import to the correct version of the name depending on the version of Python.
- **PEP 3108** – Standard Library Reorganization

Removed Modules

These modules are either no longer present at all, or have had their features merged into other existing modules.

bsddb

The `bsddb` and `dbm.bsd` modules have been removed. Bindings for Berkeley DB are now maintained outside of the standard library as `bsddb3`.

commands

The `commands` module was deprecated in Python 2.6 and removed in Python 3.0. See `subprocess` instead.

compiler

The `compiler` module has been removed. See `ast` instead.

dircache

The `dircache` module has been removed, without a replacement.

EasyDialogs

The `EasyDialogs` module has been removed. See `tkinter` instead.

exceptions

The `exceptions` module has been removed because all of the exceptions defined there are available as built-in classes.

htmllib

The `htmllib` module has been removed. See `html.parser` instead.

md5

The implementation of the MD5 message digest algorithm has moved to `hashlib`.

mimetools, MimeWriter, mimify, multifile, and rfc822

The `mimetools`, `MimeWriter`, `mimify`, `multifile`, and `rfc822` modules have been removed. See `email` instead.

popen2

The `popen2` module has been removed. See `subprocess` instead.

posixfile

The `posixfile` module has been removed. See `io` instead.

sets

The `sets` module was deprecated in Python 2.6 and removed in Python 3.0. Use the built-in types `set` and `orderedset` instead.

sha

The implementation of the SHA-1 message digest algorithm has moved to `hashlib`.

sre

The `sre` module was deprecated in Python 2.5 and removed in Python 3.0. Use `re` instead.

statvfs

The `statvfs` module was deprecated in Python 2.6 and removed in Python 3.0. See `os.statvfs()` in the `os` module instead.

thread

The `thread` module has been removed. Use the higher-level API in `threading` instead.

user

The user module was deprecated in Python 2.6 and removed in Python 3.0. See user-customization features provided by the site module instead.

Deprecated Modules

These modules are still present in the standard library, but are deprecated and should not be used in new Python 3 programs.

asyncore and asynchat

Asynchronous I/O and protocol handlers. See asyncio instead.

formatter

Generic output formatter and device interface. See Python issue 18716 for details.

imp

Access the implementation of the import statement. See importlib instead.

optparse

Command-line option parsing library. The API for argparse is similar to the one provided by optparse, and in many cases argparse can be used as a straightforward replacement by updating the names of the classes and methods used.

Summary of Changes to Modules

abc

The abstractproperty(), abstractclassmethod(), and abstractstaticmethod() decorators are deprecated. Combining abstractmethod() with the property(), classmethod(), and staticmethod() decorators works as expected (Python issue 11610).

anydbm

The anydbm module has been renamed dbm in Python 3.

argparse

The version argument to ArgumentParser has been removed in favor of a special action type (Python issue 13248).

The old form passed version as an argument.

```
parser = argparse.ArgumentParser(version='1.0')
```

The new form requires adding an explicit argument definition.

```
parser = argparse.ArgumentParser()
parser.add_argument('--version', action='version',
                    version='%(prog)s 1.0')
```

The option name and version format string can be modified to suit the needs of the application.

In Python 3.4, the version action was changed to print the version string to stdout instead of stderr (Python issue 18920).

array

The 'c' type used for character bytes in early version of Python 2 has been removed. Use 'b' or 'B' for bytes instead.

The 'u' type for characters from Unicode strings has been deprecated and will be removed in Python 4.0.

The methods tostring() and fromstring() have been renamed tobytes() and frombytes() to remove ambiguity (Python issue 8990).

atexit

When atexit was updated to include a C implementation (Python issue 1680961), a regression was introduced in the error handling logic that caused only the summary of the exception to be shown, without the traceback. This regression was fixed in Python 3.3 (Python issue 18776).

base64

The `encodestring()` and `decodestring()` have been renamed `encodebytes()` and `decodebytes()` respectively. The old names still work as aliases, but are deprecated (Python issue 3613).

Two new encodings using 85-character alphabets have been added. `b85encode()` implements an encoding used in Mercurial and git, while `a85encode()` implements the Ascii85 format used by PDF files (Python issue 17618).

bz2

`BZ2File` instances now support the context manager protocol, and do not need to be wrapped with `contextlib.closing()`.

collections

The abstract base classes formerly defined in `collections` moved to `collections.abc`, with backwards-compatibility imports in `collections`, for now (Python issue 11085).

comands

The functions `getoutput()` and `getstatusoutput()` have been moved to `subprocess` and `commands` has been deleted.

configparser

The old `ConfigParser` module has been renamed to `configparser`.

The old `ConfigParser` class was removed in favor of `SafeConfigParser` which has in turn been renamed to `ConfigParser`. The deprecated interpolation behavior is available via `LegacyInterpolation`.

The `read()` method now supports an `encoding` argument, so it is no longer necessary to use `codecs` to read configuration files with Unicode values in them.

Using the old `RawConfigParser` is discouraged. New projects should use `ConfigParser(interpolation=None)` instead to achieve the same behavior.

contextlib

`contextlib.nested()` has been removed. Pass multiple context managers to the same `with` statement instead.

csv

Instead of using the `next()` method of a reader directly, use the built-in `next()` function to invoke the iterator properly.

datetime

Starting with Python 3.3, equality comparisons between naive and timezone-aware `datetime` instances return `False` instead of raising `TypeError` (Python issue 15006).

Prior to Python 3.5, a `datetime.time` object representing midnight evaluated to `False` when converted to a Boolean. This behavior has been removed in Python 3.5 (Python issue 13936).

decimal

Python 3.3 incorporated a C implementation of `decimal` based on `libmpdec`. This change improved performance, but also includes some API changes and behavior differences from the pure-Python implementation. See the Python 3.3 release notes for details.

fractions

The `from_float()` and `from_decimal()` class methods are no longer needed. Floating point and `Decimal` values can be passed directly to the `Fraction` constructor.

gc

The flags `DEBUG_OBJECT` and `DEBUG_INSTANCE` have been removed. They are no longer needed to differentiate between new and old-style classes.

gettext

All of the translation functions in `gettext` assume Unicode input and output, and the Unicode variants such as `ugettext()`

have been removed.

glob

The new function `escape()` implements a work-around for searching for files with meta-characters in the name (Python issue 8402).

http.cookies

In addition to escaping quotes, SimpleCookie also encodes commas and semi-colons in values to better reflect behavior in real browsers (Python issue 9824).

imaplib

Under Python 3, `imaplib` returns byte-strings encoded as UTF-8. There is support for accepting Unicode strings and encoding them automatically as outgoing commands are sent or as the username and password for logging in to the server.

importlib

The `find_loader()` function is deprecated. Use `importlib.util.find_spec()` instead.

inspect

The functions `getargspec()`, `getfullargspec()`, `getargvalues()`, `getcallargs()`, `getargvalues()`, `formatargspec()`, and `formatargvalues()` have been deprecated in favor of `signature()` (Python issue 20438).

itertools

The functions `imap()`, `izip()`, and `ifilter()` have been replaced with versions of the built-in functions that return iterables instead of `list` objects (`map()`, `zip()`, and `filter:()` respectively).

The function `ifilterfalse()` has been renamed `filterfalse()`.

json

The `json` API was updated to only support `str` and not with `bytes` because the JSON specification is defined using Unicode.

locale

The normalized version of the name of the UTF-8 encoding has changed from “UTF8” to “UTF-8” because Mac OS X and OpenBSD do not support the use of “UTF8” (Python issue 10154 and Python issue 10090).

logging

The `logging` module now includes a `lastResort` logger that is used if no other logging configuration is performed by an application. This eliminates the need to configure logging in an application solely to avoid having a user see error messages in case a library imported by an application uses logging but the application itself does not.

mailbox

mailbox reads and writes mailbox files in binary mode, relying on the email package to parse messages. StringIO and text file input is deprecated (Python issue 9124).

mmap

Values returned from read APIs are byte strings, and need to be decoded before being treated as text.

operator

The `div()` function has been removed. Use either `floordiv()` or `truediv()`, depending on the desired semantics.

The `repeat()` function is removed. Use `mul()` instead.

The functions `getslice()`, `setslice()`, and `delslice()` are removed. Use `getitem()`, `setitem()`, and `delitem()` with slice indexes instead.

The function `isCallable()` has been removed. Use the abstract base class `collections.Callable` instead.

```
isinstance(obj, collections.Callable)
```

The type checking functions `isMappingType()`, `isSequenceType()`, and `isNumberType()` have been removed. Use the relevant abstract base classes from `collections` or `numbers` instead.

```
isinstance(obj, collections.Mapping)
isinstance(obj, collections.Sequence)
isinstance(obj, numbers.Number)
```

The `sequenceIncludes()` function has been removed. Use `contains()` instead.

os

The functions `popen2()`, `popen3()`, and `popen4()` have been removed. `popen()` is still present but deprecated and emits warnings if used. Code using these functions should be rewritten to use `subprocess` instead to be more portable across operating systems.

The functions `os.tmpnam()`, `os.tempnam()` and `os.tmpfile()` have been removed. Use the `tempfile` module instead.

The function `os.stat_float_times()` is deprecated (Python issue 14711).

`os.unsetenv()` no longer ignores errors (Python issue 13415).

os.path

`os.path.walk()` has been removed. Use `os.walk()` instead.

pdb

The `print` command alias has been removed so that it does not shadow the `print()` function (Python issue 18764). The `p` shortcut is retained.

pickle

The C implementation of the pickle module from Python 2 has been moved to a new module that is automatically used to replace the Python implementation when possible. The old import idiom of looking for `cPickle` before `pickle` is no longer needed.

```
try:
   import cPickle as pickle
except:
   import pickle
```

With the automatic import of the C implementation, it is only necessary to import the `pickle` module directly.

```
import pickle
```

Interoperability between Python 2.x and 3.x has been improved for pickled data using the level 2 protocol or lower to resolve an issue introduced when a large number of standard library modules were renamed during the transition to Python 3. Because pickled data includes references to class and type names, and those names changed, it was difficult to exchange pickled data between Python 2 and 3 programs. Now for data pickled using protocol level 2 or older, the old names of the classes are automatically used when writing to and reading from a pickle stream.

This behavior is available by default, and can be turned off using the `fix_imports` option. This change improves the situation, but does not eliminate incompatibilities entirely. In particular, it is possible that data pickled under Python 3.1 can't be read under Python 3.0. To ensure maximum portability between Python 3 applications, use protocol level 3, which does not include this compatibility feature.

The default protocol version has changed from `0`, the human-readable version, to 3, the binary format with the best interoperability when shared between Python 3 applications.

Byte string data written to a pickle by a Python 2.x application is decoded when it is read back to create a Unicode string object. The encoding for the transformation defaults to ASCII, and can be changed by passing values to the `Unpickler`.

pipes

`pipes.quote()` has moved to `shlex` (Python issue 9723).

platform

`platform.popen()` has been deprecated. Use `subprocess.popen()` instead (Python issue 11377).

`platform.uname()` now returns a `namedtuple`.

Because Linux distributions do not have a consistent way to describe themselves, the functions for getting the descriptions (`platform.dist()` and `platform.linux_distribution()`) are deprecated and scheduled to be removed in Python 3.7 (Python issue 1322).

random

The function `jumpahead()` was removed in Python 3.0.

re

The `UNICODE` flag represents the default behavior. To restore the ASCII-specific behavior from Python 2, use the `ASCII` flag.

shelve

The default output format for `shelve` may create a file with a `.db` extension added to the name given to `shelve.open()`.

signal

PEP 475 means that system calls interrupted and returning with `EINTR` are retried. This changes the behavior of signal handlers and other system calls, since now after the signal handler returns the interrupted call will be retried, unless the signal handler raises an exception. Refer to the PEP documentation for complete details.

socket

Under Python 2 typically `str` objects could be sent directly over a socket. Because `str` replaces `unicode`, in Python 3 the values must be encoded before being sent. The examples in the `socket` section use byte strings, which are already encoded.

socketserver

The `socketserver` module was named `SocketServer` under Python 2.

string

All functions from the `string` module that are also methods of `str` objects have been removed.

The constants `letters`, `lowercase`, and `uppercase` have been removed. The new constants with similar names are limited to the ASCII character set.

The `maketrans()` function has been replaced by methods on `str`, `bytes`, and `bytearray` to clarify which input types are supported by each translation table.

struct

`struct.pack()` now only supports byte strings when using the `s` string pack code, and no longer implicitly encodes string objects to UTF-8 (Python issue 10783).

subprocess

The default value for the `close_fds` argument to `subprocess.Popen` has changed from always being `False`. It always defaults to `True` under Unix. It defaults to `True` under Windows if the standard I/O stream arguments are set to `None`, otherwise it defaults to `False`.

sys

The variable `sys.exitfunc` is no longer checked for a clean-up action to be run when a program exits. Use `atexit` instead.

The variable `sys.subversion` is no longer defined.

Flags `sys.flags.py3k_warning`, `sys.flags.division_warning`, `sys.flags.division_new`, `sys.flags.tabcheck`, and `sys.flags.unicode` are no longer defined.

The variable `sys.maxint` is no longer defined, use `sys.maxsize` instead. See **PEP 237** (Unifying Long Integers and Integers).

The global exception tracking variables `sys.exc_type`, `sys.exc_value`, and `sys.exc_traceback` have been removed. The function `sys.exc_clear()` has also been removed.

The variable `sys.version_info` is now a :py``namedtuple`` instance with attributes `major`, `minor`, `micro`, `releaselevel`, and `serial` (Python issue 4285).

The check interval feature, controlling the number of opcodes to execute before allowing a thread context switch has been replaced with an absolute time value instead, managed with `sys.setswitchinterval()`. The old functions for managing the check interval, `sys.getcheckinterval()` and `sys.setcheckinterval()`, are deprecated.

The `sys.meta_path` and `sys.path_hooks` variables now expose all of the path finders and entry hooks for importing modules. In earlier versions, only finders and hooks explicitly added to the path were exposed, and the C import used values in its implementation that could not be modified from the outside.

For Linux systems, `sys.platform` no longer includes the version number. The value is now just `linux` and not `linux2` or `linux3`.

threading

The `thread` module is deprecated in favor of the API in `threading`.

The debugging features of `threading`, including the "verbose" argument has been removed from the APIs (Python issue 13550).

Older implementations of `threading` used factory functions for some of the classes because they were implemented in C as extension types and could not be subclassed. That limitation of the language has been removed, and so many of the old factory functions have been converted to standard classes, which allow subclassing (Python issue 10968).

The public symbols exported from `threading` have been renamed to be **PEP 8** compliant. The old names are retained for backwards compatibility, but they will be removed in a future release.

time

`time.asctime()` and `time.ctime()` have been reimplemented to not use the system functions of the same time to allow larger years to be used. `time.ctime()` now supports years from 1900 through `maxint`, although for values higher than 9999 the output string is longer than the standard 24 characters to allow for the extra year digits (Python issue 8013).

unittest

The `TestCase` methods starting with "fail" (`failIf()`, `failUnless()`, etc.) have been deprecated. Use the alternate form of the assert methods instead.

Several older method aliases have been deprecated and replaced with preferred names. Using the deprecated names produces a warning (Python issue 9424). See the table below for a mapping between the old and new names.

Deprecated unittest.TestCase Methods

Deprecated Name	Preferred Name
`assert_()`	`assertTrue()`
`assertEquals()`	`assertEqual()`
`assertNotEquals()`	`assertNotEqual()`
`assertAlmostEquals()`	`assertAlmostEqual()`
`assertNotAlmostEquals()`	`assertNotAlmostEqual()`

UserDict, UserList, and UserString

The UserDict, UserList, and UserString classes have been moved out of their own modules into the `collections` module. `dict`, `list`, and `str` can be subclassed directly, but the classes in `collections` may make implementing the subclass simpler because the content of the container is available directly through an instance attribute. The abstract classes in `collections.abc` are also useful for creating custom containers that follow the APIs of the built-in types.

uuid

`uuid.getnode()` now uses the PATH environment variable to find programs that can report the MAC address of the host under Unix (Python issue 19855). It falls back to looking in `/sbin` and `/usr/sbin` if no program is found on the search path. This search behavior may give different results than with earlier versions of Python if alternate versions of programs like `netstat`, `ifconfig`, `ip`, and `arp` are present and produce different output.

whichdb

The functionality of `whichdb` has moved to the `dbm` module.

xml.etree.ElementTree

`XMLTreeBuilder` has been renamed `TreeBuilder`, and the API has undergone several changes.

`ElementTree.getchildren()` has been deprecated. Use `list(elem)` to build a list of the children.

`ElementTree.getiterator()` has been deprecated. Use `iter()` to create an iterator using the normal iterator protocol instead.

When parsing fails, rather than raising `xml.parsers.expat.ExpatError`, `XMLParser` now raises `xml.etree.ElementTree.ParseError`.

zipimport

The data returned from `get_data()` is a byte string, and needs to be decoded before being used as a unicode string.

grp — Unix Group Database

Outside of the Standard Library

Quick Links
References
New Modules
Renamed Modules
Removed Modules
Deprecated Modules
Summary of Changes to Modules

This page was last updated 2018-12-09.

Navigation
grp — Unix Group Database
Outside of the Standard Library

The output from all the example programs from PyMOTW-3 has been generated with Python 3.7.1, unless otherwise noted. Some of the features described here may not be available in earlier versions of Python.

Looking for examples for Python 2?