

## Navigation

- [index](#)
- [modules](#) |
- [next](#) |
- [previous](#) |
- [Python v2.6.4 documentation](#) »
- [The Python Standard Library](#) »
- [17. Optional Operating System Services](#) »

## 17.5. dummy\_thread — Drop-in replacement for the [thread](#) module¶

### Note

The `dummy_thread` module has been renamed to `_dummy_thread` in Python 3.0. The [2to3](#) tool will automatically adapt imports when converting your sources to 3.0; however, you should consider using the high-level [dummy\\_threading](#) module instead.

This module provides a duplicate interface to the [thread](#) module. It is meant to be imported when the [thread](#) module is not provided on a platform.

Suggested usage is:

```
try:
    import thread as _thread
except ImportError:
    import dummy_thread as _thread
```

Be careful to not use this module where deadlock might occur from a thread being created that blocks waiting for another thread to be created. This often occurs with blocking I/O.

### Previous topic

[17.4. dummy\\_threading — Drop-in replacement for the `threading` module](#)

### Next topic

[17.6. multiprocessing — Process-based “threading” interface](#)

### This Page

- [Show Source](#)

## Navigation

- [index](#)
- [modules](#) |
- [next](#) |
- [previous](#) |
- [Python v2.6.4 documentation](#) »
- [The Python Standard Library](#) »
- [17. Optional Operating System Services](#) »

© [Copyright](#) 1990-2010, Python Software Foundation.

The Python Software Foundation is a non-profit corporation. [Please donate.](#)

Last updated on Feb 26, 2010. Created using [Sphinx](#) 0.6.3.