## 36.12. `posixfile` — File-like objects with locking support¶

*Platforms:* Unix

Deprecated since version 1.5: The locking operation that this module provides is done better and more portably by the `fcntl.lockf()` call.

This module implements some additional functionality over the built-in file objects. In particular, it implements file locking, control over the file flags, and an easy interface to duplicate the file object. The module defines a new file object, the posixfile object. It has all the standard file object methods and adds the methods described below. This module only works for certain flavors of Unix, since it uses `fcntl.fcntl()` for file locking.

To instantiate a posixfile object, use the `posixfile.open()` function. The resulting object looks and feels roughly the same as a standard file object.

The `posixfile` module defines the following constants:

`posixfile.SEEK_SET`¶

Offset is calculated from the start of the file.

`posixfile.SEEK_CUR`¶

Offset is calculated from the current position in the file.

`posixfile.SEEK_END`¶

Offset is calculated from the end of the file.

The `posixfile` module defines the following functions:

`posixfile.open`(*filename*[, *mode*[, *bufsize*]])¶

Create a new posixfile object with the given filename and mode. The *filename*, *mode* and *bufsize* arguments are interpreted the same way as by the built-in `open()` function.

`posixfile.fileopen`(*fileobject*)¶

Create a new posixfile object with the given standard file object. The resulting object has the same filename and mode as the original file object.

The posixfile object defines the following additional methods:

`posixfile.lock`(*fmt*[, *len*[, *start*[, *whence*]]])¶

Lock the specified section of the file that the file object is referring to. The format is explained below in a table. The *len* argument specifies the length of the section that should be locked. The default is `0`. *start* specifies the starting offset of the section, where the default is `0`. The *whence* argument specifies where the offset is relative to. It accepts one of the constants `SEEK_SET`, `SEEK_CUR` or `SEEK_END`. The default is `SEEK_SET`. For more information about the arguments refer to the *fcntl(2)* manual page on your system.

`posixfile.flags`([*flags*])¶

Set the specified flags for the file that the file object is referring to. The new flags are ORed with the old flags, unless specified otherwise. The format is explained below in a table. Without the *flags* argument a string indicating the current flags is returned (this is the same as the `?` modifier). For more information about the flags refer to the *fcntl(2)* manual page on your system.

`posixfile.dup`()¶

Duplicate the file object and the underlying file pointer and file descriptor. The resulting object behaves as if it were newly opened.

`posixfile.dup2`(*fd*)¶

Duplicate the file object and the underlying file pointer and file descriptor. The new object will have the given file descriptor. Otherwise the resulting object behaves as if it were newly opened.

`posixfile.file`()¶

Return the standard file object that the posixfile object is based on. This is sometimes necessary for functions that insist on a standard file object.

All methods raise `IOError` when the request fails.

Format characters for the `lock()` method have the following meaning:

| Format | Meaning |
|---|---|
| `u` | unlock the specified region |
| `r` | request a read lock for the specified section |
| `w` | request a write lock for the specified section |

In addition the following modifiers can be added to the format:

| Modifier | Meaning | Notes |
|---|---|---|
| \| | wait until the lock has been granted | |
| ? | return the first lock conflicting with the requested lock, or `None` if there is no conflict. | (1) |

Note:

1. The lock returned is in the format `(mode, len, start, whence, pid)` where *mode* is a character representing the type of lock ('r' or 'w'). This modifier prevents a request from being granted; it is for query purposes only.

Format characters for the `flags()` method have the following meanings:

| Format | Meaning |
|---|---|
| a | append only flag |
| c | close on exec flag |
| n | no delay flag (also called non-blocking flag) |
| s | synchronization flag |

In addition the following modifiers can be added to the format:

| Modifier | Meaning | Notes |
|---|---|---|
| ! | turn the specified flags 'off', instead of the default 'on' | (1) |
| = | replace the flags, instead of the default 'OR' operation | (1) |
| ? | return a string in which the characters represent the flags that are set. | (2) |

Notes:

1. The `!` and = modifiers are mutually exclusive.
2. This string represents the flags after they may have been altered by the same call.

Examples:

```
import posixfile

file = posixfile.open('/tmp/test', 'w')
file.lock('w|')
...
file.lock('u')
file.close()
```