

---

# **Flockcontext Documentation**

***Release 0.3.1***

**Antoine Cezar**

**Jul 03, 2019**



---

## Contents

---

<b>1</b>	<b>Flockcontext</b>	<b>3</b>
1.1	Usage . . . . .	3
1.2	License . . . . .	4
<b>2</b>	<b>Installation</b>	<b>5</b>
<b>3</b>	<b>Usage</b>	<b>7</b>
3.1	FlockOpen examples . . . . .	7
3.2	Flock examples . . . . .	8
<b>4</b>	<b>Contributing</b>	<b>9</b>
4.1	Types of Contributions . . . . .	9
4.2	Get Started! . . . . .	10
4.3	Pull Request Guidelines . . . . .	11
<b>5</b>	<b>Credits</b>	<b>13</b>
5.1	Development Lead . . . . .	13
5.2	Contributors . . . . .	13
<b>6</b>	<b>History</b>	<b>15</b>
<b>7</b>	<b>0.4.0 (2019-07-03)</b>	<b>17</b>
<b>8</b>	<b>0.3.1 (2015-08-24)</b>	<b>19</b>
<b>9</b>	<b>0.3.0 (2015-08-21)</b>	<b>21</b>
<b>10</b>	<b>0.2.0 (2015-08-20)</b>	<b>23</b>
<b>11</b>	<b>0.1.0 (2015-08-19)</b>	<b>25</b>
<b>12</b>	<b>Indices and tables</b>	<b>27</b>



Contents:



# CHAPTER 1

---

## Flockcontext

---

Improves `fcntl.flock` usage.

`flock` is a Unix command for [file locking](#), the mechanism that controls access restrictions of files.

## 1.1 Usage

Exclusive blocking lock:

```
from flockcontext import FlockOpen

with FlockOpen('/tmp/my.lock', 'w') as lock:
    lock.fd.write('Locked\n')
```

Exclusive blocking lock with 1 second timeout:

```
from flockcontext import FlockOpen

with FlockOpen('/tmp/my.lock', 'w', timeout=1) as lock:
    lock.fd.write('Locked\n')
```

Exclusive non-blocking lock:

```
from flockcontext import FlockOpen

try:
    with FlockOpen('/tmp/my.lock', 'w', blocking=False) as lock:
```

(continues on next page)

(continued from previous page)

```
        lock.fd.write('Locked\n')
except IOError as e:
    print('Can not acquire lock')
```

Shared blocking lock:

```
from flockcontext import Flock

with FlockOpen('/tmp/my.lock', 'w', exclusive=False) as lock:
    lock.fd.write('Locked\n')
```

Acquire and release within context:

```
from flockcontext import FlockOpen

with FlockOpen('/tmp/my.lock', 'w') as lock:
    print('Lock acquired')
    lock.fd.write('Locked\n')

    lock.release()
    print('Lock released')

    lock.acquire()
    print('Lock acquired')
    lock.fd.write('Locked\n')
```

Locking already opened file:

```
from flockcontext import Flock

with open('/tmp/my.lock', 'w') as fd:
    with Flock(fd):
        fd.write('Locked\n')
```

## 1.2 License

- Free software: BSD license



## CHAPTER 2

---

### Installation

---

At the command line:

```
$ pip install flockcontext
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv flockcontext  
$ pip install flockcontext
```



flockcontext provide two context manager for `fcntl.flock`. `Flock` locks an opened file while `FlockOpen` does the same job but opens the file for you.

### 3.1 FlockOpen exemples

Exclusive blocking lock:: python

```
from flockcontext import FlockOpen
with FlockOpen('/tmp/my.lock', 'w') as lock: lock.fd.write('Lockedn')
```

Exclusive non-blocking lock:: python

```
from flockcontext import FlockOpen
try:
    with FlockOpen('/tmp/my.lock', 'w', blocking=False) as lock: lock.fd.write('Lockedn')
except IOError as e: print('Can not acquire lock')
```

Shared blocking lock:: python

```
from flockcontext import Flock
with FlockOpen('/tmp/my.lock', 'w', exclusive=False) as lock: lock.fd.write('Lockedn')
```

Acquire and release within context:: python

```
from flockcontext import FlockOpen
with FlockOpen('/tmp/my.lock', 'w') as lock: print('Lock acquired') lock.fd.write('Lockedn')
    lock.release() print('Lock released')
    lock.acquire() print('Lock acquired') lock.fd.write('Lockedn')
```

## 3.2 Flock examples

Blocking lock:: python

```
from flockcontext import Flock
with open('/tmp/my.lock', 'w') as fd:
    with Flock(fd): fd.write('Lockedn')
```

Non blocking lock:: python

```
from flockcontext import Flock
with open('/tmp/my.lock', 'w') as fd:
    try:
        with Flock(fd, blocking=False): fd.write('Lockedn')
    except IOError as e: print('Can not acquire lock')
```

Shared lock:: python

```
from flockcontext import Flock
with open('/tmp/my.lock', 'w') as fd:
    with Flock(fd, exclusive=False): fd.write('Lockedn')
```

Acquire and release within context:: python

```
from flockcontext import Flock
with open('/tmp/my.lock', 'w') as fd:
    with Flock(fd) as lock: print('Lock acquired') fd.write('Lockedn')
    lock.release() print('Lock released')
    lock.acquire() print('Lock acquired') fd.write('Lockedn')
```

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

## 4.1 Types of Contributions

### 4.1.1 Report Bugs

Report bugs at <https://github.com/AntoineCezar/flockcontext/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

### 4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” is open to whoever wants to implement it.

### 4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

### 4.1.4 Write Documentation

Flockcontext could always use more documentation, whether as part of the official Flockcontext docs, in docstrings, or even on the web in blog posts, articles, and such.

### 4.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/AntoineCezar/flockcontext/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

## 4.2 Get Started!

Ready to contribute? Here's how to set up *flockcontext* for local development.

1. Fork the *flockcontext* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/flockcontext.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv flockcontext
$ cd flockcontext/
$ make develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. **When you're done making changes, check that your changes pass the tests::** \$ make test
6. **Check also that they pass the linter::** \$ pip install flake8 \$ make lint
7. **And that they pass other Python versions (requires <https://docs.docker.com/>):** \$ make test-all
8. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

9. Submit a pull request through the GitHub website.

## 4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.3, and 3.4, and for PyPy. Check [https://travis-ci.org/AntoineCezar/flockcontext/pull\\_requests](https://travis-ci.org/AntoineCezar/flockcontext/pull_requests) and make sure that the tests pass for all supported Python versions.





### 5.1 Development Lead

- Antoine Cezar <[antoine@cezar.fr](mailto:antoine@cezar.fr)>

### 5.2 Contributors

- Mathieu Leplatre <[contact@mathieu-leplatre.info](mailto:contact@mathieu-leplatre.info)>



## CHAPTER 6

---

History

---



## CHAPTER 7

---

0.4.0 (2019-07-03)

---

- Use docker for mutiple Python version testing from 2.7 to 3.7
- Add proper BSD licence file
- Add timeout for blocking lock
- Upgrade development status to Production/Stable



## CHAPTER 8

---

0.3.1 (2015-08-24)

---

- Add syntax highlighting for code examples
- Add Flock manager exemple in README





## CHAPTER 9

---

0.3.0 (2015-08-21)

---

- Add FlockOpen context manager.



## CHAPTER 10

---

0.2.0 (2015-08-20)

---

- Add Flock relase and acquire capability withing context.



## CHAPTER 11

---

0.1.0 (2015-08-19)

---

- Add Flock context manager.



## CHAPTER 12

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`