
aiothrottling

Release 0.0.4.post1

May 08, 2020

Contents

1	Introduction	3
1.1	Prerequisites	3
1.2	Installation	3
2	API	5
3	Examples	7
3.1	Setting the throttling rate	7
4	Indices and tables	11
	Python Module Index	13
	Index	15

Aiothrottling is a library to throttle Python coroutines. It uses asyncio under the hood.

Aiothrottling is a pure-Python throttling implementation using *asyncio*.

1.1 Prerequisites

Aiothrottling requires python ≥ 3.5 .

1.2 Installation

You can install the most recent aiothrottling release from pypi using pip or easy_install:

```
pip install aiothrottling
```


CHAPTER 2

API

Examples use *coroutines* exclusively.

3.1 Setting the throttling rate

The allowed coroutine call rate is determined by the `rate` argument. Pass the rate in the format `{limit}/{base period name}` or `{limit}/{factor}{base period name}`, for example

- **full period name**
 - `1/second`, `2/minute`, `3/hour`, `4/day`
- **short period name**
 - `4/s`, `5/m`, `6/h`, `7/d`
- **set custom period by using a factor**
 - `1/3s`, `12/37m`, `1/5h`, `8/3d`

3.1.1 Throttle

decorator

We can use `aiothrottling.Throttle` as decorator for coroutines:

```
from aiothrottling import throttle # Throttle alias
import time

@throttle(rate='1/s')
async def foo(n):
    print(n, time.time())
```

(continues on next page)

(continued from previous page)

```
for i in range(5):
    await foo(i)

# 0 1563272100.4413373
# 1 1563272101.4427333
# 2 1563272102.4441307
# 3 1563272103.445542
# 4 1563272104.4468124
```

awaitable

We can use `aiothrottling.Throttle` as awaitable object:

```
from aiothrottling import Throttle
import time

throttle = Throttle(rate='1/s')

async def foo(n):
    print(n, time.time())

for i in range(5):
    await throttle
    await foo(i)

# 0 1563275828.253736
# 1 1563275829.2547996
# 2 1563275830.2562528
# 3 1563275831.257302
# 4 1563275832.2587304
```

context manager

We can use `aiothrottling.Throttle` as context manager:

```
from aiothrottling import Throttle
import time

throttle = Throttle(rate='1/s')

async def foo(n):
    print(n, time.time())

for i in range(5):
    async with throttle:
        await foo(i)

# 0 1563275898.6722345
# 1 1563275899.673589
# 2 1563275900.6750457
# 3 1563275901.6763387
# 4 1563275902.6777005
```

3.1.2 Distributed throttle

CHAPTER 4

Indices and tables

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

a

`aiothrottling`, 5

A

`aiothrottling` (*module*), [5](#)