
zsv.ticker

Release 1.1.0

Tobias Dély

Nov 08, 2020

CONTENTS

1	Index	3
	Python Module Index	5
	Index	7

`zsv.ticker` enables flexible and idiomatic regular execution of tasks:

```
from zsv.ticker import Ticker

ticker = Ticker()
ticker.start(5)

while ticker.tick():
    execute_task()
```

Ticker aims to be more idiomatic and easy to use than a time calculation and sleep call, and further enables the instantaneous termination of a waiting task:

```
import signal
from time import sleep
from zsv.ticker import Ticker

ticker = Ticker()
ticker.start(5)

def abort(signum, frame):
    ticker.stop()

signal.signal(signal.SIGINT, abort)

while ticker.tick():
    print("tick")
    sleep(2)
    print("tock")
```

The above script wraps a *stop* call in a signal handler registered to SIGINT: hitting Ctrl+C after the script prints “tick” but before it prints “tock” will yield a final “tock” before it terminates.

class Ticker

Delivery of ticks at intervals.

Once started the *Ticker* will periodically make a boolean “tick” of True available through the *tick* method. Uncollected ticks will not stack or queue up, and the Ticker will continue to tick regardless. When stopped *tick* will return False, and any uncollected tick will be lost.

Example:

```
ticker = Ticker()
ticker.start(5)

while ticker.tick():
    execute_task()
```

start (*interval*, *immediate=False*)

Start the ticker.

Parameters

- **interval** (*int*) – Time between ticks.
- **immediate** (*bool*) – Start the ticker with a tick delivery.

Raises Exception if already running. –

Return type *None*

stop ()

Stop the ticker.

Return type *None*

tick ()

Wait for a tick to be delivered.

Will return immediately if ticker is stopped.

Return type *bool*

Returns True on tick, False if stopped.

PYTHON MODULE INDEX

Z

`zsv.ticker`, [1](#)

INDEX

M

module
 zsv.ticker, 1

S

start() (*Ticker method*), 3
stop() (*Ticker method*), 3

T

tick() (*Ticker method*), 3
Ticker (*class in zsv.ticker*), 3

Z

zsv.ticker
 module, 1