recurring Documentation

Release 1.0.0

Jeremiah Dodds

May 30, 2018

Contents

This is a simple library for running a function or callable every N seconds. It's meant for applications that need to schedule small, self-contined callable(s) on a relatively long, potentially changing period . alive-checks, state snapshots, that sort of thing.

CHAPTER 1

Use this if:

- You want to call something periodically over the lifetime of your application.
- You want to be able to change the time between calls.
- You want or need to avoid the overhead of `join`ing and `start`ing a thread every time. (up to 1/5 of a second according to my sample-size of one machine under no other load)
- The stuff you're going to call isn't going to destroy machines if it's killed abruptly at the end of the application's life.

CHAPTER 2

This is probably not appropriate for your project if:

- You're already using or likely will be using a fleshed-out concurrency framework.
- You have many things you'd like to repeatedly schedule and run.
- Your callables absolutely **must** execute some cleanup code to avoid disaster on kill.

This is not a library intended for top-level program composition.

CHAPTER 3

Installation:

```
pip install recurring
```

CHAPTER 4

Usage:

```
import recurring

def stuff():
    # do stuff ...

seconds_between_stuff = 30

job = recurring.job(stuff, seconds_between_stuff)
job.start()

# ...

seconds_between_stuff = 300000000 # this will be *from when rate is set*, not *from
↪the next scheduled call*
job.rate = seconds_between_stuff

# ...

# stop making calls until start() is called again
job.stop()

# some time later ....
job.start()

# stop making calls permanently
job.terminate()
job.start() # raises RuntimeError
job.rate = 3000 # raises RuntimeError
```

CHAPTER 5

Changelog

5.1 2.0.0 - 2018-05-30

- replaced `sched` backend with `threading.Timer`-like implementation, saving us from needing to respawn when a job's rate is changed.
- jobs can now be permanently stopped by calling `job.terminate()`

5.1.1 Backwards-Incompatible Changes

- `job.stop()` no longer takes an optional `timeout` argument

5.2 1.0.1 - 2018-05-24

- Corrected an assumption about the number of events that could be queued at once.

5.3 1.0.0 - 2018-05-22

- Initial release

CHAPTER 6

Indices and tables

- genindex
- modindex
- search