temporal-caching Documentation

Release 0.1.5

Tim Paine

Sep 07, 2023

Contents

Time-based cache invalidation

Build Status Coverage License PyPI Docs

CHAPTER 1

Install

From pip

```
pip install temporal-cache
```

Or from source

```
python setup.py install
```

CHAPTER 2

Why?

I needed something that would automagically refresh at 4:00pm when markets close.

```
    @expire(hour=16)
    def fetchFinancialData():
```

CHAPTER 3

Interval Cache

The interval cache expires every `time` interval since its first use

```
@interval(seconds=5, minutes=2)
def myfoo():
    '''myfoo's lru_cache will expire 2 minutes, 5 seconds after last use'''
```

CHAPTER 4

Expire Cache

The expire cache expires on the time given, in scheduler/cron style.

```
    @expire(second=5, minute=2)
    def myfoo():
        '''myfoo's lru_cache will expire on the second minute, fifth second of every␣
↪hour, every day, etc'''
```

CHAPTER 5

Caveats

Python hashing symantics persist. Dicts will be frozen, lists will be converted to tuples. Users are advised to pre-freeze to avoid issues.

CHAPTER 6

Development

See CONTRIBUTING.md for guidelines.

CHAPTER 7

License

This software is licensed under the Apache 2.0 license. See the LICENSE and AUTHORS files for details.

CHAPTER 8

API Documentation

temporalcache.expire.**daily**(*on=0*, *tz=None*, *maxsize=128*, *persistent=''*, *custom=None*, ***kwargs*)

temporalcache.expire.**expire**(*second=None*, *minute=None*, *hour=None*, *day=None*, *day_of_week=None*, *week=None*, *month=None*, *tz=None*, *maxsize=128*, *persistent=''*, *custom=None*, ***kwargs*)

Expires all entries in the cache @ whole number time

for example, @expire(0, 30, 16) will expire the cache at 4:30pm every day

temporalcache.expire.**hourly**(*on=0*, *tz=None*, *maxsize=128*, *persistent=''*, *custom=None*, ***kwargs*)

temporalcache.expire.**minutely**(*on=0*, *tz=None*, *maxsize=128*, *persistent=''*, *custom=None*, ***kwargs*)

temporalcache.expire.**monthly**(*on=0*, *tz=None*, *maxsize=128*, *persistent=''*, *custom=None*, ***kwargs*)

temporalcache.interval.**daily**(*maxsize=128*, *persistent=''*, *custom=None*, ***kwargs*)

temporalcache.interval.**hourly**(*maxsize=128*, *persistent=''*, *custom=None*, ***kwargs*)

temporalcache.interval.**interval**(*seconds=0*, *minutes=0*, *hours=0*, *days=0*, *weeks=0*, *months=0*, *years=0*, *maxsize=128*, *persistent=''*, *custom=None*, ***kwargs*)

Expires all entries in the cache every interval

temporalcache.interval.**minutely**(*maxsize=128*, *persistent=''*, *custom=None*, ***kwargs*)

temporalcache.interval.**monthly**(*maxsize=128*, *persistent=''*, *custom=None*, ***kwargs*)

class temporalcache.utils.**StorageBase**

Bases: object

cache_clear()

exception temporalcache.utils.**TCException**

Bases: Exception

temporalcache.utils.**calc**

temporalcache.utils.**disable**()

temporalcache.utils.**enable**()

temporalcache.utils.**should_expire**(*last*, *now*, *secondly=None*, *minutely=None*, *hourly=None*, *daily=None*, *day_of_week=None*, *weekly=None*, *monthly=None*)

should the cache expire? last - datetime now - datetime

if yearly: necessary_distance = calc(0, 0, 0, 0, 0, 0, yearly)

Python Module Index

t

Index