timeutils Documentation

Release 0.2.0

Michal Ciesielczyk

Sep 17, 2017

Contents:

CHAPTER 1

timeutils package

Submodules

timeutils.stopwatch module

Stopwatch

class timeutils.stopwatch.**Stopwatch**(*start=False*, *verbose=False*, *label=None*, *logger=None*, *logger_level=None*)

Provides a set of methods and properties that you can use to accurately measure elapsed time.

Parameters

- **start** (*bool, optional*) – if set to *True*, immediately starts measuring the time (by calling *start()*). By default set to *False*.
- **verbose** (*bool, optional*) – if set to *True*, logs the elapsed time when the *stop()* method is called. By default set to *False*.
- **label** (*str, optional*) – Optional stopwatch label to be included in the log messages (only if *verbose* is *True*).
- **logger** (logging.Logger, optional) – logger object for logging stopwatch messages if *verbose* is *True*. If set to *None* (the default), the logger is set to sys.stdout.
- **logger_level** (*int, optional*) – logging level as defined in the build-in logging package (only if the *logger* object is set).

Examples

Simple time measurement:

```
sw = Stopwatch(start=True)
# code to be measured
sw.stop()
```

Getting the elapsed time:

```
print(sw.elapsed)  # hh:mm:ss.ms
print(sw.elapsed.human_str())  # human-readable time
```

Restarting the stopwatch instance:

```
sw.restart()
```

Pausing and resuming the stopwatch:

```
sw.suspend()
# code block not included in the measurement
sw.resume()
```

Using a logger:

```
import logging

logger = logging.getLogger()
logger.setLevel(logging.INFO)
logger.addHandler(logging.FileHandler(filename='example.log'))

sw = Stopwatch(verbose=True, label='Example', logger=logger)
sw.start()
# code to be measured
sw.stop()
```

Note: *Stopwatch* methods are protected against inappropriate calls. It is an error to start or stop a *Stopwatch* object that is already in the desired state.

See also:

Documentation of the *TimeSpan* class.

elapsed
The total elapsed time measured by the current instance.

elapsed_seconds
The total elapsed time in fractions of a second measured by the current instance.

is_running
Indicates whether the *Stopwatch* instance is running.

is_suspended
Indicates whether the *Stopwatch* instance is suspended.

reset()
Stops time interval measurement and resets the *Stopwatch* instance. The time elapsed before reset is set to zero.

restart()
Stops time interval measurement, resets the *Stopwatch* instance, and starts measuring elapsed time. The time elapsed before restart is set to zero.

resume()
Resumes measuring elapsed time after calling *suspend()*.

start()
Starts measuring elapsed time for an interval.

start_time
The time at which the time measurement has been started.

stop()
Stops the time measurement. Returns the total elapsed time measured by the current instance.

stop_time
The time at which the time measurement has been stopped.

suspend()
Pauses the time measurement until the *Stopwatch* instance is resumed or stopped. Returns the total elapsed time measured by the current instance.

Call *resume()* to resume measuring elapsed time.

timeutils.timespan module

TimeSpan

class timeutils.timespan.**TimeSpan**(*t*)

days

hours

human_str(*trim_zeros=True*)
Returns a human-readable *TimeSpan* object, represented as time units such as days, hours, minutes, and seconds.

milliseconds

minutes

seconds

total_hours

total_milliseconds

total_minutes

total_seconds

Module contents

A set of methods and classes to accurately measure elapsed time.

See https://gitlab.com/cmick/timeutils for more information.

Examples

```
>>> from timeutils import Stopwatch
>>> sw = Stopwatch(start=True)
>>> sw.elapsed_seconds
16.282313108444214
```

```
>>> str(sw.stop())
'00:01:30.416'
>>> sw.elapsed.human_str()
'1 min, 30 secs'
```

See also:

Documentation of the *Stopwatch* class.

timeutils.**current_time_millis**()
Returns the difference, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

t

Index