
py-stopwatch Documentation

Release 0.1.1

Hrishikesh Terdalkar

May 17, 2022

CONTENTS:

1	Features	3
1.1	py-stopwatch	3
1.2	Installation	4
1.3	Usage	5
1.4	stopwatch	6
1.5	Contributing	7
1.6	Credits	9
1.7	History	10
2	Indices and tables	11
	Python Module Index	13
	Index	15

Stopwatch class for timing portions of python code.

- Free software: MIT license
- Documentation: <https://py-stopwatch.readthedocs.io>.

FEATURES

- Tick-based stopwatch
- Support for Pause/Resume
- Support for multiple named-ticks
- Utility functions for time between different ticks
- No third party requirements.

1.1 py-stopwatch

Stopwatch class for timing portions of python code.

- Free software: MIT license
- Documentation: <https://py-stopwatch.readthedocs.io>.

1.1.1 Features

- Tick-based stopwatch
- Support for Pause/Resume
- Support for multiple named-ticks
- Utility functions for time between different ticks
- No third party requirements.

1.1.2 Usage

```
from stopwatch import Stopwatch
t = Stopwatch()
t.start()
print("Started ..")
time.sleep(0.24)
print(f"t.tick(): {t.tick():.4f} seconds")
time.sleep(0.48)
print(f"t.tick(): {t.tick():.4f} seconds")
time.sleep(0.16)
print(f"t.tick('Named Tick-1'): {t.tick('Named Tick-1'):.4f} seconds")
t.pause()
print("Paused ..")
time.sleep(0.12)
t.resume()
print("Resumed ..")
print(f"t.last(): {t.last():.4f} seconds")
time.sleep(0.12)
print(f"t.tick(): {t.tick():.4f} seconds")
time.sleep(0.12)
print(f"t.tick('Named Tick-2'): {t.tick('Named Tick-2'):.4f} seconds")
t.stop()
print("Timer stopped.")
print("---")
print(f"Total pause: {t.time_paused:.2f} seconds.")
print(f"Total runtime: {t.time_active:.2f} seconds.")
print(f"Total time: {t.time_total:.2f} seconds.")
tij = t.time_elapsed(start_name='Named Tick-1', end_name='Named Tick-2')
print(f"Time between 'Named Tick-1' and 'Named Tick-2': {tij:.4f}")
```

1.2 Installation

1.2.1 Stable release

To install py-stopwatch, run this command in your terminal:

```
$ pip install py_stopwatch
```

This is the preferred method to install py-stopwatch, as it will always install the most recent stable release.

If you don't have `pip` installed, this [Python installation guide](#) can guide you through the process.

1.2.2 From sources

The sources for py-stopwatch can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/hrishikeshrt/py_stopwatch
```

Or download the [tarball](#):

```
$ curl -OJL https://github.com/hrishikeshrt/py_stopwatch/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

1.3 Usage

```
from stopwatch import Stopwatch
t = Stopwatch()
t.start()
print("Started ..")
time.sleep(0.24)
print(f"t.tick(): {t.tick():.4f} seconds")
time.sleep(0.48)
print(f"t.tick(): {t.tick():.4f} seconds")
time.sleep(0.16)
print(f"t.tick('Named Tick-1'): {t.tick('Named Tick-1'):.4f} seconds")
t.pause()
print("Paused ..")
time.sleep(0.12)
t.resume()
print("Resumed ..")
print(f"t.last(): {t.last():.4f} seconds")
time.sleep(0.12)
print(f"t.tick(): {t.tick():.4f} seconds")
time.sleep(0.12)
print(f"t.tick('Named Tick-2'): {t.tick('Named Tick-2'):.4f} seconds")
t.stop()
print("Timer stopped.")
print("---")
print(f"Total pause: {t.time_paused:.2f} seconds.")
print(f"Total runtime: {t.time_active:.2f} seconds.")
print(f"Total time: {t.time_total:.2f} seconds.")
tij = t.time_elapsed(start_name='Named Tick-1', end_name='Named Tick-2')
print(f"Time between 'Named Tick-1' and 'Named Tick-2': {tij:.4f}")
```

1.4 stopwatch

1.4.1 stopwatch package

Submodules

stopwatch.stopwatch module

Stopwatch class for timing portions of python code

class stopwatch.stopwatch.**Tick**(*time: float, action: str, name: str*)

Bases: object

time: float

action: str

name: str

class stopwatch.stopwatch.**Stopwatch**

Bases: object

Stopwatch Instance

A typical lifecycle of the stopwatch: [creation] → [start] → [tick, pause, resume] → [stop]

start()

Start the stopwatch

tick(*name=None*)

Record a tick

Return type Time since the last tick

pause()

Pause the stopwatch (Ticks are not recorded until resumed)

resume()

Resume

Return type Time for which the instance was paused

stop()

Stops the stopwatch

Return type Total time (including pause-time)

get_time_paused(*start_idx=0, end_idx=- 1*)

Get pause-time

property time_paused

Get pause-time

property time_active

property time_total

time_elapsed(*start_idx=0, end_idx=- 1, start_name=None, end_name=None, exclude_pause=True*)

Get time elapsed between different ticks

Parameters **exclude_pause** (*boolean*) – If True, pause-time is not counted. The default is True.

Return type Total runtime (with or without pause-time)

last()

Return the time between the last two ticks

current()

Return the time elapsed since the last tick

stopwatch.stopwatch.main()

Module contents

Stopwatch class for timing your python code with support for pause, resume and multiple named-ticks.

1.5 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

1.5.1 Types of Contributions

Report Bugs

Report bugs at https://github.com/hrishikeshrt/py_stopwatch/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

Write Documentation

py-stopwatch could always use more documentation, whether as part of the official py-stopwatch docs, in docstrings, or even on the web in blog posts, articles, and such.

Submit Feedback

The best way to send feedback is to file an issue at https://github.com/hrishikeshrt/py_stopwatch/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

1.5.2 Get Started!

Ready to contribute? Here’s how to set up *py_stopwatch* for local development.

1. Fork the *py_stopwatch* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your-username-here/py_stopwatch.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv py_stopwatch
$ cd py_stopwatch/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you’re done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 py_stopwatch tests
$ python setup.py test or pytest
$ tox
```

To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .  
$ git commit -m "Your detailed description of your changes."  
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

1.5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.5, 3.6, 3.7 and 3.8, and for PyPy. Check https://travis-ci.com/hrishikeshrt/py_stopwatch/pull_requests and make sure that the tests pass for all supported Python versions.

1.5.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_py_stopwatch
```

1.5.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bump2version patch # possible: major / minor / patch  
$ git push  
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

1.6 Credits

1.6.1 Development Lead

- Hrishikesh Terdalkar <hrishikeshrt@linuxmail.org>

1.6.2 Contributors

None yet. Why not be the first?

1.7 History

1.7.1 0.0.1 (2021-04-13)

- First release on PyPI.

INDICES AND TABLES

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

PYTHON MODULE INDEX

S

`stopwatch`, [7](#)

`stopwatch.stopwatch`, [6](#)

INDEX

A

`action` (*stopwatch.stopwatch.Tick* attribute), 6

C

`current()` (*stopwatch.stopwatch.Stopwatch* method), 7

G

`get_time_paused()` (*stopwatch.stopwatch.Stopwatch* method), 6

L

`last()` (*stopwatch.stopwatch.Stopwatch* method), 7

M

`main()` (*in module stopwatch.stopwatch*), 7

module

stopwatch, 7

stopwatch.stopwatch, 6

N

`name` (*stopwatch.stopwatch.Tick* attribute), 6

P

`pause()` (*stopwatch.stopwatch.Stopwatch* method), 6

R

`resume()` (*stopwatch.stopwatch.Stopwatch* method), 6

S

`start()` (*stopwatch.stopwatch.Stopwatch* method), 6

`stop()` (*stopwatch.stopwatch.Stopwatch* method), 6

`stopwatch`

 module, 7

`Stopwatch` (*class in stopwatch.stopwatch*), 6

`stopwatch.stopwatch`

 module, 6

T

`Tick` (*class in stopwatch.stopwatch*), 6

`tick()` (*stopwatch.stopwatch.Stopwatch* method), 6

`time` (*stopwatch.stopwatch.Tick* attribute), 6

`time_active` (*stopwatch.stopwatch.Stopwatch* property), 6

`time_elapsed()` (*stopwatch.stopwatch.Stopwatch* method), 6

`time_paused` (*stopwatch.stopwatch.Stopwatch* property), 6

`time_total` (*stopwatch.stopwatch.Stopwatch* property), 6