

---

# **pytest-logger**

***Release 0.1.4***

February 28, 2017



<b>1</b>	<b>Overview</b>	<b>3</b>
1.1	Installation . . . . .	3
1.2	Rationale . . . . .	3
1.3	Contributing . . . . .	3
<b>2</b>	<b>Usage</b>	<b>5</b>
2.1	Logging to stdout or files . . . . .	5
2.2	The logs directory layout . . . . .	6
2.3	Link to logs directory . . . . .	6
2.4	The logdir fixture . . . . .	7
2.5	API reference . . . . .	7
2.6	Command line options . . . . .	7
2.7	Configuration file parameters . . . . .	8



The pytest-logger is a [pytest](#) plugin which handles logs emitted by Python's [logging](#) module. Logs are output to terminal or files in configurable manner.



---

## Overview

---

## Installation

As simple as:

```
$ [sudo] pip install pytest-logger
```

Project is hosted on github:

<https://github.com/aurzenligl/pytest-logger>

## Rationale

I work with C++ application which logs copiously on its own and has multiple interfaces on which interesting events occur. Getting all this information as separate files in directory tree or filtered real-time logs proves to be invaluable in testing.

Unfortunately, contemporary state of pytest and plugins doesn't allow to do this out-of-the-box:

- real-time output to terminal which doesn't mix with regular pytest output ('-v' or not)
- possibility to enable and set levels on per logger basis
- test session logs persistent in filesystem and partitioned in fine-grained manner (per testcase and per logger)

Above problems require reacting on events such as session start, test start/teardown and inspecting some data stored by framework, e.g. test locations/names. This requires writing pytest plugin.

Plugin has a hook API, which means that if user doesn't implement hooks, nothing happens, and if he does - any cmdline options and logging configuration logic may be envisioned.

## Contributing

Contributions are welcome!

If you:

- find a bug in plugin,
- find important features missing,
- want to propose a new feature,

- or just like it,

please write to [github issues](#) or let me know [via email](#).



---

## Usage

---

By default `pytest-logger` does nothing in scope of logging. It's only default action is to add new hooks and fixture. Terminal output, logfiles together with their directory structure, symlink appear only when explicitly requested by user via one of hooks or fixture.

### Logging to stdout or files

Implement `pytest-logger` hooks in your `confest.py` to direct logs to terminal or files. You can make up your own cmdline arguments relevant to your test environment for convenience.

Terminal output mixes with normal `pytest`'s output in graceful manner. File output is stored in logs directory (see *logs dir layout* and *link to logs dir*).

```
import logging
def pytest_logger_stdoutloggers(item):
    # handles foo logger at chosen level and bar at all levels
    return [('foo', logging.WARN), 'bar']

def pytest_logger_fileloggers(item):
    # handles root logger
    return ['']
```

- see `LoggerHookspec.pytest_logger_stdoutloggers()`
- see `LoggerHookspec.pytest_logger_fileloggers()`

Things to note:

- **stdout capturing:** in order to see logs printed on terminal in real time during test execution, you need to disable output capturing by `-s` switch.
- **default root level:** by default root logger (and all its children) has warning level threshold set. This can filter logs regardless of handler levels user gives via hooks. Be sure to set root logger level as `NOTSET` if you don't want this to happen.
- **no handlers warning:** if log wouldn't get filtered, but there are no handlers added to logger, `unwanted message` is printed. Add `NullHandler` to such loggers.
- **default logging config:** if root logger has no handlers, using module level logging functions will setup basic logging config. It makes no sense in combination with this plugin. Be sure root logger has some handler (at least `NullHandler`) or just don't use these functions.
- **pytest-xdist:** stdout output is not printed to terminal in `pytest-xdist` runs. File output works as in single process mode.

## The logs directory layout

### Directory with logfiles is located

- under test session's `basetemp` directory named “logs”

```
tmp/
-- pytest-of-aurzenligl
  -- pytest-0
  |   -- logs
  |   (... )
  -- pytest-1
  |   -- logs
  |   (... )
  -- pytest-2
  |   -- logs
  |   (... )
```

or

- under predefined location, if `-logger-logsdire` option or `logger_logsdire` entry in configuration file defined

```
<logger_logsdire>/
-- (...)
```

It has structure following pytest test item's `nodeid`.

- test directories are directories
- test filenames are directories
- test functions are directories (each parametrized testcase variant is distinct)
- each registered logger is a file (root logger is called 'root')

```
logs/
-- classtests
|   -- test_y.py
|       -- TestClass.test_class
|       -- daemon
|       -- setup
-- parametrictests
|   -- test_z.py
|       -- test_param-2-abc
|       -- test_param-4.127-de
|       -- setup
-- test_p.py
  -- test_cat
  -- proc
```

## Link to logs directory

Implement link hook to have access to logfiles from place where you regularly run your tests. Link is created in a race-safe manner, even if multiple pytest processes run tests simultaneously.

```
# content of confetest.py
import os
def pytest_logger_logdirlink(config):
    return os.path.join(os.path.dirname(__file__), 'logs')
```

```
$ ls -o
total 80
drwxr-xr-x  9 aurzenligl 4096 Dec 22 21:09 .
drwxr-xr-x 28 aurzenligl 4096 Dec 14 23:33 ..
-rwxr-xr-x  1 aurzenligl 3028 Dec 11 02:18 confptest.py
lrwxrwxrwx  1 aurzenligl   39 Dec 22 21:09 logs -> /tmp/pytest-of-aurzenligl/pytest-2/logs
-rwxr-xr-x  1 aurzenligl  817 Dec 11 02:13 test_x.py
```

- see `LoggerHookspec.pytest_logger_logdirlink()`

## The logdir fixture

Like pytest's `tmpdir` it's a `py.path.local` object which offers `os.path` methods. Points to logs subdirectory related to particular testcase. Directory is ensured to exist and custom log files can be written into it:

```
def test_foo(logdir):
    logdir.join('myfile.txt').write('abc')
```

## API reference

**class** `pytest_logger.plugin.LoggerHookspec`

**pytest\_logger\_fileloggers** (*item*)

called before testcase setup. If implemented, given loggers will emit their output to files within logs temporary directory.

**Parameters** *item* – test item for which handlers are to be setup.

**Return list** List should contain logger name strings or tuples with logger name string and logging level.

**pytest\_logger\_logdirlink** (*config*)

called after cmdline options parsing. If implemented, symlink to logs directory will be created.

**Parameters** *config* – pytest config object, holds e.g. options

**Return string** Absolute path of requested link to logs directory.

**pytest\_logger\_stdoutloggers** (*item*)

called before testcase setup. If implemented, given loggers will emit their output to terminal output.

**Parameters** *item* – test item for which handlers are to be setup.

**Return list** List should contain logger name strings or tuples with logger name string and logging level.

## Command line options

`-logger-logsdir=<logsdir>`: where `<logsdir>` is root directory where log files are created

## Configuration file parameters

*logger\_logdir*=<logdir>: where <logdir> is root directory where log files are created

## L

LoggerHookspec (class in `pytest_logger.plugin`), 7

## P

`pytest_logger_fileloggers()`  
(`pytest_logger.plugin.LoggerHookspec`  
method), 7

`pytest_logger_logdirlink()`  
(`pytest_logger.plugin.LoggerHookspec`  
method), 7

`pytest_logger_stdoutloggers()`  
(`pytest_logger.plugin.LoggerHookspec`  
method), 7